THESE DE DOCTORAT

UNIVERSITE PARIS VI – PIERRE ET MARIE CURIE

ECOLE DOCTORALE D'INFORMATIQUE, TELECOMUNICATIONS
ET ELECTRONIQUE (EDITE) DE PARIS

Spécialité **Informatique**

Préparée à l'IRCAM
par

Grégoire Carpentier

En vue de l'obtention du titre de

DOCTEUR DE L'UNIVERSITE PIERRE ET MARIE CURIE

Approche computationnelle de l'orchestration musicale

Optimisation multicritère sous contraintes de combinaisons instrumentales dans de grandes banques de sons

Thèse soutenue le 16 décembre 2008

devant le jury composé de :

Emmanuel Saint-James	Co-directeur	LIP6-CNRS
Gérard Assayag	Co-directeur	IRCAM-CNRS
Myriam Desainte-Catherine	Rapporteure	Université de Bordeaux-1
Narendra Jussien	Rapporteur	Ecole des Mines de Nantes
Patrice Perny	Président	LIP6-CNRS
Camilo Rueda	Examinateur	Pontifica Universidad Javeriana-Cali, Colombie
Stephen McAdams	Examinateur	McGill University, Montréal
Fabien Lévy (compositeur)	Examinateur	Columbia University, New-York

Thèse préparée à l'IRCAM
(Institut de recherche et de coordination acoustique/musique)

Sous la direction de Gérard Assayag et Emmanuel Saint-James

IRCAM-CNRS UMR STMS 9912
Equipe Représentations musicales
1 place Igor Stravinsky
F-75004 Paris – France

Merci. . .

. . . à Gérard Assayag pour sa confiance et son soutien, à Emmanuel Saint-James pour ses conseils et sa précieuse relecture du manuscrit ;

. . . à Hugues Vinet pour son accueil au sein du département scientifique de l'IRCAM, à Arshia Cont pour son enthousiasme et pour les rencontres qui n'auraient pas eu lieu sans lui ;

. . . à Yan Maresz pour son implication et sa foi dans le projet Orchestration, ainsi qu'à Damien Tardieu pour la richesse de nos échanges ;

. . . à Georges Bloch pour sa sagesse et sa bienveillance ;

. . . aux nombreux compositeurs avec qui j'ai collaboré, Marco Suarez, Oscar Bianchi, Jonathan Harvey, Joshua Fineberg, Gérard Buquet, Mauro Lanza, Tristan Murail, Raphaël Cendo ;

. . . au personnel de l'IRCAM, au service informatique, à l'équipe Représentations musicales, aux doctorants, aux gars du bureau A107, et à tous ceux qui font vivre l'institut au quotidien, Sylvie, Carole, Didier, Bruno, Michel ;

. . . à mes parents, mes grand-parents, à tous ceux qui m'ont soutenu, à Sigrid.

Table des matières

Résumé

De toutes les composantes de l'écriture musicale, l'orchestration — ou l'art d'assembler les timbres intrumentaux — est longtemps demeurée, dans son enseignement comme dans sa pratique, une activité empirique. La difficulté de formaliser de manière rigoureuse l'ensemble des techniques inhérentes à cette discipline fait qu'aujourd'hui encore, l'orchestration reste un domaine peu abordé par l'informatique musicale et l'aide à la composition.

Les rares outils actuels ramènent le problème de l'orchestration à la découverte, au sein de banques d'échantillons sonores instrumentaux, de combinaisons approchant au mieux un timbre fixé par le compositeur. Cette approche sera également la nôtre. Mais là où les méthodes actuelles contournent systématiquement le problème combinatoire de l'orchestration par le recours à des principes de décomposition ou à des algorithmes de *matching pursuit*, l'originalité de notre démarche est de placer les enjeux combinatoires au cœur de nos travaux et de traiter l'orchestration à la mesure de sa complexité.

Envisageant tout d'abord la question comme un problème de sac à dos multi-objectifs, nous montrons que les non-linéarités dans les modèles de perception du timbre imposent un cadre théorique plus large pour l'aide à l'orchestration. Nous proposons une formalisation générique et extensible en nous plaçant dans un cadre de recherche combinatoire multicritère sous contraintes, dans lequel plusieurs dimensions perceptives sont optimisées conjointement pour approcher un timbre cible défini par le compositeur. Nous validons dans un premier temps notre approche théorique en montrant, sur un ensemble de problèmes de petite taille et pour une caractérisation exclusivement spectrale du timbre, que les solutions du problème formel correspondent à des propositions d'orchestration pertinentes. Nous présentons alors un algorithme évolutionnaire permettant de découvrir en un temps raisonnable un ensemble de solutions optimales. S'appuyant sur la prédiction des propriétés acoustiques des alliages instrumentaux, cette méthode propose des solutions d'orchestration en fonction de critères perceptifs et encourage ainsi la découverte de mélanges de timbres auxquels le savoir et l'expérience n'auraient pas nécessairement conduit. En outre, la recherche peut-être à tout moment orientée dans une direction privilégiée. Parallèlement, nous définissons un cadre formel pour l'expression de contraintes globales et introduisons une métaheuristique innovante de résolution, permettant de guider la recherche vers des orchestrations satisfaisant un ensemble de propriétés symboliques en lien direct avec l'écriture musicale.

Nous présentons enfin un prototype expérimental d'outil d'aide à l'orchestration utilisable directement par les compositeurs, dans lequel l'exploration des possibilités de timbres est facilitée à travers une représentation multi-points de vue des solutions et un mécanisme interactif des préférences d'écoute. Nous terminons avec une série d'exemples d'application de nos travaux à des problèmes compositionnels concrets.

Mots clés : *Orchestration, composition assistée par ordinateur, informatique musicale, timbre, description du signal, descripteurs perceptifs, optimisation multicritère, algorithmes évolutionnaires, programmation par contraintes, contraintes globales, recherche locale.*

Abstract

Among all techniques of musical composition, orchestration has never gone further than an empirical activity. Practicing and teaching orchestration — the art of blending instrument timbres together — involve hard-to-formalize knowledge and experience that computer music and composition systems have for years stayed away from.

The state-of-the-art orchestration tools search for sound combinations within instrument sample databases that best match a target timbre defined by the composer. To this end, those methods use either decomposition or matching pursuit algorithms, and therefore circumvent to the combinatorial problem of orchestration. We propose in this thesis an original approach for the discovery of relevant sound combinations, in which we explicitly address combinatorial issues and tackle orchestration in its inner complexity.

Initial considerations of the problem in a multiobjective knapsack framework shows that non-linearity and non-additivity of objective functions require a wider theoretic approach. We suggest a generic and easily extendible formalization of orchestration as a constrained multiobjective search towards a target timbre, in which several perceptual dimensions are jointly optimized. We first validate our approach on a small-size problems test set, with a rather simple, spectral-based timbre description. We show that theoretic solutions of the optimization problem correspond to perceptually relevant orchestration proposals. We then introduce a time-efficient evolutionnary orchestration algorithm allowing the discovery of optimal solutions. By estimating acoustic properties of sound mixtures, our method suggests orchestration proposals in relation with perceptual criteria and favors the exploration of somehow non-intuitive sound mixtures. From there, the search may be pursued in specific directions. To enhance the control of symbolic features in orchestrations proposals, we define a formal framework for global constraints specification and introduce an innovative repair metaheuristic. Thanks to this method the search is led towards regions fulfulling a set of musical-related requirements.

We finally present a composer-friendly computer-aided orchestration prototype, in which timbre space exploration is encouraged by a multiple viewpoints representation and an interactive mecanism for guessing the composer's listening preferences. We end this thesis with relevant application examples in real musical works.

Key words : *Orchestration, computer-aided composition, computer music, audio signal description, perceptual audio features, multi-objective optimization, evolutionary algorithms, constraint programming, global constraints, local search.*

Introduction

La musique est un « art-science » dont l'aspect en tant qu'art change nécessairement lorsque la science lui offre de nouvelles possibilités structurelles et que la vie réclame d'elle ce dont elle a besoin.

Edgar Varèse

Nos travaux sont nés d'une problématique d'écriture musicale, terrain délicat pour la recherche scientifique qui touche là une technique dont l'appréciation est soumise à un jugement esthétique, et dont les mécanismes paraissent difficilement formalisables. Au yeux des néophytes, l'informatique musicale est souvent perçue comme un non-sens, et la question des rapports entre musique et ordinateur, déplacée. Pourtant — Varèse l'exprime à travers la notion d'« art-science » — la musique n'est pas un simple écran de projection pour les techniques, mais une entité bicéphale où l'art et la technologie ne cessent de s'interroger.

A ceux qui avancent que les compositeurs n'ont pas besoin d'un ordinateur pour écrire, nous pourrions répondre que l'homme n'a pas besoin non plus d'un système symbolique complexe de notation, ni même d'instruments pour produire de la musique. Il lui suffit d'ouvrir la bouche. Mais depuis les temps primitifs où le chant était la seule pratique musicale, les sociétés ont évolué vers des formes bien plus complexes, et la musique ne peut faire autrement que cristalliser cette complexité. En témoignent aujourd'hui la richesse des formes et des pratiques d'expression musicale, ainsi que la récurrence avec laquelle sont mis en question les rapports entre musique et société.

L'histoire de la musique est jalonnée de ruptures qui marquent ces moments critiques où les ressources du système musical ne suffisent plus à pourvoir son « besoin de complexité ». La musique se tourne alors vers la technologie, dont l'apport n'est pas synonyme d'un appauvrissement des compétences, mais au contraire d'un enrichissement du vocabulaire et des pratiques. En retour la recherche scientifique, lorsqu'elle se penche sur des problématiques musicales, n'a d'autre choix que de l'aborder dans sa complexité. « Imprégnée » de la complexité du monde contemporain, la musique sollicite la science tout autant qu'elle la défie : la seconde peut-elle « penser » la complexité comme la première la reflète ?

Les problématiques musicales qui nous intéressent ici concernent une discipline particulière de l'écriture : l'orchestration, ou l'art de mélanger les timbres, d'allier les potentialités sonores des instruments pour produire des « couleurs » particulières. Aujourd'hui, l'orchestration demeure le seul domaine de l'écriture musicale à n'être que très peu abordé par l'informatique, alors que les compositeurs réclament depuis plusieurs décennies un outil d'exploration et de contrôle du timbre orchestral qui tiennent compte de la formidable extension, depuis le début du XX^{e} xiècle, des possibilités instrumentales. D'après nous, cette lacune tient à la difficulté de

formaliser les relations complexes entre les variables symboliques de l'écriture et les propriétés perceptives des mélanges instrumentaux. A y regarder de près, la complexité de l'orchestration se décline selon trois axes : complexité combinatoire, complexité de la perception du timbre, et complexité de l'évolution temporelle. Si quelques rares systèmes d'aide à l'orchestration existent aujourd'hui, aucun d'entre eux ne traite ne serait-ce que l'une de ces dimensions : le temps n'est jamais modélisé, le problème combinatoire est toujours contourné, le timbre est réduit à une unique dimension.

Le travail que nous présentons dans cette thèse a été motivé par le désir d'aborder, autant que faire ce pouvait, le problème de l'orchestration dans toute sa complexité.

0.1 Démarche scientifique

Sensible à la demande insistante des compositeurs de disposer, au sein d'un environnement de composition assistée par ordinateur, d'un outil puissant pour le contrôle et d'exploration du timbre orchestral, l'IRCAM[1] a encouragé il y a quelques années la création d'un groupe de travail réunissant chercheurs et compositeurs sur ce sujet. Il en est ressorti un « cahier des charges » définissant un ensemble de priorités de recherche à court et moyen terme. Devant l'ampleur des travaux qu'un domaine si vaste et si complexe ouvrait, deux thèses de doctorat sur l'aide à l'orchestration ont débuté à l'IRCAM durant l'année 2005, et furent conduites en parallèle. La présente thèse a été menée dans l'équipe Représentations musicales de l'IRCAM (en charge notamment du développement d'outils avancés pour la composition assistée par ordinateur), en collaboration étroite avec Damien Tardieu, doctorant dans l'équipe Analyse-synthèse (dont les travaux sont, entre autres, consacrés à la description et la modélisation du signal audio).

Cette séparation des tâches était indispensable pour aborder le problème de l'orchestration assistée par ordinateur selon différents axes de complexité. Les travaux de Damien Tardieu ont essentiellement consisté en l'élaboration d'un cadre formel adapté à la caractérisation du timbre instrumental et orchestral. Il s'agissait d'abord d'identifier un ensemble de dimensions perceptives pertinentes pour la description du timbre, puis de trouver un moyen d'extraire automatiquement ces propriétés à partir de grandes banques d'échantillons, pour enfin mettre au point un modèle capable de prédire les caractéristiques acoustiques de mélanges instrumentaux. Ce travail a débouché sur l'élaboration de *modèles d'instruments* qui permettent d'estimer la probabilité qu'une combinaison instrumentale possède une caractéristique du timbre désiré par le compositeur.

Nous nous sommes quant à nous concentré sur les difficultés liées aux aspects combinatoires du problème. L'orchestration est en effet l'art subtil et délicat des alliages instrumentaux, et il suffit de considérer l'éventail de timbres couvert par chaque instrument pour se faire une idée de l'étendue des possibilités sonores offertes par un orchestre. Une telle explosion combinatoire interdisant le recours à des méthodes complètes de résolution, nous nous sommes tourné du côté des métaheuristiques dont l'utilisation de plus en plus fréquente ces dernières années en optimisation combinatoire et en programmation par contraintes a permis de traiter des problèmes très complexes.

Menées en parallèle, ces deux approches de l'orchestration ont permis de d'aborder deux dimensions de sa complexité : celle inhérente au caractère multidimensionnel de la perception du timbre et celle liée aux enjeux combinatoires. Les aspects temporels du timbre, où se

[1] Institut de recherche et de coordination acoustique/musique. `http://www/ircam.fr/`

conjuguent verticalité et horizontalité de l'écriture[2], ont été dans un premier temps laissés de côté. Ils n'auront été abordés dans nos travaux respectifs que de manière partielle ou indirecte, dans la limite de ce que les cadres théoriques dans lequels nous traitons le problème statique du timbre pouvaient supporter.

En théorie, le développement d'un algorithme d'optimisation combinatoire ne pouvait débuter qu'une fois les recherches sur la modélisation du timbre instrumental parvenues à un degré suffisant de maturité. Or, les deux thèses ayant été menés simultanément, il nous était impossible d'attendre le terme de ce travail de modélisation. Nous avons donc élaboré la stratégie décrite ci-dessous.

Le timbre vu comme un ensemble de descripteurs du signal

Après avoir posé le problème de l'aide à l'orchestration comme la recherche de combinaisons de sons s'approchant le plus possible d'un timbre défini par le compositeur, nous avons fait le choix d'un cadre théorique dans lequel traiter rapidement le problème de la caractérisation du timbre. Nous avons opté pour une approche par descripteurs, qui consiste à calculer directement sur le signal audio une information liée aux propriétés perceptives du son. On peut ainsi, à partir de grandes banques d'échantillons sonores, parvenir à une connaissance synthétique et formalisée des possibilités de timbres offertes par les instruments. N'étant toutefois pas spécialiste du domaine, nous nous sommes restreint à des descripteurs exclusivement spectraux, et pour lesquels la construction de « fonctions d'addition » était accessible à notre expertise. En effet, il ne suffit pas d'extraire les propriétés sonores d'échantillons instrumentaux, encore faut-il pouvoir calculer ces propriétés pour des combinaisons de sons. Nous avons retenu trois descripteurs spectraux pour la caractérisation du timbre instrumental, et avons montré que leurs « fonctions d'addition » respectives permettaient bien d'estimer les descripteurs de mixtures instrumentales.

L'orchestration comme recherche multicritère de combinaisons

Le problème de l'orchestration a alors pu être reformulé : à partir d'une « cible sonore » définie par un ensemble de descripteurs, comment découvrir les combinaisons de sons dont les descripteurs s'approchent le plus possible de ceux de la cible ? Le timbre étant décrit dans un espace multidimensionnel, nous avons montré que les distances perceptives selon chaque dimension ne pouvaient être agrégées en une valeur unique et qu'il fallait nous résoudre à une approche multicritère, dans laquelle les différentes dimensions du timbre sont optimisées conjointement. Nous avons alors montré sur des instances de petite taille et pour une description réduite du timbre que les solutions optimales du problème d'optimisation sont bien des propostions pertinentes d'un point de vue perceptif. Nous nous sommes ainsi assuré que le problème de l'orchestration, formulé comme une tâche d'optimisation multicritère, était « bien posé ».

[2]Ces deux dimensions de l'écriture se réfèrent au plan de la partition. L'écriture verticale désigne l'agencement des différentes voix (c'est-à-dire les instruments de l'orchestre) à un instant donné, là où l'écriture horizontale concerne le développement dans le temps d'une seule voix.

Recherche de combinaisons à l'aide de métaheuristiques

La recherche de sous-ensembles limités par une contrainte de capacité — celle liée au nombre restreint d'instruments à disposition dans l'orchestre — nous a d'abord incité à nous ramener à un problème de sac à dos, pour lequel il existe une multitude d'approches et de méthodes de résolution. Malheureusement le caractère fortement non-linéaire et non-additif de nos fonctions objectifs nous a imposé un cadre théorique plus large. Nous avons alors introduit une représentation des propositions d'orchestration facilement manipulable par une métaheuristique d'optimisation, grâce à laquelle les contraintes liées à l'effectif orchestral sont implicitement satisfaites. Nous avons par ailleurs conçu un algorithme génétique multicritère permettant de découvrir en un temps raisonnable un ensemble diversifié de propositions efficientes d'orchestration. Notre algorithme utilise un principe d'agrégation des critères à l'aide de fonctions scalarisantes aléatoires, favorisant ainsi l'optimisation de la population selon plusieurs critères perceptifs considérés conjointement. Un mécanisme d'interaction avec le compositeur permet en outre d'inférer un certain nombre d'hypothèses sur ses préférences implicites d'écoute et de guider la recherche vers une région privilégiée de l'espace. Afin de faciliter l'intégration des solutions optimales dans un processus compositionnel de plus grande envergure, nous avons imaginé un langage élémentaire pour la spécification de contraintes globales sur les variables symboliques de l'écriture musicale. Nous avons ensuite proposé une métaheuristique innovante de satisfaction des contraintes, basée sur la distinction entre contraintes de *design* et contraintes de *conflit*. Nous avons enfin montré comment le problème du temps en orchestration pouvait être abordé à l'aide de contraintes. Toutes ces méthodes ont été validées sur différentes classes de problèmes d'orchestration par un processus d'évaluation multi-expert.

Développement d'un prototype

En parallèle de nos recherches, nous avons tiré parti de nos multiples échanges avec les compositeurs pour concevoir et développer un prototype d'outil d'aide à l'orchestration. Celui-ci a permis à la fois de valider expérimentalement notre approche computationnelle de l'orchestration et d'identifier un certain nombre de pistes de recherches futures. Si les restrictions que nous nous sommes imposées dans le cadre de ce travail confinent inévitablement ce prototype à une certaine esthétique musicale, la satisfaction des compositeurs l'ayant utilisé n'en constitue pas moins la marque d'une réussite dont nous sommes fier.

Il nous faut encore signaler ici qu'à l'heure où nous écrivons ces lignes, l'intégration au sein de notre outil du travail de Damien Tardieu sur les modèles d'instruments n'a pas encore été effectuée. Toutefois notre système, bien que basé sur une connaissance réduite des possibilités instrumentales, nous a déjà permis d'obtenir des résultats significatifs, laissant augurer des performances décuplées grâce aux modèles d'instruments. Aussi ne manquerons-nous pas de discuter, en divers endroits de cette thèse, leur apport potentiel pour l'aide à l'orchestration.

0.2 Structure du manuscrit

Ce document est organisé en trois grandes parties. La première partie est un *état des arts* qui, en guise d'entrée en matière dans une recherche pluridisciplinaire, commence par interroger (chapitre 1) la nature des rapports entre musique et technologie ainsi que les enjeux de l'informatique musicale. La question du timbre est soulevée au chapitre 2. De son

émergence comme fonction essentielle de l'écriture musicale au XX^{e} siècle jusqu'à sa compréhension à travers des études perceptives psychoacoustiques, et sa caractérisation par l'analyse et la description du signal audio, un éclairage multiple est porté sur la complexité de cet objet d'étude. Le chapitre 3 traite de l'orchestration musicale proprement dite, expose les fondements et les limites de sa pratique traditionnelle et tente de dégager le rôle potentiel de l'ordinateur dans une vision contemporaine de l'orchestration. Nous montrons ensuite (chapitre 4) en quoi la rencontre entre, d'une part, le besoin actuel des compositeurs, et d'autre part des technologies et des savoirs suffisament matures, ouvre un vaste champ de recherche pour l'orchestration assistée par ordinateur. Nous présentons les contributions récentes dans le domaine, exposons leurs apports et leurs limites et discutons en quoi ces dernières peuvent être dépassées. Au chapitre 5, nous entrons au cœur de notre sujet de recherche : nous introduisons les concepts fondamentaux des approches multicritères et proposons un état de l'art des méthodes d'optimisation combinatoire, en portant une attention toute particulière aux algorithmes génétiques. Nous détaillons en outre deux concepts récents dont s'inspire notre algorithme de recherche d'orchestrations, l'agrégation des critères par fonctions scalarisantes de Tchebycheff et l'estimation de la densité locale au sein de la population par la méthode PADE (*Population size Adaptive Density Estimation*). Enfin, le chapitre 6 est consacré au paradigme de programmation par contraintes et aux méthodes associées, ainsi qu'à leur utilisation en informatique musicale. Là encore, nous approfondissons deux heuristiques de recherche locale ayant influencé nos travaux, la recherche adaptative et l'heuristique $\mathcal{CN}$-Tabou.

La seconde partie de ce manuscrit est consacrée à notre propre contribution au problème de l'aide à l'orchestration. Nous commençons par poser, au chapitre 7, un cadre formel générique grâce auquel l'orchestration musicale peut être saisie selon deux axes de complexité. Le premier est celui de l'explosion combinatoire des possibilités sonores au sein de l'orchestre, le second celui du caractère multidimensionnel de la perception du timbre. Nous validons cette approche au chapitre 8 sur des instances de complexité réduite. Nous calculons pour cela les solutions optimales de problèmes de petite taille et de faible dimensionnalité et montrons qu'elles correspondent à des alliages pertinents d'un point de vue perceptif. Un algorithme génétique multicritère de recherche d'orchestrations est alors proposé au chapitre 9. Basé sur une agrégation aléatoire des critères, il permet de chercher des solutions selon différentes préférences d'écoute. Nous le modifions au chapitre 10 afin de prendre en compte des contraintes additionnelles. Nous introduisons un cadre formel pour l'expression de contraintes symboliques globales ainsi qu'une heuristique de résolution basée sur la distinction entre *contraintes de design* et *contraintes de conflit*. Enfin, nous présentons au chapitre 11 une évaluation multi-expert et itérative de notre méthode de recherche. L'intérêt de chaque module de l'algorithme y est discuté selon plusieurs critères de performance.

Un prototype expérimental d'outil d'orchestration fait l'objet de la troisième partie. Nous en présentons l'architecture générale au chapitre 12, montrons comment il s'articule avec les technologies existantes, en discutons les limitations techniques et esthétiques actuelles. Au chapitre 13, un scénario d'utilisation retrace un processus complet d'aide à l'orchestration, depuis la définition du timbre souhaité jusqu'à la partition finale. L'accent est porté sur l'interaction avec le compositeur et la découverte de ses préférences d'écoute. Le chapitre 14 rapporte enfin cinq exemples concrets d'utilisation de notre outil, dont certains sont issus de collaborations étroites avec des compositeurs et ont été utilisés dans l'écriture de leurs œuvres.

0.3 Grilles de lectures

Selon les connaissances du lecteur et son intérêt pour l'un ou l'autre des thèmes abordés, de nombreux parcours de lecture sont possibles. Nous fléchons quelques itinéraires :

- Le lecteur moins sensible aux problématiques musicales contemporaines et au contexte esthétique dans lequel s'inscrit cette recherche peut se rendre directement au chapitre 4 qui, sans rentrer encore dans des considérations techniques, définit le cadre général de nos travaux.
- Les compositeurs sont tout particulièrement concernés par la troisième partie. Dans la première, ils peuvent se contenter du chapitre 4, dans la seconde du chapitre 7, qui suffit pour saisir notre approche formelle de l'aide à l'orchestration.
- Pour le lecteur familier de la programmation par contrainte et de l'optimisation combinatoire multicritère, la fin du chapitre 2 (en particulier les sections 2.5 et 2.6) et le chapitre 4 doivent suffire pour saisir les enjeux et les difficultés de notre sujet.
- Le psychoacousticien ou le chercheur concerné par des problématiques de perception auditive est invité se rendre aux chapitres 2, 7 et 8, ainsi bien sûr qu'à la partie III.
- Enfin, le chapitre 11 s'adresse en priorité aux lecteurs familiers des méthodes multicritères et de l'évaluation de leurs performances.

En toutes circonstances, la lecture du chapitre 4 reste indispensable pour situer notre démarche par rapport à la demande des compositeurs et aux solutions scientifiques qui leur sont aujourd'hui offertes. Quant au chapitre 7, il condense l'essentiel de notre contribution sur le plan théorique au problème de l'aide à l'orchestration. Le lecteur souhaitant reposer ses yeux pourra en outre écouter tous les exemples de la partie III (et bien d'autres) sur le site : `http://recherche.ircam.fr/equipes/repmus/carpentier/phdsounds/`.

0.4 Contributions

La contribution de notre travail à la littérature se résume en sept points :

- une formalisation générique et extensible du problème de l'aide à l'orchestration (chapitre 7) comme une tâche d'optimisation multicritère sous contraintes ;
- une validation expérimentale (chapitre 8) de la pertinence de descripteurs spectro-harmoniques pour la caractérisation perceptive du timbre dans le cadre de sons complexes ;
- un algorithme évolutionnaire d'optimisation combinatoire multicritère adapté au problème de l'orchestration (chapitre 9) ; respectant l'équilibre entre intensification et diversification de la recherche et pouvant tenir compte des préférences de l'utilisateur.
- une généralisation des notions d'*instanciation partielle* et de *consistance locale* dans les problèmes de satisfaction de contraintes, à travers les concepts de *contraintes de design* et *contraintes de conflit* ;
- une métaheuristique de recherche locale pour la satisfaction de contraintes globales (chapitre 10), s'appuyant sur la distinction entre contraintes de *design* et contraintes de *conflit*.

- une validation expérimentale, sur des instances de petite taille, de l'efficacité des deux méthodes précédentes (chapitre 11) ;
- un prototype expérimental d'aide à l'orchestration (chapitres 12, 13 et 14), facilitant l'exploration du timbre orchestral via une représentation multi-points de vue des solutions et favorisant la découverte des préférences implicites du compositeur.

Première partie

Etats des arts

Chapitre 1

Informatique musicale

En quoi la musique a « besoin » de la technologie
La composition comme processus
Informatique musicale et paradigmes de programmation

Les amalgames trop fréquents entre *informatique musicale*, *musique assistée par ordinateur*, *musique électronique* et *musique contemporaine* sont source de bien des malentendus. Si l'ordinateur est devenu un outil incontournable pour les musiciens actuels, l'informatique musicale se limite pour la plupart des utilisateurs aux séquenceurs, instruments virtuels et éditeurs de partition. Les rapports riches et multiples entre ordinateur et création musicale contemporaine (et plus généralement entre informatique et musique) sont la plupart du temps dénigrés ou incompris du public. Du côté scientifique également, les idées reçues sont légion. Pour de nombreux aficionados des « sciences dures », l'informatique musicale fait référence à une poignée de chercheurs et musiciens contemporains « profitant » des technologies à leur disposition, et dont les objectifs théoriques ou esthétiques sont peu lisibles. En vérité, les préoccupations musicales ont toujours interrogé de près les ressources technologiques, et les rapports entre informatique et musique vont aujourd'hui bien au-delà d'une utilisation unilatérale et aveugle des outils. A y regarder de près, ils prennent davantage la forme d'un échange stimulant où sont régulièrement soulevées des problématiques de représentation des structures et de formalisation de la pensée.

1.1 Musique et technologie

Les rapports entre musique et technologie ne sont pas un particularisme du XX^{e} siècle. L'histoire de la musique occidentale est au contraire jalonnée de confrontations avec la technique, motivées tantôt par une demande musicale, tantôt par une irruption technologique. L'orgue (du grec *organon* : machine) est en ce sens emblématique de ces rapports complexes. Initialement perçu par le clergé comme une mécanique suspecte risquant de distraire les croyants de leur pratique religieuse, son acceptation dans les églises n'a pas été sans peine : en quoi la pratique musicale au sein du culte religieux, limitée alors au chant choral, avait-elle besoin de cette technologie infernale ?

C'est qu'en musique comme dans la plupart des expériences humaines, une technologie n'est jamais adoptée par « forçage », mais uniquement si elle répond à un besoin *ancien*. En ce sens, la notation musicale, le tempérament égal ou encore le piano sont des « technologies »,

et par là même, des révolutions : elles modifient en profondeur pratique et pensée musicales. Le codage, à travers la partition, de la musique sous forme symbolique a permis la manipulation du matériau musical à un niveau d'abstraction plus élevé que la seule pratique instrumentale. La notion occidentale de « composition » s'est alors enrichie de nouveaux procédés (inversions, symétries, miroirs, répétitions, imitations, esquisses...), révélés par la représentation *graphique* de la musique. De façon similaire, l'instauration progressive du tempérament égal a conduit à la possibilité (voire la nécessité) d'une musique modulante, de même que le piano a apporté les nuances de dynamiques dans le jeu au clavier.

Possibilité offerte d'abord, puis *évolution nécessaire*, un changement technologique ne survient que lorsque la pratique ou la pensée piétinent dans une impasse et ne peuvent s'en extraire avec leurs outils du moment. En même temps qu'elle comble un besoin, l'adoption d'une technologie engendre toujours un changement de paradigme. C'est sous ce double rapport de cause à effet que doit être considérée l'articulation contemporaine entre informatique et musique.

1.2 Le tournant de l'électronique

Durant la première moitié du XX^{e} siècle, le monde musical occidental est en crise : les tentatives dodécaphoniste et sérielle, cherchant le renouveau dans une nouvelle organisation des hauteurs, n'ont pu palier l'effondrement de la tonalité. Le changement viendra avec l'arrivée de l'électronique au milieu du XX^{e} siècle, mais pas de la manière qu'on croit. Certes, l'électronique introduit de nouvelles possibilités sonores, mais la grande nouveauté est ailleurs, dans les procédés qui lui sont propres et qui vont engendrer de nouvelles manières de composer. Les techniques de synthèse sonore d'une part, d'enregistrement sur bande d'autre part conduisent les compositeurs à penser le son comme un objet à part entière, totalement indépendant du jeu instrumental traditionnel. Louis et Bebe Barron (avec *Forbidden Planet*), Herbert Eimert, Karlheinz Stockhausen (avec *Gesang der Jünglinge*) ou encore John Cage (avec *Imaginary Landscapes*) seront les premiers à explorer ces potentialités. Dans le rapport d'intimité qui se crée avec le son « objectivé », les structures usuelles de l'écriture, thèmes, motifs, mélodies, contrepoints, cèdent la place aux bouclages, ralentissements, accélérations, superpositions, mixages, découpes, recombinaisons, stratifications... Cette pratique aura des répercussions immenses sur les musiques aussi bien électroacoustiques que mixtes ou même purement instrumentales. Par exemple, le fait de considérer l'orchestre comme un ensemble de solistes, en divisant à l'extrême les pupitres[1], est un mode d'écriture qui découle en partie de l'expérience des possibilités d'accumulations offertes par la pratique du multipiste. De même, les canons asynchrones de Steve Reich n'auraient sans doute pu être imaginés sans la possibilité de jouer simultanément un son enregistré à des vitesses et intervalles de temps différents.

1.3 Fonctions de l'Informatique musicale

Les rapports entre musique et informatique qui nous intéressent ici sont ceux qui introduisent des pratiques et des concepts nouveaux. Entre les années 50 et 70, l'ordinateur se

[1]Dans sa pièce *Atmosphères*, György Ligeti a recours à une division totale des pupitres de cordes ; apparente micropolyphonie, totalement asservie à la texture de l'orchestre. Il en va de même, dans un tout autre registre, pour la construction tonale et contrapuntique qui rythme *In C* de Terry Riley.

substitue au magnétophone dont les possibilités en termes de maîtrise du temps et de manipulation du son concret ont été rapidement épuisées par la musique électronique et électroacoustique. L'*Informatique musicale*, dont le fondement est la représentation numérique des données, se scinde alors rapidement en deux grands courants de recherche : l'analyse et la synthèse numérique des sons d'une part, et la formalisation des processus musicaux d'autre part. Le second nous intéresse particulièrement, tant d'un point de vue compositionnel qu'informatique, car la question de l'articulation entre algorithmique et données musicales symboliques y est centrale. Il donnera naissance, en 1956, au célèbre quator à cordes *Illiac Suite* de Hiller & Isaacson, première pièce entièrement « composée » par ordinateur à partir d'un ensemble de règles tirées d'un traité d'harmonie. Les recherches en *musique algorithmique* (ou *composition algorithmique*) se poursuivront par la suite en Europe avec Pierre Barbaud, Iannis Xenakis ou encore André Riotte.

Toutefois, l'idée de concevoir un programme capable de générer de la musique de manière automatique révèle rapidement ses limites : on ne pourra jamais écrire un ensemble de procédures suffisamment génériques pour anticiper les besoins de tous les compositeurs. La composition algorithmique va alors peu à peu céder le pas aux environnements de *composition assistée par ordinateur* (CAO), avec lesquels le compositeur interagit en tant que « programmeur » et définit ses propres procédures. Les outils de CAO sont basés sur une mise en correspondance de paradigmes de programmation avec la formalisation des structures et des processus de l'écriture musicale, et permettent d'établir une équivalence entre une « intention compositionnelle » et un calcul. A titre d'exemple, Agon & al. [AHA02] ont montré qu'on peut représenter un rythme musical sous forme d'un arbre (au sens informatique du terme), un rythme n'étant en effet rien d'autre que la division d'un intervalle de temps en sous-intervalles, chacun d'entre eux pouvant à son tour se subdiviser en durées plus petites. Comme un arbre, un rythme est donc une structure de données définissable de manière récursive. La manipulation des rythmes peut dès lors être appréhendée de manière nouvelle à l'aide d'opérations sur les arbres.

Parmi les principaux paradigmes de programmation utilisés dans la conception d'environnements de CAO, les plus récurents sont la programmation par objets [Pop91], la programmation déclarative ou programmation par contraintes [RAQ+01], et surtout la programmation fonctionnelle, dont les mécanismes d'abstraction et d'application permettent d'établir une équivalence entre programmes et données. Cet aspect est fondamental en musique, où le même matériau est conçu tantôt comme un objet figé, tantôt comme une structure dont on peut dériver de nouveaux objets[2]. Cette dualité entre données et processus a largement contribué au succès du langage Lisp [Ste90] dans le développement d'environnements de CAO.

Une étape majeure dans l'interaction avec le compositeur/programmeur sera alors franchie avec la *programmation visuelle*, dont le paradigme central postule l'équivalence entre un programme et sa représentation graphique. D'un point de vue formel, un « programme visuel » est un graphe orienté dont les nœuds sont des données ou des fonctions et les arrêtes des « connections » symbolisant un passage d'arguments entre les entrées/sorties (des fonctions) ou les champs (des objets). En pratique, l'évaluation d'un nœud déclenche l'évaluation récursive (éventuellement avec des règles de priorité) des nœuds « en amont ». La CAO basée sur la programmation visuelle permet donc au compositeur de tirer facilement parti de la puissance expressive des langages de programmation, et de concevoir (cons-voir) des programmes

[2]Il suffit pour s'en convaincre de considérer les multiples variations que l'on peut engendrer à partir d'un même motif, procédé omniprésent dans le répertoire classique (chez Beethoven, par exemple).

dans lesquels les structures usuelles de l'écriture musicale et l'ensemble des opérations s'y rapportant partagent le même environnement graphique. Les environnements de CAO les plus célèbres ayant recours à la programmation visuelle sont historiquement PatchWork [LD89], Max/MSP [Puc91], Elody [OFL97] et OpenMusic [Ago98], développé à l'IRCAM[3] par Gérard Assayag, Carlos Agon et Jean Bresson, et que nous détaillerons à la partie III.

Aujourd'hui, les deux courants de recherche en informatique musicale que nous évoquions en début de section se sont rejoints, et l'on dispose actuellement d'outils de CAO puissants dans lesquels le son comme matière sensible coexiste et interagit avec des structures formelles. Il devient donc possible pour le compositeur de relier l'exploration du sonore à l'organisation de données symboliques, et d'ainsi renouer avec une conception plus « classique » de l'écriture dont la musique électronique s'était éloignée (cf. paragraphe 1.2). Cette « convergence » est toute bienvenue pour nos travaux : car l'orchestration est en effet l'art de contrôler le timbre de l'orchestre à l'aide des paramètres symboliques de l'écriture musicale (voir chapitre 3). Aussi verrons-nous au cours de cette thèse qu'un des intérêts théoriques majeurs de l'orchestration assistée par ordinateur réside dans le recours à des représentations multi-échelle permettant la mise en relation de données hétérogènes. Se dessinera ainsi, au travers de notre recherche, un des enjeux contemporains de l'informatique musicale : l'intéraction signal/symbolique.

[3]Institut de recherche et de coordination acoustique/musique. `http://www/ircam.fr/`

Chapitre 2

Le timbre, objet et fonction

Tentatives de définition du timbre en musique
Le rôle essentiel du timbre dans l'écriture contemporaine
Approches scientifiques du timbre : psychoacoustique et description du signal

2.1 Emergence du timbre dans la musique du XX^{e} siècle

Depuis la fin du XIX^{e} siècle, l'emploi de plus en plus fréquent des percussions permet à la fois de retrouver ce côté « primitif » (au sens d'un rapport immédiat entre un geste et le son qui en résulte) de la production sonore, et de se familiariser avec de nouveaux timbres pour lesquels la notion de hauteur, élément jusqu'alors central dans la musique occidentale, n'est plus nécessairement pertinente. C'est l'occasion de prendre conscience que la mise en vibration d'un instrument produit d'abord un effet acoustique qui s'adresse à la perception, et qui avant le XX^{e} siècle dépasse toute organisation formelle du matériau musical.

Un second apport, dans les années 20, viendra des premiers instruments électroniques tels que le Thérémine ou l'Onde Martenot. Ces inventions, qui offrent la possibilité de produire un « continuum de hauteurs », ont une incidence profonde sur les compositeurs. Plus que renouveler l'effectif orchestral, elles introduisent des timbres « inouïs », jamais entendus jusqu'alors, profilant ainsi une nouvelle dimension dans l'imaginaire sonore.

Enfin, sous l'impulsion de certains compositeurs comme Strauss, Bartók ou Varèse[1], les instrumentistes eux-mêmes affichent peu à peu un goût pour l'exploration du vocabulaire acoustique de leurs instruments, et le langage musical va progressivement s'étoffer avec les pratiques nouvelles que constituent les modes de jeu[2]. Aujourd'hui, les pizzicati Bartók, jeux d'archets *battuto*, *sul ponticello*, *sul tasto* ou encore *col legno tratto*, le *slap*, le *growl*, le souffle aéolien ou encore les doigtés microtoniques font partie intégrante du discours musical contemporain.

Tandis qu'est redécouvert le « son réel » des instruments se dévoile donc progressivement un éventail de sonorités nouvelles, reléguant au second plan l'organisation traditionnelle des hauteurs. Les instruments produisent des sons et le rôle du compositeur consiste à les combiner, telle est l'image de la nouvelle pensée musicale qui s'instaure au début du XX^{e} siècle.

[1] En 1899, Richard Strauss demande au corniste de chanter dans son instrument dans *La vie d'un héros*. En 1936, Edgar Varèse prescrit au flûtiste de frapper les clés dans *Densités 21,5*.

[2] Cette pratique, déjà fréquente à l'ère baroque, devient quasi systématique à la seconde moitié du XX^{e} siècle

L'orchestre est peu à peu vu comme une gigantesque palette sonore avec laquelle composer : créer un « nouveau son », un « son d'orchestre », à partir de sons individuels. Le timbre, jusque là conçu comme une propriété intrinsèque des instruments, va progressivement s'en détacher pour devenir dans un premier temps l'élément d'un nouveau langage musical, jusqu'à être pensé comme le pivot central de l'écriture.

Varèse (et à sa suite Xénakis) est sans doute le compositeur le plus emblématique de cette nouvelle tendance. Il s'agit véritablement pour lui de créer de nouveaux timbres, de « composer le timbre » en puisant dans les richesses multiples du vocabulaire instrumental. « Varèse a en permanence inventé de nouvelles structures sonores, et leur individualité est si marquée que les termes musicaux usuels (accord, voix, doublure...) ne conviennent plus pour les décrire. [...] Varèse s'emploie à créer des couches de son, et ces couches occupent l'espace des hauteurs. Il y a des bandes de sons, des strates, et l'évolution de la musique est entièrement gouvernée par les confluences et les disjonctions de ces strates. » (Erickson, *Sound Structure in Music* [Eri75]).

Cette approche sémantique du timbre, dont les impacts sont colossaux dans la musique contemporaine, trouvera un écho dans la seconde moitié du XX^{e} siècle avec la « musique concrète instrumentale », chez des compositeurs comme Lachenmann ou Sciarrino. Tirant parti des possibilités bruitistes des instruments traditionnels, Lachenmann développe et organise un vocabulaire sonore extrêment riche en une « Klang Komposition ». Il en ressort une musique rapide, précise et rigoureuse, où les timbres fourmillent au service de la composition du son.

Parallèlement à la démarche sémantique, synthétique, ascendante de Varèse se développe à partir des années 50 une approche analytique, grammairienne, descendante, héritée des techniques récentes de compréhension et de représentation du phénomène sonore. La mise au point, au cours de la seconde guerre mondiale, du sonographe, a d'une part fourni un accès à la « réalité » du son à travers une représentation temps/fréquence/amplitude, d'autre part permis de considérer le son comme un mélange de composantes élémentaires : à la *synthèse* du timbre dans la composition musicale s'oppose sa *décomposition* par les nouveaux instruments de mesure.

Conscient de ce double mouvement, le compositeur François-Bernard Mâche a tenté d'en unifier les tendances en préfigurant ce qui allait devenir la « musique spectrale ». A l'initiative de Gérard Grisey et Tristan Murail, les compositeurs de cette école considèrent le timbre comme une structure préexistante à l'œuvre et dont on peut déduire une certaine organisation, en particulier de hauteurs. Démarche analytique certes, mais aussi synthétique, puisque l'écriture spectrale se caractérise par une orchestration très précise, souvent microtonale[3], devant atteindre un degré suffisant de fusion instrumentale[4] pour permettre l'émergence d'un timbre orchestral unifié et cohérent.

Ainsi le timbre, longtemps occulté par la pensée musicale du XIX^{e} siècle[5], revient-il au devant de la scène. Or, nous allons le voir, cela ne va pas sans poser de grandes difficultés,

[3]Depuis l'instauration du tempéramment égal, l'octave est divisé en douze demi-tons égaux. L'écriture microtonale divise le demi-ton en sous-intervalles : quarts de tons, huitièmes de ton, etc.

[4]Lorsque plusieurs instrumentistes jouent simultanément, on parle de fusion instrumentale si les différents timbres ne forment plus qu'un unique agrégat sonore.

[5]Il est significatif de remarquer que les instruments à timbre très caractéristique employés dans la musique du Moyen Age ou au début de l'ère baroque n'ont pas survécu à la standardisation de l'orchestre classique. De même, les instruments populaires y sont régulièrement absents avant le XX^{e} siècle. Question de tempérament, dirons certains. Il n'empêche, l'évolution de la facture instrumentale est allé dans le sens d'une standardisation — donc d'un appauvrissement — du timbre, sans laquelle la composition à travers le prisme classique des hauteurs n'eût pas été possible. En conséquence, le timbre est resté l'affaire du seul facteur d'instrument.

dont certaines font toujours débat aujourd'hui.

2.2 Paradoxes et polysémies

Qui tente de formuler une définition juste du timbre se heurte immanquablement à un certain nombre de difficultés, tant l'usage systématique du mot a fini, en multipliant les significations, par le vider de sa substance. Pour preuve la définition négative du timbre qu'on rencontre habituellement : « ce qui n'est ni hauteur, ni durée, ni intensité ». L'American Standard Association le définit en 1960 comme « ce par quoi on peut distinguer deux sons de même hauteur et même intensité ». Le timbre serait donc une propriété intrinsèque à l'instrument, interviendrait dans des tâches de différenciation (voire de catégorisation ou de reconnaissance), mais ne serait pas exprimable positivement pour autant.

Quoique regrettable, ce point de vue n'est pas blâmable. N'oublions pas qu'à l'origine, le timbre désignait la corde en boyau mise en double au-dessous de la caisse d'un tambour pour mieux le faire résonner, à l'instar du treillis métallique aujourd'hui fixé sous les caisses claires. Dès le départ, le timbre est donc quelque chose de fixé, d'indissociable de l'instrument, et qui lui donne une « couleur » particulière. Avant le XX^{e} siècle, il n'y a pas, rappelle Hugues Dufourt, d'autre science acoustique que celle des hauteurs ; aussi n'est-il guère étonnant qu'à l'époque, « le timbre [ne soit] pas [encore] une fonction du langage [musical], il n'est qu'une matérialité obligée du processus producteur de la vibration sonore. [...] En l'absence d'un rôle fonctionnel musical explicite [...] il est tout naturel que le concept de timbre tende à renvoyer [...] à la cause productrice du son : l'instrument. » (Claude Cadoz, *Timbre et causalité*, in [Bar85]). Il semble que ce soit ce lien primitif[6] à l'instrument qui soit à l'origine de la définition négative du timbre, témoin muet du lien entre le son et sa cause qui se dérobe de l'intérieur aux catégories abstraites et organisables de hauteur, de durée, d'intensité.

A l'inverse, on trouve dans l'*Harmonielehre* de Schoenberg [Sch11] les propos suivants : « Je ne peux accepter sans réserve la distinction entre couleur sonore et hauteur, telle qu'on l'exprime habituellement. Je pense que la hauteur devient perceptible grâce à la couleur sonore dont une des dimensions est la hauteur. La couleur sonore est donc la catégorie principale, la hauteur, une subdivision. La hauteur n'est pas autre chose que la couleur sonore mesurée selon une direction[7]. » L'opposition entre ce point de vue et la définition négative du timbre qui précède illustre la difficulté pour la pensée musicale au début du XX^{e} siècle de qualifier le timbre avec le vocabulaire dont elle dispose. Pour certains, le timbre est le « secret ineffable » de l'instrument alors que pour d'autres, c'est une totalité brute dont procède tout objet sonore. Tantôt reste hors-langage, une fois le sonore épuisé par le vocabulaire musical, tantôt objet supra-langagier, total, qui transcende les catégories de la pensée musicale traditionnelle, le timbre est simultanément en exclusion interne et externe par rapport au langage : il ne se donne aux mots que par passage à la limite.

Cette double exclusion se reflète dans le caractère polysémique du timbre. Voyons Schaeffer [Sch66] : « [Le timbre] peut être défini comme l'ensemble des caractères du son qui le

[6] Par « primitif » nous désignons ici un état antérieur au langage, où le timbre fonctionne comme le signe univoque de la cause qui le produit. A distinguer du « retour au réel du son » dont fait état la section 2.1, lequel n'a lieu que dans un second temps et qui constitue la première étape d'un « arrachage symbolique » visant à incorporer le timbre dans un langage (voir section 2.3).

[7] Cette position préfigure celle de Stockausen pour qui le timbre est toujours pensé comme une combinaison de hauteurs. Elle se rapproche en cela de la conception analytique du timbre chère à la musique spectrale (voir supra).

réfèrent à un instrument donné. [Cependant, on peut très bien parler du] timbre d'un son sans le rapporter clairement à un instrument déterminé, mais plutôt en le considérant comme une caractéristique propre de ce son. » Voilà donc une difficulté supplémentaire : chaque instrument possède son timbre, mais aussi chaque son qu'on en tire. Erickson [Eri75] explique cette contradiction en rappelant qu'avant le XX^{e} siècle l'écriture tire parti — en même temps qu'elle la conforte — de la notion de « consistance subjective » du timbre. Une ligne mélodique est en général confiée à un instrument ou un groupe d'instruments donnés, invariants pour l'intégralité de la mélodie. Les différences de timbre résultant des différences de hauteur et d'intensité sont alors identifiées comme nuances d'un timbre global qui fonctionne — Erickson use d'une analogie avec la modulation de fréquence — comme une « porteuse » de la ligne mélodique[8].

Bregman va encore plus loin : « De même qu'il existe le timbre d'une note isolée, on peut parler de timbre orchestral pour un groupe d'instruments. » (Albert Bregman, *Timbre, orchestration, dissonance et organisation auditive*, in [Bar85]) Il semble falloir s'y résoudre, le timbre renvoie à plusieurs significations. Il désigne soit l'ensemble des sonorités produites par un instrument donné (fonction de « consistance », ou « porteuse »), soit la couleur sonore d'un son instrumental isolé (fonction « dissociante »), soit encore le son résultant d'une mixture instrumentale (fonction « unifiante »). Mais nous allons voir que cette polysémie peut être dépassée en considérant, à l'instar des compositeurs contemporains depuis Varèse, le timbre comme un langage, et en tentant de comprendre le rôle qu'il remplit dans l'écriture musicale.

2.3 Le timbre comme langage

S'il est courant de rencontrer le terme de « langage musical », qu'entend-on exactement par là ? Sans entrer dans un débat de linguiste ou de sémiologue de la musique, nous nous contenterons de constater un certain nombre de points communs entre musique et langage. Tous deux se déroulent dans le temps, communiquent par le canal auditif et sont constitués d'entités élémentaires transcriptibles dont la notation renvoie à des phénomènes sonores identifiés et reproductibles. Ainsi hauteurs, durées, intensités, articulations, accents sont des éléments du langage musical. En est-il de même du timbre ? Oui, écrit Erickson [Eri75] : « De même que les unités élémentaires du discours sont les phonèmes, de même les timbres constituent les unités élémentaires du discours musical. »

Il est une condition nécessaire au langage : c'est la circulation, la mobilité de ces éléments, et cette fluidité requiert une certaine neutralité des phonèmes. Un son ne peut fonctionner comme phonème que déchargé de toute signification a priori. En d'autres termes, la fonction de « consistance » du timbre doit être brisée, le timbre ne pouvant faire langage tant qu'il renvoie de manière univoque à l'instrument qui le produit. « La standardisation [des instruments] a appauvri, dans un certain sens, la famille des timbres, mais elle leur a permis de communiquer entre eux. [...] Cette transformation va se refléter dans le timbre, qui ne sera plus considéré comme un principe de l'identification mais comme un principe de transition, sinon de confusion. [...] On convient de dire que l'orchestre moderne est né avec le XIX^{e} siècle ; il est né, en effet, de cet emploi mobile de l'instrument. [...] Le rôle [des instruments] va évoluer de l'identifiable au méconnaissable à cause de la brièveté des mélanges. [...] L'instrument est recherché pour ses possibilités de fusion, de neutralité, de perte d'une identité excessive qui,

[8] La rupture de ce principe est tardive. Elle n'advient qu'au début du XX^{e} siècle avec la nouvelle école de Vienne et la « Klangfarbenmelodie ».

évidemment, empêcherait les phénomènes de fusion. A partir de Schoenberg, en particulier, l'instrument sera de plus en plus considéré comme partie d'une texture [...] à saisir chaque fois dans un contexte différent. » (Pierre Boulez, *Le timbre et l'écriture, le timbre et le langage*, in [Bar85])

On retrouve là les deux fonctions du timbre mentionnées au paragraphe 2.2 : dissociation et unification. C'est par ce double mouvement permanent que le langage produit des signifiants et les agrège en de nouvelles significations. De même le « forçage » du tempérament aura-t-il permis, en séparant le ton de l'instrument qui le jouait, l'éclosion du langage tonal, de même le « retour au réel sonore » de l'instrument n'était-il que le premier pas vers un langage possible pour le timbre. Quant à s'entendre sur le choix, sans même parler d'une notation, d'un vocabulaire, c'est là un autre débat qui sera discuté en 7.1.1.

2.4 Fonctions du timbre : contraste et continuité

Depuis Berlioz, le timbre entre dans le langage musical, s'organisant en un ordre complexe qui s'érige au dessus du système instrumental. Le définir comme un ensemble de sonorités à disposition du compositeur ou comme le produit d'un mélange est une chose, tenter de cerner sa fonction dans le processus de composition en est une autre. Jean-Claude Risset [Ris86] ouvre la voie : « La notion de timbre implique la fusion, elle correspond à la qualité sonore d'un ensemble de composantes intégrées en une entité auditive et assignées à une même source sonore réelle ou virtuelle. » Le concept de timbre serait donc corrélatif de celui de fusion instrumentale, phénomène sonore perçu comme un tout, unique et cohérent. Cet agrégat sonore n'est pas nécessairement statique, il peut évoluer dans le temps. Le timbre comme langage transpose le concept de « consistance subjective » de l'instrument, vers la mixture. « La notion de cohérence du comportement ne se limite pas nécessairement à la source du son ; elle peut, en effet, s'appliquer à des groupes de sons, comme les accords ou les complexes de timbres, où plusieurs sources forment une image musicale unique. » (Stephen McAdmas et Kaija Saariaho, *Qualités et fonction du timbre musical*, in [Bar85]). C'est une des fonctions remplies, en orchestration, par les doublures.

Est-ce là tout ? Le timbre comme langage n'a-t-il d'autre fonction dans l'écriture que d'organiser la fusion instrumentale, que d'agencer les phonèmes pour produire de nouveaux objets sonores ? Ce serait oublier la fonction autogénérative du langage, lequel ne se contente pas d'agencer les signifiants, mais en produit également de nouveaux. Varèse, parlant de ses procédés de composition, utilise les termes de « masses sonores », « interactions », « projections », « pénétrations », « opacités », « raréfactions », « oxygénations »... tout un monde de mouvements, d'agitations dont la fusion instrumentale n'est qu'un état parmi d'autres. Aussitôt le timbre hybride formé, pourquoi ne pas le fragmenter en sous-ensembles, le faire s'entrechoquer avec une autre masse, ou encore l'incorporer dans une mixture plus vaste ? Voyons Erickson [Eri75] : « Varèse ne travaille pas seulement avec des timbres fusionnés, mais avec l'intervalle complet de la séparation à la fusion, et le mouvement entre ces états est fondamental dans son art. » De même que le langage se constitue sur le couplage des fonctions dissociante et unifiante, de même Erickson en vient à identifier les deux fonctions du timbre dans le processus compositionnel : contraste et continuité. « Tous les exemples que j'ai commentés peuvent être analysés en termes de contraste et continuité de timbre. Ces deux fonctions de timbre ne peuvent jamais être tout à fait dissociées, bien qu'il y ait des situations musicales dans lesquelles le timbre agit davantage comme porteuse (cf. 2.2) et d'autres pour lesquel l'intérêt

réside dans les agrégats de timbres. » Continuité et contraste, vision duale, temporelle de l'opposition boulézienne entre « articulation et fusion [...,] les deux pôles extrêmes de l'emploi du timbre dans le monde instrumental. » (Pierre Boulez, *Le timbre et l'écriture, le timbre et le langage*, in [Bar85])

2.5 Espaces de timbres

Que le timbre soit abordé de manière analytique ou synthétique, et quelles que soient les fonctions qu'il endosse dans la composition musicale, il n'en demeure pas moins du ressort de la perception : car l'oreille *fabrique* le timbre tout autant qu'elle le *perçoit*. « De façon évidente notre système auditif est capable d'une analyse multivariée extrêmement subtile des [éléments sonores], mais la perception s'en tient au pattern global qui en résulte. » (Schouten [Sch68]) La perception auditive est un phénomène éminemment complexe, multi-échelle, remplissant une double fonction, à la fois dissociante (ce qui nous permet d'isoler un source précise dans l'espace sonore) et unifiante[9] : « la chose la plus déroutante dans notre expérience sonore subjective est que nous entendons des sons multidimensionnels comme des objets perceptuels unifiés. » (Erickson, *Sound Structure in Music* [Eri75])

Les recherches récentes en perception se sont ainsi peu à peu détournées de la notion floue de timbre pour se concentrer sur un ensemble de paramètres précis et mesurables, en lien avec des attributs perceptifs. « A la question unique du timbre, se substitue alors une série de questions opératoires : comprendre comment l'oreille organise le sonore, comprendre dans quelles conditions le sonore est porteur de forme. » (Claude Cadoz, *Timbre et causalité*, in [Bar85]) McAdams partage ce point de vue en affirmant que « le timbre ne peut être un matériau propice pour l'écriture que si l'on parvient à identifier et à manipuler, parmi ses diverses composantes, des éléments porteurs de forme, identifiables comme tels dans le champ perceptif. [...] Nous parlons des "aspects du timbre", car la multidimensionnalité de celui-ci nous semble à présent bien établie, et nous estimons que ces multiples dimensions ne sont pas toutes aptes à porter la structure des formes. »

L'identification de ces dimensions privilégiées de l'écoute, leur mise en relation avec des attributs perceptifs, tels sont les objectifs des psychoacousticiens dans la construction des *espaces de timbres*.

Un espace de timbres est une représentation en deux ou trois dimensions d'un corpus de sons, chaque dimension étant reliée à une composante particulière du timbre. En général, sa construction a lieu en quatre étapes :

1. La première étape est la création du corpus. Dans le cas de sons instrumentaux, il arrive fréquemment qu'il soient reproduits par une méthode de synthèse afin de s'affranchir des conditions d'enregistrement et d'en éliminer les parasites.
2. Les sons sont ensuite présentés par paires à un ensemble de sujets qui doivent leur attribuer une mesure de similarité ou de dissemblance.
3. On utilise alors une méthode de *Multi-Dimensionnal Scaling* (MDS) permettant de déterminer les coordonnées de sons dans un espace euclidien à deux ou trois dimensions, de telle manière que les distances entre les points correspondent aux dissimilarités moyennes attribuées par les sujets.

[9] Chose amusante, on retrouve là les deux fonctions du timbre dans la musique du XX^{e} siècle.

4. Ultime étape, on tente alors de corréler les dimensions de l'espace de timbres à un ensemble d'attributs percepitfs.

Plomp (1970) fut le premier représenter des spectres synthétisés dans un espace euclidien tridimensionnel, sans toutefois interpréter la nature psychophysique de chaque dimension. Par la suite, les travaux de Wedin & Goude (1972) sur des sons d'instruments enregistrés, puis Miller & Carterette (1975) sur des sons de synthèse ont conduit à des modèles eux aussi tridimensionnels, dont les composantes sont essentiellement spectrales. A l'inverse, les études de Grey (1977), Wessel (1979) et Krumhansl (1989) ont révélé l'importance de paramètres temporels et spectro-temporels dans la perception du timbre. Leur travaux ont permis de construire un espace à trois dimensions comprenant le centroïde spectral (que Wessel reliera à la brillance), le flux spectral (variations du spectre dans le temps) et le temps d'attaque. Krimphoff et al. (1994) ont de leur côté proposé comme dimensions le temps d'attaque, le centre de gravité spectral (centroïde), et l'irrégularité spectrale (différences d'amplitude des partiels consécutifs). McAdams et al. [MWD$^+$95] ont plus tard confirmé ces résultats sur un plus grand nombre de sujets, d'expériences musicales variées. Pour cette étude, McAdams et al. ont travaillé sur des sons de synthèse et font observer que « le nombre de dimensions perçues est très inférieur au nombre de degrés de liberté disponibles dans la fabrication des stimuli. » Cette remarque illustre d'une part la fonction « unifiante » du timbre, d'autre part le fait que toutes les dimensions ne sont pas équivalentes dans la perception, et que seules les composantes prédominantes se retrouvent dans l'espace de timbres.

2.6 Timbre et description du signal audio

Depuis environ une dizaine d'années, les recherches menées sur la description et l'indexation automatique du signal audio ont connu en essor majeur, donnant naissance à la famille d'outils que sont les descripteurs. Un descripteur (ou descripteur audio) peut être défini comme une fonction permettant de calculer, directement à partir du signal, certaines caractéristiques acoustiques ou perceptives du son. C'est donc avant tout une méthode de calcul, même si l'usage commun fait qu'on parle fréquemment de descripteur pour désigner la valeur qui en résulte.

Les descripteurs du signal sont d'un grand intérêt dans un contexte de composition assitée par ordinateur, car ils délivrent une information condensée, de haut niveau d'abstraction par rapport à la représentation du signal comme suite d'échantillons. Pourvus d'une signification perceptive, ils constituent à n'en pas douter une piste sérieuse pour la formalisation du timbre et l'articulation signal/symbolique que nous évoquions à la fin du chapitre 1.

Ce n'est pas le lieu ici d'établir une liste exhaustive ou une classification de l'ensemble des descripteurs audio. Nous nous contenterons de les distinguer les selon les différents points de vue énumérés par Geoffroy Peeters [Pee04] :

- Les descripteurs peuvent être une *valeur calculée* directement sur une partie du signal (temps d'attaque) ou un *paramètre estimé* d'un modèle de signaux (ratio entre les partiels et impairs, inharmonicité).
- Ils peuvent se rapporter au signal dans son intégralité ou n'en décrire qu'une partie. Plus précisément, on distinguera les descripteurs valables — qu'ils en soit calculés sur la totalité ou non — pour l'ensemble du signal (descripteurs globaux, comme le temps d'attaque, le temps de décroissance, les modulations d'amplitude) de ceux ne concernant qu'une seule fenêtre temporelle (descripteurs instantanés, comme l'intensité perceptive).

- Les processus d'extraction des descripteurs diffèrent selon les cas. Certains descripteurs sont calculés directement sur la forme d'onde du signal (autocorrélation, zero-crossing rate), d'autres nécessitent une tranformation préalable (l'extraction des pics spectraux passe par exemple par une FFT). Ils peuvent aussi être obtenus par estimation (dans le cas d'un modèle de signaux) ou dépendre d'un modèle perceptif (filtrage de l'oreille interne, modélisation des bandes critiques...)

La description du son occupe certes une place importante dans nos travaux, mais non centrale. Nous nous contenterons donc d'introduire, au chapire 8, les descripteurs utilisés par notre système et d'ici-là retiendrons d'eux :

1. qu'ils peuvent être calculés directement à partir de la représentation du signal audio, donc extraits automatiquement aussi bien d'un son cible que d'une banque d'échantillons instrumentaux ;
2. que certains d'entre eux peuvent être corrélés à des attributs perceptifs : le centroïde spectral peut être relié à la brillance, le taux de passage par zéro (*zero-crossing rate*) au niveau de bruit dans le signal, le logarithme du temps d'attaque à l'aspect percussif du son. Encore une fois, nous renvoyons le lecteur désirant davantage de précisions aux travaux de Geoffroy Peeters [PMH00].

Conclusion

En résumé, nous dirons que le timbre est une donnée musicale dont la conception s'est peu à peu transformée depuis la fin du XIX^{e} siècle. Originellement considéré comme un élément étranger au système symbolique et propre au jeu instrumental, le timbre a fait son entrée dans l'écriture et en est devenu une variable centrale. Nous allons voir au chapitre suivant que l'art de l'orchestration consiste justement en la maîtrise du timbre dans la composition. Nous verrons en outre au long de ce manuscrit que la convergence entre l'approche psychoacoustique et le traitement du signal à travers un ensemble de mesures objectives (les descripteurs) permettra à la composition assitée par ordinateur d'aborder le problème du timbre.

Chapitre 3

Orchestration : un savoir empirique en mutation

L'orchestration comme art des mélanges
Savoir, transmission, limites
L'ordinateur comme outil de stimulation de l'imaginaire

On pourrait dire de l'orchestration qu'elle est l'ensemble des techniques d'écriture dans lesquelles le timbre agit comme paramètre principal. Walter Piston écrit dans son traité [Pis55] : « Lorsqu'on analyse une orchestration, il s'agit de découvrir comment l'orchestre est utilisé pour traduire la pensée musicale. [...] C'est un moyen d'étudier la manière dont les instruments se mélangent pour créer un équilibre de sonorités, l'unité et la variété des timbres, la clarté, la brillance, l'expressivité, ainsi que d'autres paramètres musicaux ». Plus simplement, on dira que l'orchestration est l'art de combiner des sonorités instrumentales pour créer des agrégats sonores et en contrôler l'évolution temporelle.

Il est intéressant de remarquer combien la position de l'orchestration par rapport à l'ensemble du savoir musical varie selon les époques et les auteurs. Koechlin, par exemple, semble la considérer comme une technique totalement asservie aux autres dimensions de l'écriture : « Il faut planifier avec soin, par rapport à l'œuvre dans son ensemble, les éléments orchestraux suivants : accents, cadences, progressions dynamiques, sommets, chevauchements. Les changements de timbre sont fonction du contexte musical. Le rythme d'entrée ou de retrait des timbres contribue, particulièrement dans une même phrase, aux effets de tension et de détente. Le degré de contraste timbral doit correspondre au degré de contraste formel : un changement de section exige davantage de contraste orchestral que l'apparition, dans une phrase, d'un nouveau motif. » (Alan Belkin, relatant Koechlin dans [Bel01]). Ce point de vue, en contraste avec celui de Piston, reflète une conception encore classique du timbre, selon laquelle les mélanges instrumentaux sont conditionnés par les catégories usuelles de la pensée musicale avant le XX^{e} siècle : « Masses, nuages et autres effets de composition de texture ne sont en aucun cas une invention récente. Debussy, Strauss, Wagner, Berlioz et leur contemporains ont intensément développé et exploité le potentiel des textures orchestrales. Mais dans la plupart des œuvres de Varèse, et dans certaines compositions de Schoenberg, Berg, Webern et Stravinsky la texture est au centre de l'écriture, et ne résulte pas d'un procédé contrapuntique, harmonique, mélodique ou rythmique. » Tant que la pensée musicale ne s'empare pas du timbre comme un objet en soi, l'orchestration reste une science secondaire. Mais chez Berlioz [Ber55], déjà on trouve les prémices de l'émancipation future : « L'emploi de ces divers éléments sonores et leur

application soit à colorer la mélodie, l'harmonie et le rythme, soit à produire des impressions *sui generis* (motivées ou non par une intention expressive), indépendantes de tout concours des trois autres puissances musicales, constitue l'art de l'instrumentation. »

3.1 Orchestration et instrumentation

C'est peut-être ici le lieu de préciser en quoi l'orchestration se distingue de l'instrumentation, avec laquelle elle est souvent confondue. L'instrumentation concerne l'étude des instruments, de leur étendue, de leurs registres et de la manière de les utiliser, de leurs potentialités sonores, de leur jouabilité, de leurs ressources de dynamique, d'articulation, de vitesse et de longueur de souffle, de leur place et de leur fonction dans l'orchestre, éventuellement de leur timbre, et plus rarement de leur capacité à traduire *individuellement* une idée musicale[1]. C'est un préliminaire incontournable pour l'étude de l'orchestration. Pour Piston [Pis55], « on n'insistera jamais assez sur l'importance de la connaissance instrumentale. Une écriture appropriée aux instruments est sans aucun doute le facteur déterminant dans la réussite d'une orchestration. A tel point qu'on peut affirmer en toute bonne foi que si les parties individuelles sont correctement écrites l'ensemble ne peut que bien sonner. »

En guise d'exemple, nous reportons ici quelques conseils d'instrumentation tirés du *Traité de l'Orchestration* de Koechlin [Koe43] :

A propos des cordes : « Comme le chœur vocal, la famille des cordes offre une belle homogénéité de timbres et peut exécuter une grande variété de traits, de la simple ligne monophonique à la plus riche des polyphonies. Pratiquement tout ce qui convient aux voix sonnera bien aux cordes. L'écriture pour cordes favorise les croisements, contrairement à l'écriture chorale. »

A propos des bois : « Les unissons produisent un effet de chœur. Un changement de registre provoque un changement radical de timbre. On peut presque aller jusqu'à considérer un instrument par registre. »

A propos des cuivres : « Les cuivres sont plus homogènes que les bois mais moins flexibles que les cordes. On peut leur attribuer, aussi efficacement, des rôles mélodiques, rythmiques, contrapuntiques ou harmoniques. Ils reproduisent, mieux que les bois, le chant choral. »

L'orchestration, quant à elle, renvoie aux multiples manières de mélanger les timbres individuels. « Nous ne nous sommes point imposé la tâche de donner une suite de méthodes des divers instruments ; mais bien d'étudier de quelle façon ils peuvent concourir à l'effet musical dans leur association. » [Ber55] C'est donc une science d'un niveau de complexité supérieur à l'instrumentation, et qui, d'une certaine façon, la contient : « Pour nous, comme domaine d'étude, l'orchestration succède à l'instrumentation, cette étape préliminaire où l'étudiant explore le fonctionnement des instruments et réalise ce qui est raisonnablement jouable par un exécutant professionnel. La conception trop répandue que l'orchestration n'est que l'attribution de timbres aux diverses lignes nous semble très inadéquate. » (Alan Belkin, citant Koechlin dans [Bel01]).

[1] On trouve parfois dans les traités que pour une situation donnée certains instruments conviennent mieux que d'autres, encore que ce jugement soit souvent très subjectif : « On apprend ce qui convient aux divers instruments, ce qui pour eux est praticable ou non, aisé ou difficile, sourd ou sonore ; on peut dire aussi, que tel ou tel instrument est plus propre que tel autre à rendre certains effets, à exprimer certains sentiments. » [Ber55]

3.2 Les traités célèbres

Les traités d'orchestration sont peu nombreux. Les plus célèbres sont ceux de Berlioz [Ber55] et de Koechlin [Koe43]. Parmi les ouvrages plus récents, on trouve ceux de Piston [Pis55], Rimski-Korsakov [RK64] ou encore Adler [Adl89]. Plusieurs raisons à ce petit nombre. Tout d'abord la relative jeunesse, dans le savoir musical, de leur propos. Ensuite et surtout, la difficulté de trouver un formalisme adéquat, comme il en va pour l'harmonie, le contrepoint ou l'organisation sérielle. Aussi l'empirisme est-il de mise dans l'enseignement de cet art ; « Orchestrer, c'est créer, ça ne s'enseigne pas », affirme Rimski-Korsakov. Ce point de vue était déjà celui de Berlioz [Ber55] : « J'ai déjà dit, je crois, qu'il me semblait impossible d'indiquer comment on peut trouver de beaux effets d'orchestre, et que cette faculté, développée sans doute par l'exercice et des observations raisonnées, était comme les facultés de la mélodie, de l'expression, et même de l'harmonie, au nombre des dons précieux que le musicien-poète, calculateur inspiré, doit avoir reçus de la nature. [...] Considéré sous son aspect poétique, cet art s'enseigne aussi peu que celui de trouver de beaux chants, de belles successions d'accords et des formes rythmiques originales et puissantes. »

Faute de pouvoir en exposer les règles, que peut donc faire l'auteur d'un traité d'orchestration, sinon produire une collection d'exemples appropriés tirés du répertoire, et discuter l'effet orchestral obtenu en fonction de l'effectif instrumental mobilisé ? « Ce n'est certainement pas par une étude systématique du timbre qu'on apprend l'instrumentation, mais en relevant ça et là des échantillons particulièrement réussis, choisis comme modèles. » (Pierre Boulez, *Le timbre et l'écriture, le timbre et le langage*, in [Bar85]).

Berlioz [Ber55] rapporte ainsi une série de « recettes », en repérant dans le répertoire[2] des « archétypes » orchestraux, orchestrations similaires qui produisent des effets similaires. En voici quelques exemples :

A propos des violoncelles : « Quand les violoncelles chantent, il est quelquefois excellent de les doubler à l'unisson par les altos. Le son des violoncelles acquiert alors beaucoup de rondeur et de pureté, sans cesser d'être prédominant. »

A propos des harpes : « Isolément ou par groupes de deux, trois ou quatre, il est singulier que ce soit le timbre des cors, des trombones, et en général des instruments de cuivre, qui se marie le mieux avec le leur. [...] Mais rien ne ressemble à la sonorité de ces notes mystérieuses unies à des accords de flûtes et de clarinettes jouant dans le médium. »

Sur l'utilisation du cor anglais : « Le mélange des sons graves du cor anglais avec les notes basses des clarinettes et des cors pendant un trémolo du contrebasses donne une sonorité spéciale autant que nouvelle, propre à colorer de ses reflets menaçants les idées musicales où dominent la crainte, l'anxiété. »

On trouve également dans les traités des principes d'orchestration plus généraux, qui n'ont pas valeur de règles auquelles se soumettre impérativement, mais que le jeune orchestrateur est invité à suivre. Ainsi chez Koechlin [Koe43] :

[2]Les compositeurs les plus fréquemment rencontrés dans le traité de Berlioz sont Gluck, Beethoven, Weber, et Berlioz lui-même.

Au sujet des dynamiques : « Evitez d'indiquer des dynamiques différentes aux différents instruments : cette façon de faire exige beaucoup d'expérience puisque les exécutants, dans l'ignorance des indications de leurs collègues et sans consignes particulières du chef d'orchestre, cherchent normalement à s'équilibrer entre eux. »

Au sujet des « plans sonores » : « Des plans sonores distincts exigent une individualisation forte, grâce à des contrastes de registre, de timbre et/ou de rythme. Si les plans dialoguent, il faut s'assurer d'une équivalence à la fois de force et de volume. Les couleurs, le registre et le rythme génèrent alors le contraste. »

Sur la manière d'écrire un tutti : « La recherche d'équilibre, entre les masses sonores, impose des contraintes acoustiques qui limitent les possibilités d'organisation d'un tutti : les cuivres et les percussions génèrent, inévitablement, le plus de puissance. Certaines combinaisons, par exemple, les bois dans le médium pendant que les cuivres jouent forts ne fonctionnent jamais. »

Aussi riches d'exemples, aussi exhaustifs que soient les traités, la question de leur précision et de leur portée mérite toutefois d'être posée. Quelle est leur utilité pour un compositeur d'aujourd'hui ? Quel est l'apport de cette impressionnante connaissance pour la musique contemporaine ? « On vous donne quelques recettes puisées dans les œuvres du répertoire, recettes très disparates, sans cohérence, auxquelles se mêle une certaine affectivité liée directement aux modèles proposés. » (Pierre Boulez, *Le timbre et l'écriture, le timbre et le langage*, in [Bar85]). Et sans même discuter l'éventuelle obsolescence du répertoire d'étude, le vocabulaire employé ne reflète-t-il pas déjà la vision esthétique d'une époque ? Nous ne résistons pas ici à citer encore une fois Berlioz [Ber55], dont le traité vaut certainement davantage aujourd'hui pour sa valeur littéraire que musicale :

> « Le trombone est, à mon sens, le véritable chef de cette race d'instruments à vent que j'ai qualifiés d'épiques. Il possède en effet au suprême degré la noblesse et la grandeur ; il a tous les accents graves ou forts de la haute poésie musicale, depuis l'accent religieux, imposant et calme, jusqu'aux clameurs forcenées de l'orgie. Il dépend du compositeur de le faire tour à tour chanter un chœur de prêtres, menacer, gémir sourdement, murmurer un glas funèbre, entonner un hymne de gloire, éclater en horribles cris, ou sonner sa redoutable fanfare pour le réveil des morts ou la mort des vivants. »

3.3 Vers un traité d'orchestration contemporain ?

Si l'enseignement de l'orchestration dans les conservatoires s'appuie aujourd'hui sur un répertoire d'étude récent, la musique contemporaine manque toutefois d'un traité tenant compte à la fois des nouvelles possibilités instrumentales et des avancées récentes dans la compréhension de la perception du timbre. Les compositeurs actuels ne se réfèrent plus à l'ouvrage de Berlioz [Ber55], considéré comme désuet. Quant à ceux de Koechlin [Koe43] et de Rimski-Korsakov [RK64], ils sont consultés pour des exemples isolés et sont rarement étudiés dans leur intégralité. L'art de mélanger les timbres reste donc très empirique et le jugement qualitatif des couleurs d'orchestre très subjectif. La seule vérité, aujourd'hui, est du côté de l'instrumentation. Ainsi le traité de Casella [Cas58] demeure-t-il toujours un ouvrage de référence, et

de nombreuses contributions récentes recensent un large éventail de techniques instrumentales nouvelles, tirées de l'exploration des modes de jeu contemporains.[3]

Cependant, le compositeur Yan Maresz rappelle que « parmi les compositeurs d'aujourd'hui, beaucoup sont résolument tournés vers les qualités expressives du son, vers le potentiel des sons complexes, bruités ou de source électronique comme matériau de base. L'orchestration dans le cadre de ces sonorités et techniques de composition devient donc une opération cruciale. » A la question de savoir dans quelle mesure on peut échafauder une théorie actuelle de la perception du timbre instrumental et du mélange des timbres, et comment cette théorie peut s'articuler avec la pensée créatrice contemporaine, Pierre Boulez répondait, il y a vingt ans, de manière pessimiste : « Selon les buts poursuivis, il existe deux manières de considérer le timbre. D'une part, une façon objective, scientifique, hors du langage, sans critère esthétique à proprement parler [...] On décrit [...] nombre de caractères acoustiques ; quant à la qualité d'intégration du son et du timbre dans la composition, elle est absente de ces mesures. Même lorsqu'on se préoccupe de la perception du phénomène sonore et de la qualité de cette perception, il s'agit d'une perception isolée, déliée d'un contexte quelconque. Dans cette approche, la valeur proprement artistique du timbre me semble oubliée. D'un autre côté s'impose la façon subjective, artistique, d'aborder le timbre comme composante du langage avec les critères esthétiques et formels qui s'y rapportent, d'où la difficulté inverse, voire l'impossibilité de relier le sentiment instinctif du qualitatif à une estimation plus raisonnée du quantitatif. » (Pierre Boulez, *Le timbre et l'écriture, le timbre et le langage*, in [Bar85])

3.4 Orchestrer avec l'ordinateur

Ne peut-on désormais dépasser ce dilemme ? Aujourd'hui, l'ordinateur offre la possibilité de manipuler un objet sonore dans un processus de composition, et simultanément d'en étudier les propriétés acoustiques au travers d'outils d'analyse. Les deux visions antagonistes du timbre que Pierre Boulez dénonçait ne peuvent-elles donc se rejoindre dans cette simultanéité ? Les fonctions d'analyse acoustique et de traitement des objets musicaux ne peuvent-elles pas interagir dans un environnement commun, subordonné à la pensée créatrice du compositeur ? S'il faut actuellement renoncer à un traité d'orchestration contemporain, ne peut-on alors concevoir un système informatique assistant le compositeur dans l'exploration et la manipulation du timbre orchestral, dans lequel l'analyse et la fabrication du sonore seraient intimement articulés ?

N'oublions pas qu'avant le XX^{e} siècle, les bons orchestrateurs sont les compositeurs ayant acquis une solide expérience de chef d'orchestre. Ce sont donc ceux qui ont pu, régulièrement, à travers leurs propres œuvres ou celle du répertoire, expérimenter les mélanges orchestraux et la qualité de leurs effets. Aujourd'hui, l'étendue des possibilités sonores des instruments est si vaste qu'on ne peut décemment attendre des compositeurs contemporains — y compris ceux passés par la direction d'orchestre — un savoir instrumental aussi complet qu'à l'époque de leurs aînés.

Comme souvent en CAO, l'ordinateur peut être conçu comme une machine à « faire des propositions ». En un sens, la CAO se rapproche sous certains aspects de la recherche opérationnelle, soit, de manière générale, l'ensemble des techniques d'aide à la décision face à

[3]Parmi les plus pertinents et les plus actuels : *New sounds for woodwinds* de Bartoluzzi, *Flûtes aux présent* de Pierre-Yves Artaud, *Saxologie* de Daniel Kientzy, *Les Modes de jeu de la contrebasse* de Jean-Pierre Robert, *The Techniques of Oboe Playing* de Peter Veale, *Le Tuba contemporain* de Gérard Buquet.

des problèmes complexes. L'idée que nous défendons dans cette thèse est alors la suivante : la puissance de calcul de l'ordinateur doit permettre d'explorer de manière efficace l'espace des alliages de timbre et d'évaluer en un temps raisonnable les combinaisons instrumentales a priori les plus pertinentes pour le compositeur. Un système d'aide à l'orchestration agirait donc avant tout comme un outil de « stimulation de l'imaginaire », en proposant au compositeur des solutions auxquelles son savoir et son expérience personnelle ne l'aurait pas nécessairement mené. Il ne s'agit évidemment en aucun cas de subsituer aux techniques usuelles d'orchestration la formalisation d'un ensemble de règles (et d'abord lesquelles ?), mais plutôt d'utiliser l'ordinateur comme un « explorateur » qui, de manière interactive et guidée, « découvrirait » de nouveaux territoires dans l'écriture du timbre orchestral. Une nouvelle manière d'orchestrer certes, mais qui n'invalide pas l'ancienne, et la pourvoie au contraire d'une dimension nouvelle.

Chapitre 4

Aide à l'orchestration

L'orchestration encore peu abordée par la CAO
La demande des compositeurs ne date pas d'aujourd'hui
Revue et critiques des outils actuels d'aide à l'orchestration

Comme il a été dit au chapitre 1, l'informatique musicale s'est rapidement scindée, après sa naissance dans les année 50, en deux courants de recherche longtemps restés imperméables l'un à l'autre, en partie en raison de l'hétérogénéité de leurs objets d'études. D'une part l'analyse et la synthèse numérique des sons a largement contribué, à travers le développement continu de méthodes spectrales et spectro-temporelles, à la compréhension du phénomène sonore et a permis la fabrication de sons jusqu'alors inouïs. Le logiciel AudioSculpt [BR05] et le langage CSound [Bou00] découlent directement de ces travaux. D'autre part, l'approche algorithmique s'est concentrée sur les structures symboliques de l'écriture musicale et a donné naissance aux environnements de Composition assistée par ordinateur (CAO), tels PatchWork [LD89] et OpenMusic [Ago98]. Ces outils, pour lesquels la notion de calcul est centrale, permettent de générer et de transformer des structures musicales à travers ce qu'on convient aujourd'hui d'appeler un « processus compositionnel ».

Or l'orchestration se situe précisément à l'intersection de ces deux axes de recherche ; si elle prétend créer des timbres, c'est encore par le biais de l'écriture. Elle est une station de correspondance entre le symbolique et le sonore. Aussi n'est-elle appréhendable par l'informatique musicale qu'à la condition d'une convergence de l'approche « signal » et de l'approche algorithmique. Le programme Max/MSP [Puc91] fut sans doute un des premiers outils à permettre cette rencontre, mais sa conception basée sur le contrôle en temps réel le destine essentiellement à la performance.

Aujourd'hui, la cohabitation du signal et du symbolique au sein d'un même environnement ne pose plus de difficulté majeure, mais le problème de l'orchestration assistée par ordinateur n'est pas résolu pour autant.

4.1 Convergence et maturité des savoirs

L'idée d'un outil d'orchestration assistée par ordinateur n'est pas nouvelle. Lorsqu'au début des années 70 les compositeurs familiers des possibilités combinatoires et algorithmiques que leur offraient les environnements de CAO se sont tournés vers les propriétés spectrales du son, germa aussitôt l'idée d'un programme d'aide à l'orchestration. Cependant, au delà de

la difficulté même que représente la formalisation du passage de la notation musicale à la réalisation acoustique, un tel souhait n'était réalisable qu'une fois différents champs du savoir parvenus à maturité.

En premier lieu, rappelons qu'il n'est pas d'orchestration possible sans la connaissance préalable des instruments et de leurs capacités sonores. La représentation en machine des étendues, des registres et des modes de jeu instrumentaux, ainsi que des ressources en dynamique propre à chaque hauteur ne pose pas beaucoup de problèmes, car ce sont là essentiellement des données symboliques. Mais peut-on concevoir un formalisme qui puisse capturer les propriétés acoustiques des instruments ?

On ne peut répondre à cette question sans en passer au préalable par la compréhension du timbre instrumental. Les études psychoacoustiques récentes y ont largement contribué, en révélant l'aspect multidimensionnel de la perception du timbre. Elles ont notamment abouti à l'identification de directions spécifiques de l'écoute et à la construction d'« espaces de timbres » expliquant une partie des phénomènes de similarité et de dissemblance dans la perception des sons. Parallèlement, les avancées récentes en traitement du signal ont favorisé la mise en correspondance de certains attributs perceptifs avec un ensemble de mesures calculées directement sur la représentation des signaux audio. Ce n'est donc que très récemment qu'un lien a pu être établi entre l'enregistrement d'un son et la caractérisation objective, calculée, de certains aspects de son timbre.

Dès lors, la question de la connaissance instrumentale passe par celle de l'accumulation d'échantillons de sons représentatifs des possibilités sonores des instruments. Or l'enregistrement, mais aussi le stockage et la mise à disposition de ces gigantesques bases de données sonores n'est possible que depuis une dizaine d'années.

Enfin, n'oublions pas ce qui constitue l'essentiel du travail présenté dans cette thèse : le grand nombre d'instruments et la diversité des timbres offrent une palette quasi infinie de couleurs orchestrales. La recherche d'un mélange instrumental par l'ordinateur soulève donc des problèmes combinatoires colossaux que seules les méthodes récentes issues de la recherche opérationnelle peuvent appréhender. On comprendra alors aisément pourquoi l'orchestration assistée par ordinateur n'en est aujourd'hui qu'à ses débuts.

4.2 Discours de compositeurs

Conscient des problématiques actuelles des compositeurs, et fort de son rôle de médiateur entre recherche et création musicales, l'IRCAM a encouragé en 2003 la création d'un groupe de réflexion rassemblant chercheurs et compositeurs autour du problème de l'orchestration assistée par ordinateur. De ces multiples tables rondes s'est dégagé un ensemble de suggestions et de directions de recherche, que nous rapportons sommairement ici.

> « Pour de nombreux compositeurs, il apparaît urgent de disposer d'un outil d'exploration et de contrôle des alliages de timbres imaginables qui tirerait parti de la puissance informatique pour en tester les combinaisons instrumentales et en évaluer la qualité. Cette idée n'est pas nouvelle : elle revient de manière récurrente depuis les débuts de la CAO et représente peut-être le dernier bastion de la technique de composition "à l'ancienne" qui n'ait bénéficié de l'informatique à des fins d'analyse et de formalisation. »
>
> « L'idée de base serait d'avoir un outil d'analyse sonore et d'orchestration expérimentale qui puiserait dans une banque de données d'informations instrumentales

(spectres, formants, etc...) pour proposer un arrangement instrumental qui se rapprocherait le plus possible de l'analyse de la source d'origine. »

« C'est précisément un projet qui nécessiterait de bons programmes d'analyse, et de bonnes connexions entre lecture d'échantillons, analyse, représentation et synthèse. »

« La prise en compte des instruments comme générateurs de sons complexes pouvant être mélangés pour obtenir une cible acoustique ou symbolique est-elle une problématique "explorable" par le calcul ? »

« Pour un problème d'orchestration donné, y aurait-il des solutions pertinentes que le compositeur n'entend pas spontanément et qui peuvent donc faire l'objet d'une proposition par le système, ce dernier jouant alors le rôle d'un stimulateur de son imaginaire ? »

« Les compositeurs de musique symbolique (écrite) restent encore au stade d'un travail très intuitif sur ces sons bruités, ou utilisent des modèles assez simplifiés (le spectre harmonique). D'où l'idée de développer avec l'aide de scientifiques un outil d'aide à l'orchestration, qui pourrait englober ces catégories de sons non harmoniques. En d'autres termes, il s'agit d'élaborer des notions grammaticales solides, c'est-à-dire des lois de consonance psychoacoustique (au sens large), et des outils informatiques permettant d'assembler les sons et d'assister le compositeur dans ses recherches. »

« La portée pédagogique d'un tel outil est aussi une composante importante des motivations à la base de ce projet car la composante "analyse" du programme doit permettre de comprendre les règles et usages de l'orchestration telles que nous les trouvons décrites dans l'histoire et d'expliciter de manière plus scientifique les choix empiriques de nos aînés. »

Analyse de sons, banque de données d'informations instrumentales, représentations et synthèse, cible acoustique, sons bruités ou inharmoniques, consonance psychoacoustique, exploration des alliages de timbres, stimulation de l'imaginaire... : sans entrer dans des détails de conception, nous voyons là se profiler un ensemble de problématiques dont une vie entière de recherche ne viendrait pas à bout. De fait, l'outil d'aide à l'orchestration dont cette thèse fait l'objet est loin de répondre à toutes ces exigences, dont certaines sont encore utopiques aujourd'hui. Nous verrons en outre que, par sa conception même et les choix faits en termes de caractérisation du timbre instrumental, notre outil renvoie implicitement à une certaine vision de l'orchestration, et penche de fait en faveur d'une esthétique musicale particulière. Nous discuterons ces aspects en gardant à l'esprit qu'un tel outil, motivé par des enjeux aussi bien artistiques que scientifiques, ne peut être conçu qu'au travers d'un compromis entre liberté et contrôle.

4.3 Outils actuels d'aide à l'orchestration

Les système d'aide à l'orchestration sont encore peu nombreux, essentiellement pour les raison exposées en 4.1. Nous présentons ici les travaux existant aujourd'hui à notre connaissance, en exposons les concepts sous-jacents et en exposons les limites.

Rose et Hetrick [RH05] ont proposé un outil à la fois pédagogique et créatif, permettant d'une part d'analyser une orchestration existante, d'autre part de créer des orchestrations

originales dont la sonorité s'apparente à un son cible donné en entrée du système. La connaissance des possibilités instrumentales est obtenue par analyse d'échantillons instrumentaux, en calculant des spectres harmoniques moyens sur la partie entretenue des sons. Le spectre du son cible est calculé par une analyse similaire. L'algorithme d'orchestration de Rose et Hetrick repose sur une décomposition en valeurs singulières du spectre de la cible, celle-ci s'exprimant alors comme une somme pondérée des spectres de la base de données.

Cette méthode présente deux avantages significatifs. Elle garantit que la décomposition trouvée est la plus proche du spectre de la cible au sens de la norme $\mathbb{L}^2$. En outre, le coût de calcul est dérisoire. En revanche, les spectres sont calculés avec des FFT[1] de 4096 points, et la comparaison de spectre sur quelques miliers de points est, sur le plan perceptif, sujette à caution. En outre, cette approche ne permet pas de prendre en compte les contraintes liées à l'effectif instrumental : il se peut en effet que l'outil de Rose et Hetrick propose en combinaison nécessitant, mettons, trois violoncelles, alors que l'orchestre n'en compte qu'un seul.

SPORCH (SPectral ORCHestration) est un système développé en Lisp par David Psenicka [Pse03] dans lequel la recherche s'effectue sur les instruments eux-mêmes, non sur les sons. SPORCH utilise un algorithme itératif de « matching pursuit » pour trouver une combinaison d'instruments s'approchant au mieux d'une cible sonore, donnée comme précédemment sous forme d'un son enregistré. Chaque instrument de la base est associé à un ensemble de hauteurs et de dynamiques, auxquelles s'ajoute une collection de partiels les plus importants d'un point de vue perceptif. L'algorithme de SPORCH extrait les partiels principaux de la cible et cherche dans la base les sons les plus pertinents. Le critère de sélection est une distance euclidienne entre les partiels suffisamment proches en fréquence. Un partiel présent dans un son candidat mais absent dans la cible augmente donc la valeur de distance.

L'algorithme de SPORCH est itératif. Après avoir trouvé le candidat qui minimise la distance au son à reproduire, son spectre est soustrait à celui de la cible et l'algorithme reprend la recherche avec pour cible l'ensemble de partiels résultant de la soustraction. Evidemment, l'instrument sélectionné à l'étape précédente est exclu du domaine de recherche. Cette méthode permet de trouver rapidement des solutions exécutables par l'orchestre (les contraintes liées à l'effectif instrumental sont toujours vérifiées), mais rien ne garantit l'optimalité de la solution obtenue. En outre, cette technique a tendance à privilégier les configurations de faible cardinalité.

Un troisième système est dû à Thomas Hummel [Hum05]. Un algorithme itératif est également utilisé, mais la mesure de distance entre les sons candidats et la cible se base sur l'enveloppe spectrale plutôt que sur les partiels. Par conséquent, les hauteurs perçues dans la mixture proposée peuvent être très différentes du contenu harmonique de la cible. Hummel conseille d'ailleurs d'utiliser son système pour les sons dépourvus de hauteur, comme les voyelles chuchotées.

Avant d'aller plus loin, quelques remarques s'imposent. Ces trois systèmes présentent de nombreux points communs. Voyons lesquels, et discutons les implications des choix faits par leurs auteurs :

[1] L'algorithme de FFT (*Fast Fourrier Transform*, en français Transformée de Fourrier rapide) est une méthode particulièrement efficace pour le calcul de la Transformée de Fourrier à court terme (TFCT, ou STFT — *Short Time Fourrier Transform* — en anglais), lorsque la taille de la transformée est une puissance de 2.

- Il est à chaque fois question d'approcher une « cible sonore », et celle-ci est toujours donnée au système comme un son enregistré. C'est là se restreindre à un cas très particulier de l'orchestration, car la plupart du temps, le compositeur ne dispose pas au préalable de ce son. L'orchestration, en général, a pour but de *produire*, non de *re-produire*, un timbre. « Savoir comment obtenir tel effet précis ne suffit pas à la création. Cela suppose que l'effet en question soit déjà conçu. Or, le processus de création est aussi celui de la genèse des idées. Un outil de création doit permettre cette genèse. » (Claude Cadoz, *Timbre et causalité*, in [Bar85]) La démarche sémantique chère à Varèse est donc exclue de l'approche adoptée par les auteurs.
- Pour chacun de ces systèmes, la connaissance instrumentale se résume à un ensemble de sons instrumentaux. Sans même parler de leur accessibilité, la généricité de ces bases de données mérite d'être questionnée. Quelle est l'information apportée par un Si bémol médium de tombone ténor joué ordinario en dynamique mezzo-forte ? Est-elle indépendante des conditions d'enregistrement de ce son ? L'acoustique du lieu d'enregistrement, l'instrumentiste, la qualité et le type de prise de son constituent, tout comme les indications mentionnées sur la partition, des paramètres majeurs du timbre. Ne court-on pas alors un risque de « surapprentissage » ? Quelle est la part variationnelle liée aux conditions d'enregistrement dans ce que le système « apprend » de l'échantillon ? Les orchestrations obtenues par ces outils produiront-elles les mêmes effets de timbre dans des conditions de jeu différentes ?
- Tous les auteurs utilisent un critère spectral unique pour l'évaluation des orchestrations proposées. Certes, il fut longtemps considéré que l'information spectrale était prédominante dans la caractérisation du timbre instrumental, mais on sait aujourd'hui que les phénomènes temporels et spectro-temporels ont une importance considérable. Par ailleurs, bien que l'information spectrale soit dans ces sytèmes représentée par des données multidimensionnelles, l'évaluation des propositions se résume toujours à un critère unique, sans doute pertinent pour une certaine classe de problèmes, mais certainement pas pour tous.
- Les trois systèmes formulent l'hypothèse implicite de l'additivité des timbres instrumentaux. Or la simple considération de l'effet de phase suffit à la mettre en défaut. Les compositeurs connaissent bien ce phénomène. C'est lui qui explique en grande partie pourquoi un ensemble de quatorze violons à l'unisson ne sonne pas quatorze fois plus fort qu'un violon solo.[2] Il est donc permis de douter des capacités prédictives d'un modèle additif, aussi avantageux soit-il d'un point de vue mathématique.
- Les effets de salle (notamment la réverbération) ne sont pris en compte dans aucun de ces systèmes. Or, la réverbération naturelle des lieux de concert a un impact énorme sur les phénomènes de fusion acoustique. Les compositeurs le savent bien, et comptent les effets de salle parmi leurs alliés pour la réussite de leurs orchestrations.[3]
- Enfin les trois auteurs, recourant à des algorithmes itératifs ou de décomposition,

[2] Koechlin [Koe43] fait à ce sujet la distinction entre le « volume » (à entendre au sens spatial) du son, c'est-à-dire son épaisseur, son étalement dans l'espace, et la « force », c'est-à-dire sa puissance. Ainsi, peu importe le niveau dynamique, un cor sonnera toujours plus ample, plus épais, qu'un violon. La règle générale concernant les doublures à l'unisson est qu'elle ajoutent plus de volume que de force.

[3] Berlioz [Ber55] est même allé jusqu'à dire que sans réverbération il n'y avait pas de musique : « Voilà pourquoi la musique en plein air n'existe pas. Le plus terrible orchestre placé au milieu d'un vaste jardin ouvert de toutes parts, comme celui des Tuileries, ne produira aucun effet. »

contournent systématiquement le problème combinatoire inhérent à l'orchestration. Imaginons un petit ensemble de six instruments, chacun d'entre eux étant représenté dans une base de données par seulement[4] une centaine d'échantillons sonores. L'ensemble des combinaisons possibles atteint déjà mille milliards. Du point de vue combinatoire, l'orchestre symphonique donne une vague idée de l'infini. Or il est clair que les méthodes itératives employées s'apparentent à une variété d'algorithmes gloutons, dont l'efficience dans les problèmes combinatoires n'est prouvée que pour un nombre très réduit de cas. Afin de ménager le temps de calcul, les auteurs ont donc recours à une forme d'heuristique qui ne garantit en rien l'optimalité du résultat obtenu.

Nous rassemblons au chapitre 7 les idées maîtresses de notre outil d'aide à l'orchestraion, et décrivons plus profondément ce dernier à la partie III de ce document. Nous verrons que les limites des systèmes existants ne peuvent être simultanément dépassées. Aussi retomberons nous parfois, en connaissance de cause, dans les écueils que nous venons de dénoncer. Cela dit, la conception et le développement de notre outil ont toujours été gouvernés par un souci de généricité et d'extensibilité future. Il faut donc voir dans notre travail une évolution vers un environnement de création puissant et complet, répondant aux exigences actuelles des compositeurs.

[4]Pour donner un ordre de grandeur, la base de données de l'IRCAM *Studio Online* [BBHL99] compte en moyenne un millier de sons par instrument. Quant à la célèbre *Vienna Symphonic Library* [VSL], elle dénombre pas moins de 250000 échantillons.

Chapitre 5

Optimisation combinatoire multicritère

Problèmes $\mathcal{NP}$
Introduction à l'optimisation multicritère
Panorama des algorithmes génétiques
Fonctions « scalarisantes » de Tchebycheff
Population size Adaptive Density Estimation

L'outil d'aide à l'orchestration que nous introduisons au chapitre 7 cherche à découvrir des mélanges de timbres instrumentaux, plus précisément des combinaisons de sons dans de grandes bases d'échantillons sonores, les plus proches possible d'un timbre cible. Or, nous le verrons, le caractère multidimensionnel de la perception du timbre d'une part (voir section 2.5), et la difficulté de mesurer l'importance relative de chacune de ces dimensions d'autre part (voir paragraphe 7.2.4), imposent une approche multicritère. Ce chapitre est donc consacré à la théorie des méthodes multicritères et aux algorithmes génétiques (AGs), couramment employés dans des tâches d'optimisation combinatoire (notre algorithme d'orchestration est d'ailleurs lui-même un AG). Nous détaillons en particulier le principe d'agrégation des critères par fonctions « scalarisantes » de Tchebycheff ainsi qu'une procédure efficace pour le maintien de la diversité des solutions, méthodes dont s'inspire notre propre algorithme.

5.1 Recherche opérationnelle, complexité

La recherche opérationnelle désigne l'ensemble des méthodes d'aide à la décision dans le cadre de problèmes complexes. Elle est couramment employée lorsque le décideur est confronté à un problème combinatoire, comme c'est le cas de l'orchestration assistée par ordinateur. De nombreux problèmes abordés par la recherche opérationnelle, tels que l'ordonnancement de taches, la gestion de projets, la logistique (tournées de véhicule, conditionnement), la planification, la gestion d'emplois du temps, les politiques d'investissement, les problèmes concurrentiels (stratégies militaire ou d'entreprise), les choix d'architecture (informatique ou organisationnelle), sont réputés pour leur complexité et classés dans la catégorie des problèmes $\mathcal{NP}$.

En théorie de la complexité, les problèmes $\mathcal{NP}$ peuvent être résolus sur une machine de Turing non déterminste par un algorithme polynomial. En d'autres termes, le temps de calcul est une fonction polynomiale de la taille du problème. A l'opposé d'une machine de Turing

déterministe, la machine non-déterministe peut effectuer plusieurs actions différentes pour un même état. En pratique, on ne sait pas construire une telle machine (car elle se décompose en autant de machines déterministes que d'actions possibles pour un état donné), pas plus qu'on ne sait simuler le comportement d'une machine non-déterministe à l'aide d'une machine déterministe.

Les problèmes $\mathcal{NP}$ admettent un algorithme polynomial capable de tester la validité ou la qualité des solutions, mais non de les trouver. On ne connaît pas, pour ces problème, d'autre méthode de résolution que l'énumération systématique de l'ensemble des solutions, et leur évaluation à l'aide d'un algorithme polynomial. En d'autres termes, le temps de calcul sur une machine déterministe est exponentiel à la taille du problème.

Dans la suite de cette section nous énumérons brièvement les approches les plus usitées pour les problèmes d'optimisation combinatoire multicritère. Ces derniers sont caractérisés par des variables de contrôle (ou de décision) discrètes, ne pouvant prendre qu'un nombre fini de valeurs. Nous verrons au pargraphe 7.4 que l'orchestration assistée par ordinateur appartient à cette catégorie. Pour une étude générale et appronfondie des méthodes multicritères, nous renvoyons le lecteur aux ouvrages de Miettinen [Mie99], Collette & Siarry [YP02], ou encore Ehrgott [Ehr05].

5.2 Théorie des approches multicritères

Les méthodes multicritères sont préconisées lorqu'il s'agit d'atteindre conjointement des objectifs souvent inconciliables. Dans le cas de l'orchestration, il s'agira de minimiser simultanément un ensemble de distances à un timbre cible selon plusieurs critères perceptifs. Un problème de minimisation multicritère est simplement défini par la donnée d'un espace de recherche (appelé parfois également *espace des décisions*) E et un ensemble de fonctions $f = (f_1, \ldots, f_N)$ à minimiser sur E :

$$\begin{cases} \min \ f(x) = (f_1(x), \ldots, f_N(x)) \\ \text{s.t. } x \in E \end{cases} \tag{5.1}$$

Définition 5.1. *Soit E un espace de décisions et $\{f_n\}_n$ les fonctions d'un problème multicritère sur E, alors l'espace :*

$$C = \{(f_1(x), \ldots, f_N(x)) \mid x \in E\} \tag{5.2}$$

est communément appelé ***espace des critères****.*

Il arrive que E ne soit pas défini *explicitement* (par des bornes d'intervalles par exemple) mais *implicitement*. C'est le cas notamment d'une optimisation sous contraintes, dans lequel les solutions de l'équation 5.1 doivent également satisfaire un ensemble de conditions. Nous verrons au paragraphe 7.5 que bien souvent l'espace de recherche dans un problème d'orchestration est défini à la fois explicitement (par la donnée des instruments de l'orchestre) et implicitement (par un ensemble de contraintes).

En général, la solution idéale x_{ideal} du problème 5.1, qui minimise *simultanément tous les critères*, n'existe pas :

$$\nexists x_{ideal} \in E, \ \forall n \in \{1, \ldots, N\}, \ f_n(x_{ideal}) = \min_E \ f_n$$

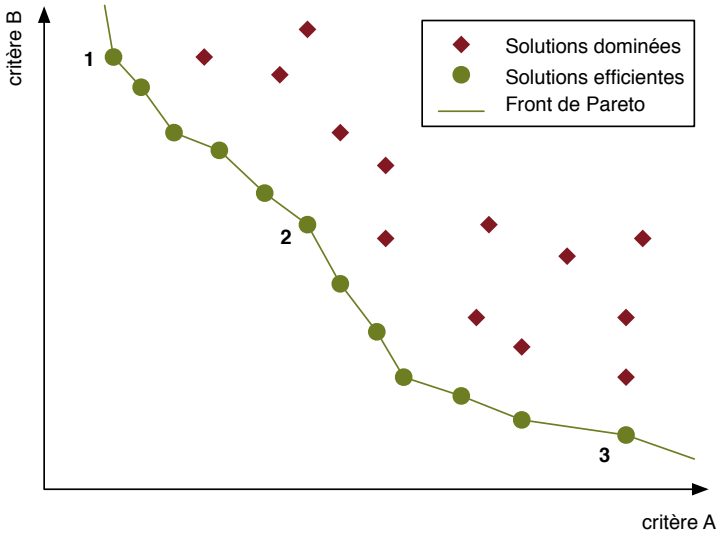

FIG. 5.1 – Solutions efficientes et solutions dominées d'un problème de minimisation à deux critères

Dès lors, le problème 5.1 ne se résout pas en *une* solution idéale *unique*, mais en un *ensemble* de solutions optimales dites *solutions efficientes* ou encore *solutions de Pareto*, ou *solutions non dominées*, ou *solutions de compromis*. Une solution efficiente ne peut être améliorée sur un des critères sans être simultanément dégradée sur un autre critère. Autrement dit, on ne peut améliorer conjointement l'ensemble des critères d'une solution efficiente.

Définition 5.2. ***Dominance de Pareto.*** *Soient x et y deux points d'un espace de recherche E sur lequel on a posé un problème de minimisation multicritère. On dira que x domine y au sens de Pareto (on notera $x \preceq y$) si et seulement si :*

$$\forall n \in \{1, \ldots, N\},\ f_n(x) \leq f_n(y) \tag{5.3}$$

On dira que x domine strictement y au sens de Pareto (on notera $x \prec y$) si et seulement si :

$$\begin{cases} \forall n \in \{1, \ldots, N\},\ f_n(x) \leq f_n(y) \\ \exists n_0 \in \{1, \ldots, N\},\ f_{n_0}(x) < f_{n_0}(y) \end{cases} \tag{5.4}$$

Définition 5.3. *L'ensemble des éléments non dominés de E est appelé* ***front de Pareto****. La résolution d'un problème multicritère consiste en la découverte du front de Pareto.*

La figure 5.1 illustre la notion de dominance au sens de Pareto dans le cas bi-critères. D'un point de vue formel, aucune des solutions de Pareto ne peut être préférée aux autres. La relation de dominance $\succ$ n'induit qu'un ordre partiel sur l'espace des critères, comme le montre figure 5.2. Pour un élément x donné, l'espace des critères se divise en trois régions selon que leurs éléments dominent x ($E \prec x$) , sont dominés par x ($x \prec E$) ou ne peuvent être comparés à x ($x \not\prec E \wedge E \not\prec x$).

Remarque 5.1. *Supposons maintenant que la figure 5.1 représente un problème d'optimisation bi-critères. Même si les solutions du front de Pareto ne peuvent en théorie être ordonnées,*

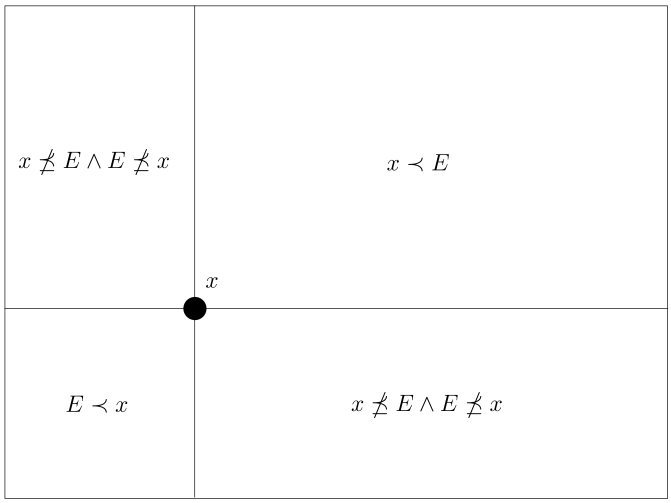

FIG. 5.2 – Relations de dominance dans un espace à deux critères pour un problème de minimisation

il y a fort à parier que certaines solutions non dominées soient plus pertinentes que d'autres pour le décideur. Si ce dernier émet une préférence pour la solution 1 par exemple, nous pouvons alors en déduire que le critère A est prédominant pour ce problème. Le décideur est en effet prêt à accepter une erreur conséquente sur le critère B pour satisfaire le critère A en premier. A l'opposé, si la solution 3 est préférée, c'est le critère B qui prime. Enfin, la solution 2 indique que les deux critères sont mobilisés à égale importance dans le choix d'une solution.

5.3 Classification des méthodes

Les méthodes d'optimisation combinatoire multicritère sont nombreuses. Les auteurs distinguent habituellement deux grandes catégories d'approches :

1. les *méthodes complètes*, qui garantissent l'optimalité des solutions, mais ne peuvent pas toujours être exécutées en un temps raisonnable ;
2. les *métaheuristiques*, qui permettent de découvrir rapidement des solutions approchées.

Dans la littérature, de nombreux problèmes combinatoires sont abordés à l'aide d'outils d'optimisation classiques telles que la programmation linéaire en nombres entiers ou la programmation dynamique. D'autres méthodes utilisent une représentation arborescente de l'espace de recherche, comme l'algorithme A^* ou les algorithmes de séparation et évaluation (*branch and bound*). Ces derniers imposent la possibilité d'estimer, pour chaque sous-arbre de recherche, des bornes inférieures des objectifs (dans le cas d'un problème de minimisation). Le sous-arbre courant n'est pas exploré si les bornes associées sont dominées par les configurations efficientes courantes. En général, le calcul de bornes « efficaces » dépend fortement des propriétés des fonctions objectifs (linéarité, monotonie, additivité...).

Si la plupart des auteurs s'accordent à dire que les méthodes complètes sont très efficaces pour une certaine classe de problèmes, Talbi [Tal99] fait observer que « pour des problèmes à plus de deux critères ou de grande taille, il n'existe aucune procédure exacte efficace, étant donné les difficultés simultanées de la complexité $\mathcal{NP}$ et le cadre multicritère des problèmes. »

Il sera donc illusoire de prétendre aborder le problème de l'orchestration avec des méthodes complètes, d'autant plus que les fonctions à optimiser ne sont ni linéaires, ni additives, ni monotones (voir annexe A).

Les métaheuristiques se proposent de contourner ces limitations en se restreignant à des solutions satisfaisantes bien que très souvent non-optimales. Le terme d'« heuristique » désigne une méthode permettant de calculer en temps polynomial une solution réalisable d'un problème d'optimisation $\mathcal{NP}$. L'efficacité d'une heuristique ne peut être démontrée, mais seulement vérifiée par l'expérience. En général, une heuristique s'appuie sur la structure propre du problème à résoudre et propose une méthode de recherche ad-hoc. A contrario, les « métaheuristiques » regroupent un ensemble de méthodes générales, d'un haut niveau d'abstraction, pouvant s'appliquer à différents problèmes.

L'abondante littérature sur les problèmes d'optimisation combinatoire multicritère a coutume de séparer les métaheuristiques en méthodes dites de « voisinage » et en méthodes « à population ». Les premières ont la particularité de n'utiliser, à tout moment de l'exécution, qu'une seule configuration, itérativement modifiée dans le but de converger vers le front de Pareto. Lorsque l'évolution n'est plus possible, la recherche peut éventuellement repartir d'une nouvelle configuration pour converger vers une autre zone de la frontière efficiente. Les méthodes de voisinage les plus utilisées aujourd'hui sont la recherche tabou, le recuit simulé et la méthode *GRASP (Greedy RAndomized Search Procedure)*.

A l'opposé, les méthodes « à population » utilisent un ensemble de configurations dont les coordonnées dans l'espace des décisions sont mises à jour à chaque itération de l'algorithme. L'intégralité de la population converge alors vers le front de Pareto, sans qu'aucune région de la frontière ne soit, en théorie, privilégiée. Les algorithmes génétiques (AGs) constituent la classe la plus ancienne et la plus utilisée des méthodes d'optimisation à population. Ils sont à l'origine de méthodes plus récentes comme les systèmes immunitaires artificiels et les algorithmes à évaluation de distribution. Une autre famille de méthodes, inspirées des phénomènes d'organisation complexe chez les animaux vivant en groupe, se base sur une collaboration entre les individus qui permet à la population de « s'auto-organiser » et de « s'adapter » au problème courant. Les plus célèbres sont les colonies de fourmis et l'optimisation par essaims particulaires.

Plus récémment, des auteurs ont proposé des méthodes hybrides, couplant en général un algorithme génétique à une méthode recherche locale. La figure 5.3 propose une classification des méthodes d'optimisation combinatoire multicritère. Le lecteur désirant davantage de détails pourra consulter les états de l'art dressés par Ulungu & Teghem [UT94], Talbi [Tal99], ou encore Ehrgott & Gandibleux [EG00].

5.4 Convexité du front de Pareto

Etant donné un problème de minimisation multicritère :

$$\min_k \ f_k \ ,$$

certaines solutions efficientes peuvent être trouvées en résolvant le problème uni-critère suivant :

$$\min \ \sum_k \ \lambda_k f_k \ , \tag{5.5}$$

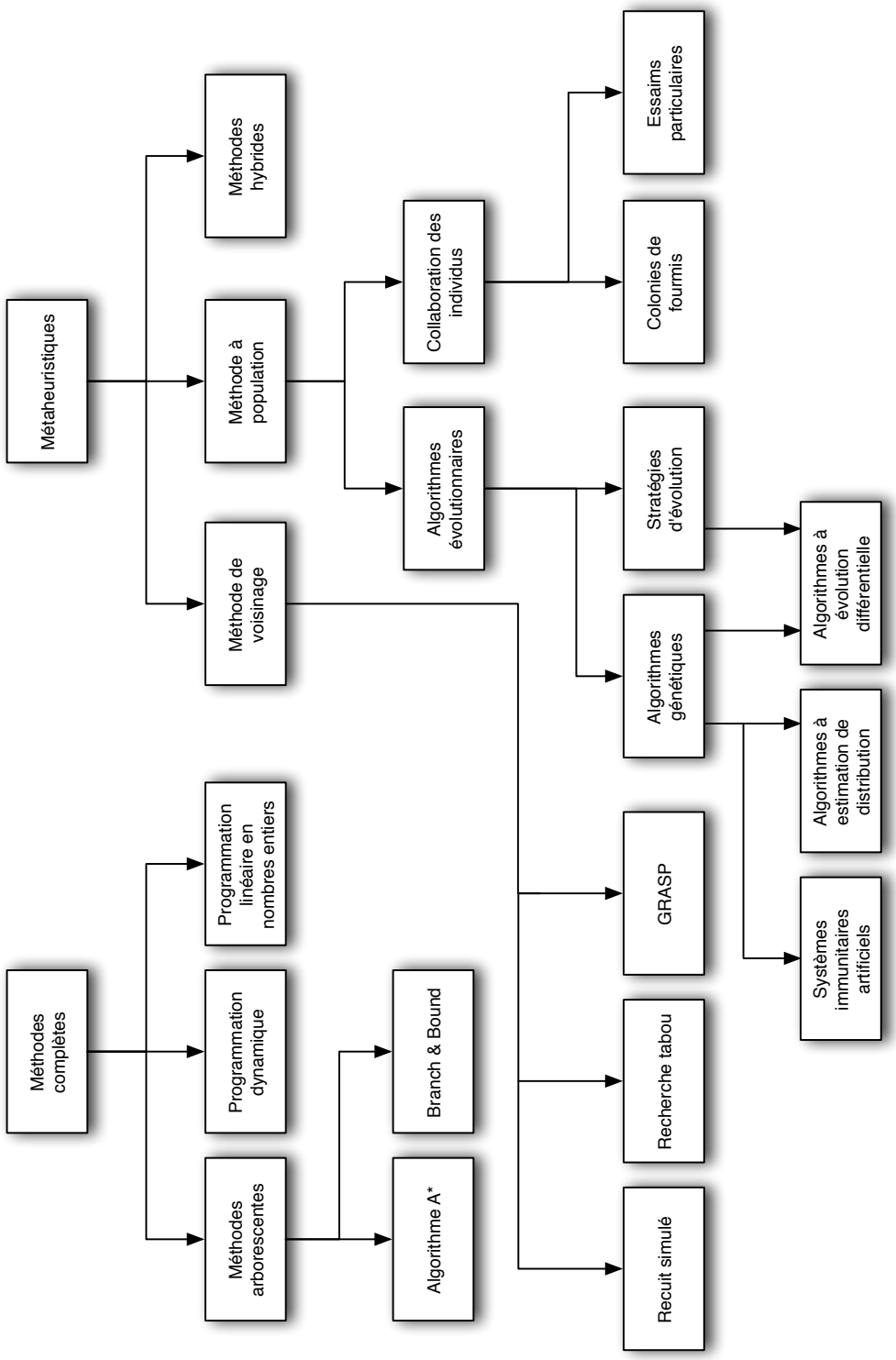

FIG. 5.3 – Classification des méthodes d'optimisation combinatoire multicritère

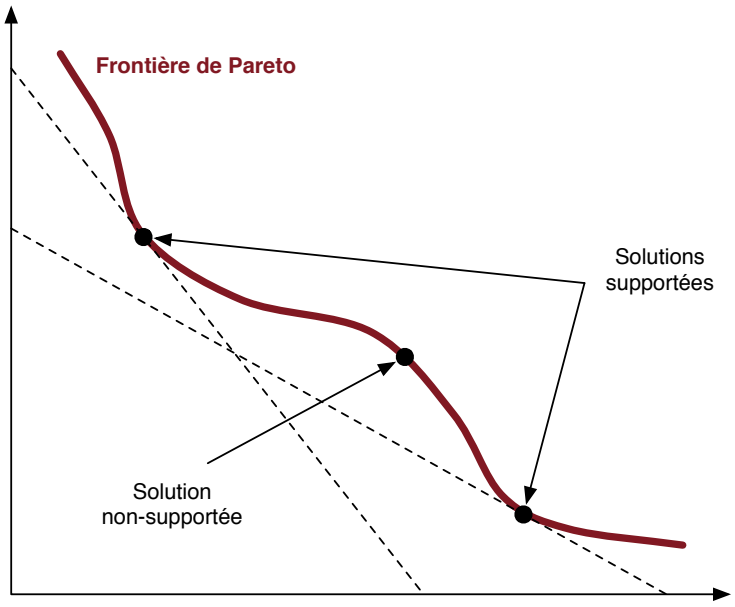

FIG. 5.4 – Solutions supportées et non-supportées dans un problème de minimisation

pour différents jeux de poids $(\lambda_1, \ldots, \lambda_k) \in \Lambda$. Les configurations efficientes solutions du problème 5.5 sont dites *supportées*. Elles se situent dans les zones convexes du front de Pareto (voir figure 5.4). Les solutions *non-supportées*, dans les zones concaves du front, ne peuvent pas être découvertes par la transformation vers l'uni-critère proposée en 5.5.

La première idée qui vient à l'esprit lorsqu'on a affaire à un problème multi-critère est évidemment de se ramener un problème uni-critère, afin de bénéficier des méthodes classiques d'optimisation. L'*agrégation linéaire* proposée en 5.5 est l'approche la plus intuitive. Une alternative est la méthode *ϵ-contrainte*, qui consiste à transformer $N-1$ des N objectifs en contraintes et à optimiser sur l'objectif restant. Une troisième approche, la *programmation par buts (goal programming)*, propose de minimiser la distance à un vecteur objectif de référence selon une norme $\mathbb{L}_p$ pondérée. Talbi [Tal99] note que seule cette dernière permet de découvrir des solutions non-supportées, à condition toutefois que le décideur soit capable de définir un but à atteindre et un ensemble de poids relatifs associés à chacun des critères. En pratique, cela nécessite une très bonne connaissance du problème et des fonctions objectifs qui le modélisent, ce qui est très rarement le cas.

Certains auteurs ont tenté d'aborder les problèmes multicritères en traitant séparément chaque objectif. La sélection des critères peut être menée de façon parallèle (algorithme de Schaffer), lexicographique (algorihme de Fourman) ou encore aléatoire (algorithme de Kursawe). Cependant, Talbi [Tal99] et Fonseca & Fleming [FF95] ont montré que ces algorithmes ne permettaient pas de trouver des solutions non-supportées.

Les méthodes énumérées ci-dessus sont souvent dénommées *approches non-Pareto*, car la dominance Pareto n'y est jamais prise en compte ni exploitée. Elles ne peuvent découvrir que des solutions supportées et sont donc inutilisables en cas de non-convexité du front de Pareto. A contrario, les *approches Pareto*, initialement introduites par Goldberg [Gol89], utilisent

directement la notion de dominance de Pareto dans l'évaluation et la mise à jour des configurations. Elles permettent notamment de trouver des solutions non-supportées. Les méthodes récentes d'optimisation combinatoire multicritère appartiennent toutes à cette catégorie. Les algorithmes génétiques en constituent la famille la plus répandue, nous les détaillons donc à la section suivante.

5.5 Introduction aux algorithmes génétiques

Les algorithmes génétiques, introduits en 1975 par Holland [Hol75], s'inspirent des mécanismes de sélection naturelle dans l'évolution des espèces. Au sein d'une population, les individus les moins adaptés à l'environnement disparaissent. Les autres survivent et transmettent leurs caractéristiques aux générations suivantes. La population devient ainsi, au cours des générations, de plus en plus adaptée à son environnement.

Un des intérêts majeurs des algorithmes génétiques est de ne faire, contrairement aux méthodes mathématiques plus classiques, aucune hypothèse quant à la nature des fonctions à optimiser. Ils peuvent donc en théorie traiter tous types de problèmes, et s'adapter à des fonctions non dérivables ou non continues, ce qui rend très vaste leur champ d'application.

5.5.1 Principes généraux

Les individus formant la population d'un AG sont représentés par des chaînes de caractères sur un alphabet A, et de longeur fixe L. Les éléments d'une chaîne de caractères sont appelés « gènes », et la chaîne elle-même constitue le « génome » (ou le « chromosome ») d'un individu. L'optimisation par algorithmes génétiques suppose donc l'existence d'une fonction de « lecture » du génome, associant à tout élément de A^L une configuration de l'espace de recherche. Traditionnellement, les auteurs préconisent un encodage binaire des individus ($A = \{0; 1\}$). Dans une étude théorique, Whitley [Whi93] a montré que les capacités exploratoires des AGs résolvant des problèmes continus étaient maximales avec des alphabets binaires. De nombreuses méthodes récentes ont toutefois recours à des alphabets de plus grande cardinalité, voire infinis (alphabets réels).

La conception d'un algorithme génétique repose sur quatre piliers fondamentaux :

1. une fonction de lecture permettant de représenter tout élément de l'espace de recherche par au moins un chromosome ;
2. une fonction d'évaluation à valeurs réelles, mesurant le niveau d'adaptabilité d'un individu à l'environnement, appelée couramment fonction de *fitness*[1] ;
3. un ensemble d'opérateurs génétiques, c'est-à-dire des opérations internes, d'arité variable, sur l'espace des chromosomes, permettant d'engendrer de nouveaux individus ;
4. des stratégies de sélection et de remplacement, qui gouvernent l'évolution de la population.

[1] Dans le cas de problèmes uni-critère, la fitness se confond en général avec la fonction à optimiser. En multicritère, les choix de fitness diffèrent selon les auteurs. Les approches les plus courantes sont d'une part l'agrégation des critères [IM96] [Jas02], d'autre part l'ensemble des méthodes tirant parti de la dominance de Pareto : « ranking » des individus [SD94], ou comptage des individus dominés ou dominants [ZT99] [ELT07].

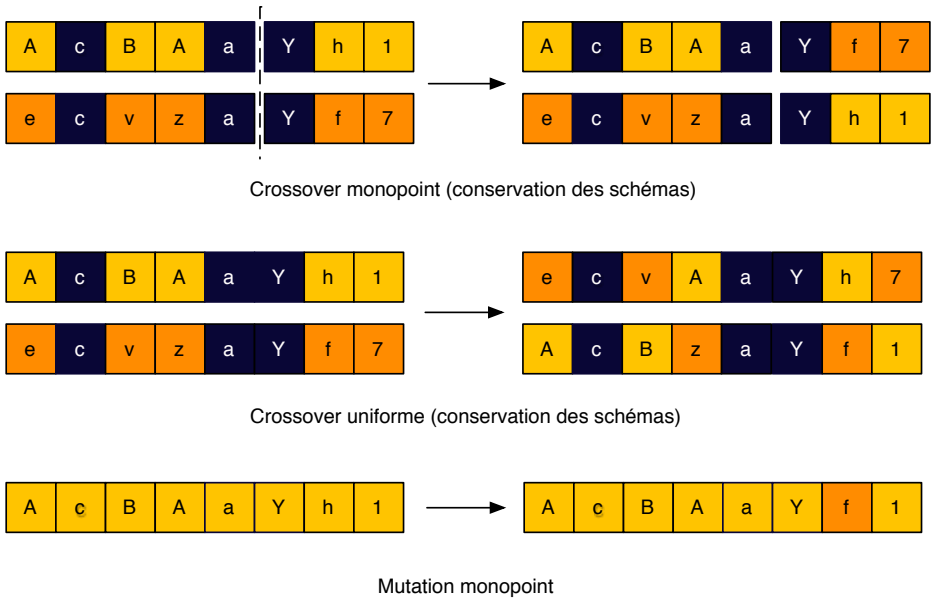

FIG. 5.5 – Opérateurs génétiques usuels

5.5.2 Opérateurs génétiques

En général, les opérateurs génétiques sont de deux types. On distingue les opérateurs binaires (souvent appelés *opérateurs de croisement*, ou *crossovers*), qui à une paire de chromosomes associent généralement un ou deux « rejetons », et les opérateurs unaires (*mutation*), n'agissant que sur un seul chromosome. Les opérateurs binaires imitent les mécanismes de la reproduction sexuée, dans laquelle les gènes de deux « parents » sont recombinés pour créer de nouveaux chromosomes, tandis que la mutation simule, comme son nom l'indique, des « accidents » survenant dans le génome au cours de l'évolution. Alors que les croisements favorisent la transmission des caractéristiques d'une génération à la suivante, les mutations en introduisent de nouvelles, parfois salutaires.

Les croisements les plus utilisés dans les AGs sont le crossover monopoint et le crossover uniforme. Dans le crossover monopoint, deux chromosomes sont « sectionnés » en un point aléatoire et les branches résultantes sont échangées. Dans le crossover uniforme, les gènes sont échangés aléatoirement entre les deux chromosomes. Quant à la mutation, elle consiste en général à altérer un seul gène du chromosome. La figure 5.5 illustre ces trois opérations.

Remarque 5.2. *Les opérations de crossover (aussi bien monopoint qu'uniforme) conservent les schémas génétiques (voir Holland [Hol75] et Whitley [Whi93]) : si les mêmes gènes sont présents chez les deux chromosomes parents (en bleu sur la figure 5.5), ils sont alors transmis aux deux chromosomes issus du crossover.*

5.5.3 Exploration et intensification

Comme toutes les métaheuristiques pour l'optimisation, les algorithmes génétiques doivent respecter l'équilibre délicat entre *intensification* (i.e. la découverte, dans le voisinage d'une

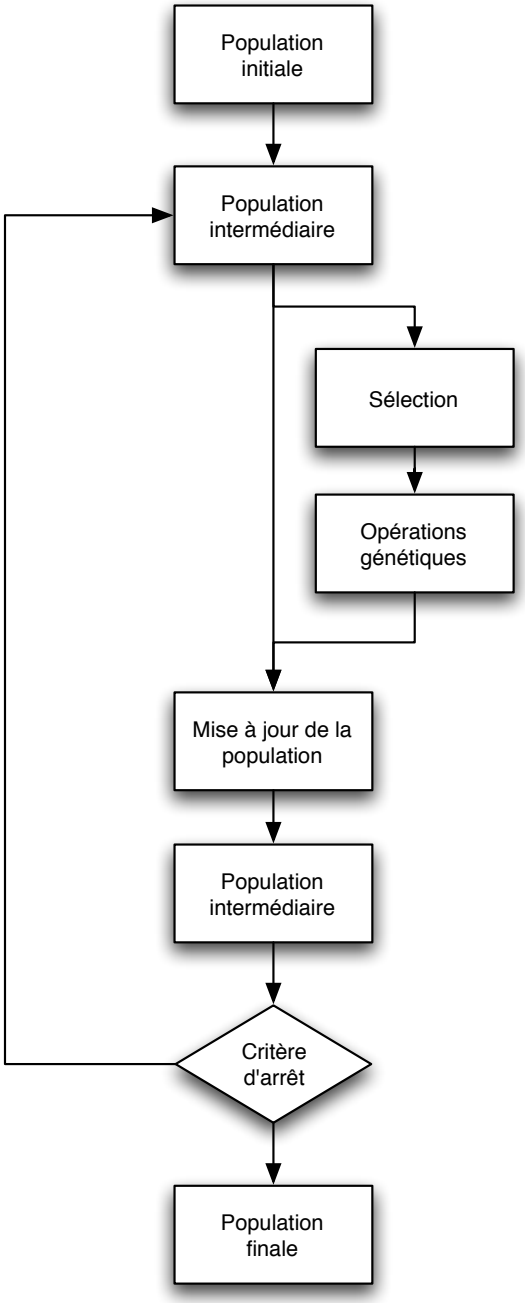

FIG. 5.6 – Schéma générique d'un algorithme génétique

« bonne » configuration, d'une solution encore meilleure) et *diversification* de la recherche (i.e. la découverte de bonnes solutions dans des zones différentes de l'espace de recherche, ou de l'espace des critères). Dans une approche multicritère, intensification et diversification correspondent respectivement à la convergence vers le front de Pareto et à la couverture de la frontière optimale. Une méthode privilégiant l'intensification risque de ne mettre en évidence qu'une seule région d'optimalité — c'est la *dérive génétique* des AGs (voir section 5.5) ; à l'inverse, privilégier la diversification se fait souvent au détriment de la convergence vers l'optimialité.

Dans les algorithmes génétiques, l'utilisation simultanée du crossover et de la mutation (voir figure 5.5) permet généralement de garantir à la fois l'intensification et la diversification de la recherche, sans que toutefois l'un ou l'autre des opérateurs ne corresponde à une tâche précise :

- Lorsque les deux chromosomes parents impliqués dans le crossover partagent des schémas génétiques communs, ce dernier intervient comme opérateur d'intensification de la recherche autour du schéma commun. En revanche, un croisement entre deux chromosomes très différents profite à la diversification de la population.
- La mutation ne modifiant généralement que quelques gènes (la plupart du temps un seul) du chromosome, elle agit a priori comme un opérateur d'intensification. Cependant, elle entraîne souvent un « saut qualitatif » majeur caractérisé par un déplacement significatif dans l'espace des critères.

D'une manière générale, crossover et mutation sont donc deux opérateurs d'intensification et de diversification de la recherche, leurs effets dépendant à la fois de la proximité génétique, en termes de schémas, des chromosomes parents, à la fois des discontinuités des fonctions conduisant de l'espace des chromosomes à l'espace des critères.

5.5.4 Evolution et sélection naturelle

Les mécanismes de sélection naturelle — responsables de l'élimination progressive des individus les moins adaptés à l'environnement — interviennent, au sein des algorithmes génétiques, à deux niveaux.

La stratégie de *sélection* permet d'isoler un sous-ensemble de la population courante, qui seul donnera naissance aux nouveaux individus. Le mécanisme qui consiste à favoriser les meilleurs individus dans la sélection est appelé *élitisme*. Selon les champs d'application et la conception des algorithmes, la sélection peut être :

- aléatoire (absence d'élitisme) ;
- totalement élitiste (seuls les meilleurs individus sont retenus) ;
- proportionnelle à la fitness (un individu a d'autant plus de chance d'être sélectionné qu'il est adapté à l'environnement — cette approche est connue sous les noms de *roulette wheel selection* et *stochatic remainder* [Gol89]) :
- basée sur un tournoi : sur plusieurs individus tirés aléatoirement dans la population, on retient ceux de meilleure fitness.

Les individus franchissant avec succès l'étape de sélection vont alors engendrer, par application des opérateurs génétiques, de nouvelles solutions. L'idée maitresse des AGs est que les régions optimales de l'espace de recherche sont caractérisées par des schémas génétiques

qui se transmettent au cours des générations. La mise à jour de la population — augmentée des nouvelles solutions — constitue alors le deuxième temps de la sélection naturelle. Si la population est de taille fixe, il s'agit de décider lesquelles des nouvelles solutions remplacent les anciennes. Si la population est de taille variable, il faut convenir d'un mécanisme de disparition progressive des individus les moins adaptés. Là encore, les approches diffèrent suivant les auteurs.

La principale difficulté, lors de la mise à jour de la population courante, consiste à encourager la convergence vers le front de Pareto tout en maintenant la diversité de la population dans l'espace des critères. Goldberg [Gol89] fait en effet remarquer que les AGs ont une tendance naturelle à « dériver » vers une zone particulière de la frontière de Pareto au fur et à mesure des générations, se détournant irrémédiablement des autres solutions optimales. Pour contrecarrer ce phénomène connu sous le nom de « dérive génétique » (*genetic drift*), les auteurs ont imaginé différentes approches pour mesurer la densité locale au sein de la population et favoriser l'exploration des zones les moins peuplées.

La figure 5.6 illustre le déroulement type d'un algorithme génétique. La plupart du temps, le critère d'arrêt est déterminé par un nombre maximum de générations ou d'évaluation de fitness.

5.6 Algorithmes de référence

Dans cette section nous présentons les algorithmes génétiques les plus célèbres, identifions leurs atouts et leurs travers. Cette discussion servira de référence pour la conception d'un AG adapté au problème de l'orchestration.

5.6.1 MOGA, NPGA, NSGA

Les premiers algorithmes génétiques à exploiter directement la dominance de Pareto dans l'évaluation des individus sont l'algorithme MOGA (Mutliple Objective Genetic Algorithm) de Fonseca et Fleming [FF93], NPGA (Niched Pareto Genetic Algorithm) de Horn, Nafpliotis & Goldberg [HNG94] et NSGA (Nondominated Sorting Genetic Algorithm) de Srinivas & Deb [SD94].

Dans l'algorithme NPGA, la sélection des individus pour la reproduction consiste à tirer deux individus au hasard dans la population courante, ainsi qu'un ensemble d'individus de taille t_{dom} qui va servir à déterminer le meilleur des deux candidats selon la relation de dominance de Pareto. Si NPGA est une méthode de référence dans l'industrie, certains auteurs [ZT98b] [SD94] font observer que le choix du paramètre t_{dom} est aussi délicat que fondamental pour la performance de l'algorithme. L'algorithme MOGA contourne ce problème avec un calcul de fitness basé sur le nombre de points dominants (au sens de Pareto) la solution à évaluer. Quant au NSGA, les solutions sont évaluées par extractions itératives de fronts de Pareto successifs, auxquels sont attribués des rangs croissants (les meilleurs individus ont le rang le plus faible). Deb et al. [DPAM00] regrettent toutefois que cette méthode de classement des solutions (connue sous le non de *non-dominated sorting*) ait une complexité en temps en $\mathcal{O}(KN^3)$, où K est le nombre de critères et N la taille de la population.

Les auteurs ont testé ces méthodes avec différents principes de sélection des individus pour la reproduction (voir section 5.5 et figure 5.6). Il semble que le tournoi binaire (*binary tournament*) outrepasse largement les performances des autres méthodes de sélection (*roulette*

wheel selection, stochastic remainder), en raison notamment de son insensibilité aux intervalles de valeurs prises par les critères. En outre, les auteurs mettent en garde contre un risque de comportement chaotique des AGs lorsque le tournoi binaire est combiné avec une méthode de partage (*sharing*), qui consite à pénaliser la fitness dans les zones de forte densité.

Bien que les algorithmes sus-cités aient été, ou continuent d'être, implémentés dans de nombreux systèmes d'aide à la décision, les recherches récentes en optimisation multicritère ont conduit à les critiquer sur un certain nombre de points. Deb et al. [DPAM00] identifient par exemple trois défauts majeurs de l'algorithme NSGA :

1. une complexité en temps souvent rédhibitoire pour des problèmes à grand nombre de critères ;
2. une convergence vers la frontière Pareto relativement lente, due à la disparition progressive de bonnes solutions au fur et à mesure des générations (absence d'élitisme) ;
3. des paramètres difficiles à calibrer et devant être adaptés à chaque nouvelle instance d'un problème.

Jaszkiewicz [Jas01] remarque en outre que la notion de dominance assure la convergence de la population vers l'ensemble de Pareto mais pas la dispersion sur l'intégralité de la frontière (c'est effet de « spéciation », ou « dérive génétique »).

Pour remédier à ces inconvénients, certains auteurs ont proposé de renforcer l'élitisme dans les AGs en conservant une « mémoire » des solutions optimales, servant de référence pour l'évaluation des individus. Les algorithmes SPEA, PAES, et NSGA-II sont conçus selon ce principe. D'autres auteurs [Tal99] [FF95] [BAKC99] font observer qu'en général les AGs et les méthodes de recherche locale (cf. section 5.3 et figure 5.3) donnent des solutions différentes et proposent des algorithmes hybrides combinant les deux méthodes (MOGLS, M-PAES, SPEA-II).

5.6.2 SPEA (I-II), PAES, M-PAES, NSGA-II, MOGLS (I-II-III)

L'algorithme SPEA (Strength Pareto Evolutionary Algorithm) a été proposé par Zitzler & Thiele [ZT98b] [ZT98a] [ZT99]. Les auteurs suggèrent de maintenir une population externe d'élite, rassemblant les solutions non-dominées courantes. Cet ensemble d'élite intervient dans toutes les opérations génétiques et sert également de bassin de sélection pour la reproduction. Une méthode de clustering permet de limiter la taille de l'ensemble d'élite et de s'assurer que les solutions de ce dernier se répartissent de façon uniforme sur l'approximation courante du front. La complexité de SPEA est en $\mathcal{O}(KN^3)$ pour l'implémentation suggérée dans [ZT98a], mais Deb et al. [DPAM00] ont montré qu'une structure de données adaptée permettait de réduire cette complexité à $\mathcal{O}(KN^2)$.

Dans leur algorithme PAES (Pareto-Archived Evolution Strategy), Knowles and Corne [KC99] utilisent un AG avec parent unique et enfant unique. Seul l'un des deux est retenu pour la génération suivante après comparaison à une archive des solutions optimales générées jusqu'alors. L'évaluation est basée sur la dominance Pareto et sur une mesure de voisinage dans l'espace des paramètres, garantissant une couverture uniforme de l'ensemble de Pareto. La diversité est maintenue en divisant l'espace de recherche en d^n sous-espaces dynamiquement actualisés, où d est la profondeur de la partition et n le nombre de variables de décision. La complexité de PAES est en $\mathcal{O}(aKN)$, où a est la taille de l'archive. Plus récemment,

les auteurs ont proposé M-PAES [KC00] (Memetic Pareto-Archived Evolution Strategy), une version hybride de l'algorithme PAES dans laquelle chaque progéniture est améliorée par une méthode de recherche locale avant le tournoi contre son géniteur. Au cours de cette phase d'optimisation, l'acceptation d'un nouveau voisinage est soumis aux mêmes règles que celle du tournoi final entre parent et enfant.

Deb et al. [DPAM00] ont proposé une amélioration de NSGA (cf. supra) corrigeant les défauts principaux de l'algorithme initial. La complexité de NSGA-II est ramenée à $\mathcal{O}(KN^2)$ et la diversité est maintenue grâce à une fonction d'estimation de la densité d'échantillonnage du front de Pareto. Par ailleurs, NSGA-II propose une gestion très pertinente des éventuelles contraintes, en introduisant une mesure de consistance (fonction de coût) de celles-ci dans la dominance Pareto.

En 1996, Ishibuchi & Murata [IM96] ont introduit une première version de l'algorithme MOGLS (Mutli Objective Genetic Local Search). A chaque génération les nouvelles solutions sont améliorées grâce à une méthode de recherche locale, optimisant une somme pondérée des critères, dont le jeu de poids est tiré aléatoirement. Jaszkiewicz [Jas01] fait observer que l'algorithme MOGLS, bien que basé sur une agrégation linaire des critères le rapprochant des méthodes de *goal programming*, n'effectue pas une transformation vers l'uni-objectif pour autant, du fait de sa structure d'AG permettant une recherche simultanée de solutions multiples dans des directions multiples. MOGLS tire parti des résultats de Chen et Liu [CL94] et Steuer [Ste86], prouvant que l'intégralité de l'ensemble Pareto peut être obtenu en résolvant un problème de goal programming pour l'ensemble de jeux de poids possibles.

Une seconde version de l'algorithme MOGLS a été proposée par Jaszkiewicz [Jas01] [Jas02]. Le mécanisme de *roulette wheel selection* est remplacé par un tirage aléatoire parmi les meilleurs candidats, et la somme pondérée par une distance de Tchebycheff pondérée (cf. définition 8.5.1). Jaszkiewicz [Jas02] fait observer que les fonctions de Tchebycheff sont plus adaptées que les fonctions linéaires pour des problèmes où la forme du front de Pareto est complexe ; elles permettent notamment de trouver des solutions non-supportées (voir section 5.4). L'auteur fournit en outre une méthode efficace pour calculer le nombre de solutions initiales.

Ishibushi et al. [IY02] ont suggéré une autre amélioration de l'algorithme MOGLS en exploitant l'idée que la performance de l'AG peut être accrue en utilisant dans la recherche locale une direction d'optimisation adaptée à chaque solution et indépendante du jeu de poids employés dans le mécanisme de sélection des parents. La méthode permet ainsi de séparer totalement évolution génétique et exploration du voisinage des solutions, et peut donc s'adapter très facilement à tout type de recherche locale. L'algorithme SPEA-II [ZLT02] exploite également une hybridation de ce type.

Par ailleurs, Ishibuchi et al. [IY02] font remarquer que dans de nombreux algorithmes hybrides la majeure partie du temps de calcul est dévouée à la recherche locale, et fournissent quelques pistes pour équilibrer la balance entre les deux metaheuristiques.

5.6.3 APAES, MAREA, MEMOTS, PFGA

Ce chapitre ne saurait prétendre recenser l'ensemble des techniques d'optimisation multicritère, tant les variantes dans les approches et cas d'application sont nombreux. Nous renvoyons le lecteur avide de connaissances aux ouvrages de référence mentionnés en section 5.1. Toutefois, deux innovations récentes nous semblent devoir encore être mentionnées, car elles introduisent deux nouveaux paradigmes dans la recherche multicritère à l'aide de métaheuristiques.

Grosan et al. font remarquer que les alphabets binaires ne sont pas adaptés à tous les problèmes, et qu'il existe de nombreux cas où il est plus judicieux d'utiliser une autre encodage des solutions. Malheureusement, le choix d'un alphabet nécessite une très bonne connaissance de l'algorithme employé et du problème à résoudre, laquelle ne peut s'acquérir qu'a posteriori, et par un décideur maîtrisant les concepts scientifiques des algorithmes génétiques.

Les auteurs ont donc proposé de laisser à l'algorithme de choix de l'alphabet, en codant la taille de ce dernier directement dans le génome. Les opérateurs génétiques permettent alors soit de modifier le génome au sein du même alphabet (*mutation*), soit de traduire l'information portée par les gènes dans un autre alphabet (*transmutation*). De façon évidente, le crossover ne peut s'appliquer, car les chromosomes apairés ne partagent pas nécessairement le même alphabet.

Ce principe d'adaptabilité de l'alphabet a été exploité dans les aglorithmes APAES (Adaptive Pareto-Archive Evolution Strategy) [OGAK05] et MAREA [Gro06]. Dans chacune de ces deux méthodes, lorsqu'un individu ne peut plus être amélioré par mutation génétique, une transmutation est appliquée pour changer d'alphabet.

Une autre évolution récente dans les métaheuristiques pour l'optimisation multicritère concerne le maintien de la diversité dans la population. La préservation de la diversité est essentielle dans une méthode à population, car elle empêche que l'ensemble des individus soit attiré par un région unique de l'espace. En recherche uni-critère, elle permet d'éviter le piège des minima locaux ; en multicritère, elle garantit une répartition uniforme des solutions le long du front de Pareto.

Traditionnellement, le maintien de la diversité passe par une estimation de la densité locale dans l'espace des critères, afin soit de pénaliser les zones les plus denses lors de la sélection pour la reproduction, soit d'en retirer des individus lors de la mise à jour de la population. Elle est basée sur la notion de « voisinage » dans l'espace des critères, définie à l'aide de paramètres comme le rayon de voisinage ou la distance aux plus proches voisins. Hélas, les valeurs de ces paramètres sont très difficiles à définir ou à interpréter, car elles dépendent fortement du problème considéré.

Lust & Teghem [LT06] ont donc proposé l'algorithme MEMOTS (Memetic Multi-Objective Tabu Search), dans lequel la densité est calculée à l'aide d'une hypergrille divisant l'espace des critères en hypervolumes. La densité locale y est alors définie comme le nombre de solutions dans chaque hypervolume. Les auteurs fournissent également une méthode simple pour définir la taille de l'hypergrille en fonction de la taille de la population.

Une autre approhe est proposée par Elaoud et al. [ELT07] avec l'algorithme PFGA (Pareto Fitness Genetic Algorithm). Là encore, une hypergrille est utilisée, mais celle-ci s'adapte automatiquement à la taille de la population et aux échelles de valeurs prises par les critères. La mesure de la densité locale par la méthode PADE (Population size Adaptive Density Estimation) ne nécessite aucun paramètre dont la valeur doit être définie par le décideur et peut donc en théorie s'adapter à tout type de problème. Cette méthode est détaillée à la section 5.8

5.7 Fonctions « scalarisantes » de Tchebycheff

Une des étapes les plus délicates dans un algorithme génétique multicritère est sans doute l'évaluation de la qualité des individus, ou *calcul de fitness*. Les algorithmes NSGA [SD94] et NSGA-II [DPAM00] utilisent directement le rang (au sens de pareto) des individus, alors que

d'autres se basent sur le dénombrement des configurations dominant et/ou dominées par la configuration évaluée. SPEA [ZT98a] [ZT98b] [ZT99], SPEA-II [ZLT02], MEMOTS [LT06] et PFGA [ELT07] fonctionnent sur ce principe. Toutes ces méthodes corrigent parfois la valeur de fitness à l'aide d'une estimation de densité locale, afin de défavoriser les zones les plus peuplées de l'espace des critères au profit des moins denses. Une troisième approche, exploitée dans les différentes versions de l'algorithme MOGLS [IM96] [Jas01] [Jas02], procède à une agrégation des critères par normes de Tchebycheff pondérées aléatoirement (voir définition 8.2).

L'agrégation par normes de Tchebycheff pondérées (dites aussi *fonctions scalarisantes de Tchebycheff*) s'appuie sur une propriété énoncée par Chen & Liu [CL94] et Steuer [Ste86], selon laquelle un problème multicritère est équivalent à un ensemble de problèmes mono-critère « scalarisés ».

Définition 5.4. *Soit Λ la partie de $[0;1]^N$ telle que :*

$$\forall \lambda = (\lambda_1, \ldots, \lambda_N) \in \Lambda,\ \sum_i \lambda_i = 1 \tag{5.6}$$

Soit $\lambda = (\lambda_1, \ldots, \lambda_N) \in \Lambda$. La norme définie sur un espace de critères C, de dimension N, par :

$$\forall x \in C,\ \|x\|_\lambda = \max_i\ \lambda_i x_i \tag{5.7}$$

est appelée ***norme de Tchebycheff pondérée*** *par le jeu de poids $\lambda = (\lambda_1, \ldots, \lambda_N)$.*

Propriété 5.1. *Avec les notations introduites à la définition 5.4, une solution x appartient au front de Pareto si et seulement si :*

$$\exists \lambda \in \Lambda,\ x = \underset{y \in C}{argmin} \|y\|_\lambda\ ,$$

La démonstration de la propriété 5.1 utilise la notion de *norme induite* qui sera introduite au paragraphe 8.5.1.

La figure 5.7 illustre cette dualité entre un problème multicritère et un ensemble de problèmes mono-critère. Les trois configurations efficientes x_1, x_2 et x_3 sont respectivement solutions de problèmes de minimisation des normes N_1, N_2 et N_3. Pour chacune d'entre elles, les jeux de poids associés définissent des « directions d'optimisation » différentes. Notons que les normes de Tchebycheff pondérées, par les boules hyper-rectangulaires qu'elles induisent, permettent de « s'adapter » aux portions concaves du front : les solutions non-supportées (voir section 5.4) peuvent être découvertes par minimisation d'une norme de Tchebycheff, ce que l'agrégation linéaire des critères ne permet pas.

Dans les algorithmes MOGLS, un nouveau jeu de poids est tiré aléatoirement à chaque itération. La population est donc chaque fois évaluée avec une fonction scalarisante différente. La minimisation de la norme pondérée courante assure la convergence vers une zone particulière du front de Pareto, tandis que le renouvellement des poids favorise la diversité des solutions le long de la frontière efficiente. Jaszkiewicz expose dans [Jas01] et [Jas02] une méthode efficace pour générer uniformément des poids aléatoires normalisés. Elle garantit une couverture

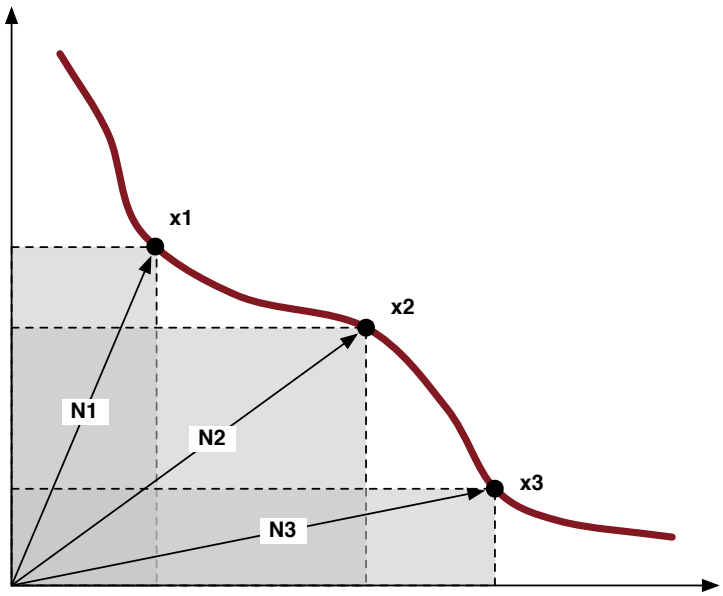

FIG. 5.7 – Trois solutions efficientes obtenues par optimisation unicritère dans un problème de minimisation

uniforme de l'espace des préférences :

$$\begin{cases} \lambda_1 = 1 - \sqrt[K-1]{X} \\ \dots \\ \lambda_k = \left(1 - \sum_{l=1}^{k-1} \lambda_l\right) \left(1 - \sqrt[K-1-k]{X}\right) \\ \dots \\ \lambda_K = 1 - \sum_{l=1}^{K-1} \lambda_l \end{cases} \tag{5.8}$$

où X est une variable aléatoire uniforme dans $[0;1]$.

En pratique, si $\lambda = (\lambda_1, \dots, \lambda_K) \in \Lambda$ est le jeu de poids courant, un point x de l'espace des critères est évalué à l'aide de la fonction de fitness suivante :

$$f(\lambda, x) = \max_{1 \leq k \leq K} \lambda_k \bar{x}_k \ , \tag{5.9}$$

où $(\bar{x}_1, \dots, \bar{x}_K)$ sont les coordonnées « réduites » de x :

$$\bar{x}_k = \frac{x_k - x_k^{min}}{x_k^{max} - x_k^{min}}$$

Les coordonnées réduites permettent de s'affranchir des différentes échelles de valeurs prises par chacun des critères. Pour chaque dimension k, x_k^{min} et x_k^{max} sont estimées par les plus petite et plus grande valeurs de x_k obtenues depuis le début de l'exécution de l'algorithme.

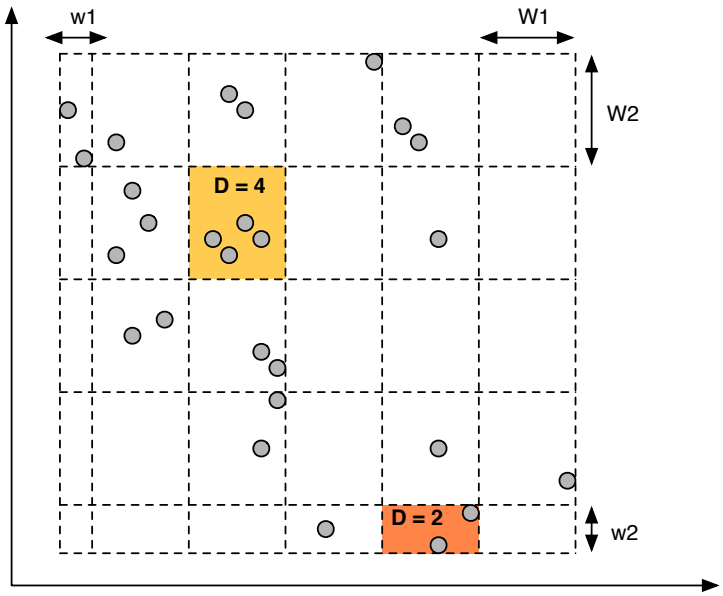

FIG. 5.8 – Estimation de la densité locale par la méthode PADE : Population size Adaptive Density Estimation (d'après Elaoud & Teghem [ELT07]).

5.8 Maintien de la diversité par la méthode PADE

Afin d'éviter la convergence prématurée de la population vers une zone particulière de l'espace de recherche (*genetic drift*), la littérature propose des méthodes de « partage » (*sharing procedures*). Elles évaluent dans un premier temps la densité locale de la population, puis pénalisent les valeurs de fitness dans les zones les plus denses. En général, le calcul de la densité autour d'une solution est basée sur la distance moyenne aux N plus proches voisins ou sur le nombre d'individus situés à un distance inférieure à d_{max} de la solution. Les auteurs s'accordent cependant sur la difficulté de déterminer des valeurs adéquates pour les paramètres N ou d_{max}, pourtant cruciales pour les performances des algorithmes.

Face à ce constat, Elaoud & Teghem [ELT07] ont proposé un principe efficace de calcul de densité, s'adaptant à la population et ne nécessitant aucun paramètre extérieur. La méthode PADE (*Population size Adaptive Density Estimation*) part du principe qu'une population de N individus est uniformément répartie dans l'espace des critères lorsque chaque individu occupe une fraction $1/N$ de l'espace total. Les auteurs suggèrent donc de « découper » l'espace de critères en N cellules et de définir la densité de chaque cellule comme le nombre d'individus qu'elle contient. La densité locale associée à chaque individu est alors égale à la densité de la cellule qui le contient. La complexité de la méthode PADE est en $\mathcal{O}(NK)$.

En pratique, si X est la population courante, la méthode PADE quadrille l'hyper-rectangle de diagonale :

$$\left[\left(\min_{x\in X} f_k(x)\right)_k ; \left(\max_{x\in X} f_k(x)\right)_k\right]$$

en un ensemble de cellules dont la largeur W_k sur chaque objectif k est donnée par :

$$W_k = \frac{\max_{x \in X} f_k(x) - \min_{x \in X} f_k(x)}{\sqrt[K]{N}}$$

Le nombre n_c de cellules sur chaque dimension est donc égal à $E(\sqrt[K]{N}) + 1$, où E désigne la fonction partie entière, et le nombre total de cellules à n_c^K, légèrement supérieur à N. Si $\sqrt[K]{N}$ n'est pas un nombre entier, alors les cellules les plus proches des axes ont une largeur donnée par :

$$w_k = W_k \left(\sqrt[K]{N} - E(\sqrt[K]{N}) \right)$$

La figure 5.8 illustre le calcul de densité par la méthode PADE dans un cas bicritère.

Conclusion

Les méthodes multicritères permettent d'aborder des problèmes d'optimisation pour lesquels l'importance relative des objectifs ne peut être connue a priori. Cependant, la complexité $\mathcal{NP}$ de la plupart des problèmes combinatoires multicritères les rend inabordables par les méthodes complètes, et le recours aux métaheuristiques est la plupart du temps incontournable. Les algorithmes génétiques en constituent la variante la plus courante, et leur efficacité dépend avant tout des méthodes choisies pour évaluer et diversifier les solutions.

Chapitre 6

Programmation par contraintes

Programmation par contraintes et informatique musicale
Méthodes complètes et métaheuristiques
Recherche adaptative et heuristique $\mathcal{CN}$-Tabou
Gestion des contraintes au sein les algorithmes génétiques

Sans anticiper sur la seconde partie de ce document, il est aisé de comprendre que la recherche d'orchestrations ne se ramène pas uniquement à une tache d'optimisation, mais implique également un ensemble de conditions que les solutions proposées se doivent de vérifier. Dans sa version actuelle, notre outil est une aide à l'écriture pour orchestre ; aussi les orchestrations qu'il suggère doivent-elles se plier aux contraintes imposées par l'effectif instrumental : une orchestration nécessitant huit trombones n'est pas réalisable par un orchestre symphonique classique, qui n'en comporte que trois ou quatre. En outre, comme nous le verrons au chapitre 10, il doit être possible pour le compositeur d'exiger, par exemple, que les orchestrations découvertes par notre système ne mobilisent qu'une partie de l'orchestre, qu'elle laissent à la disposition du compositeur au moins un instrument de chaque type ou de chaque famille, qu'elles couvrent une certaine « densité » harmonique (nombre de notes de hauteurs différentes jouées simultanément), etc. La formalisation et la gestion de contraintes au sein de notre système est donc inévitable. Nous proposons dans ce chapitre un aperçu des principes de la programmation par contraintes et des méthodes de résolution s'y rapportant. Nous détaillons en outre deux approches récentes, la recherche adaptative et l'heuristique $\mathcal{CN}$-tabou, dont s'inspire notre algorithme *CDCSolver* présenté au chapitre 10.

6.1 Principes généraux

La *programmation par contraintes* (PPC) est un paradigme de *programmation déclarative*, désignant à la fois un formalisme pour la définition de CSPs (*Constraint Satisfaction Problems*) et un ensemble de méthodes pour les résoudre. La donnée d'un CSP consiste en l'énumération de relations logiques (ou *contraintes*) portant sur les variables du problème. Le problème est résolu dès que l'on découvre une configuration dite *consistante*, c'est-à-dire un ensemble de valeurs telles que toutes les relations logiques entre les variables du problème soit vérifiées. S'il n'existe aucune configuration consistante, le problème est dit *surcontraint*.

La formulation d'un problème de satisfaction de contraintes est donné par un ensemble de variables $V_1, \ldots, V_N$ à valeurs dans des domaines $D_1, \ldots, D_N$ (finis ou infinis), ainsi que d'un

jeu de contraintes (ou *réseau de contraintes*) $C_1, \ldots, C_p$, fonctions booléennes sur $(\bigotimes D_i)_{i \in I}$, où I est un sous-ensemble de $\{1, \ldots, N\}$. On parle de *contrainte unaire* (ou de *filtre*) si I est un singleton, de *contrainte binaire* si I est un couple, de *contrainte N-aire* sinon. De façon générale, une contrainte représente une information partielle sur les valeurs que peuvent prendre les variables d'un problème donné. La programmation par contraintes propose des méthodes de résolution basées sur la manipulation et la propagation de ces informations partielles au cours du calcul. Lorsqu'on « avance » dans la résolution, les incertitudes sur les valeurs autorisées pour les variables sont peu à peu levées, et une configuration consistante est alors synonyme d'information totale.

En pratique, de nombreux domaines d'application se prêtent à la programmation par contraintes, en raison notamment du fort pouvoir expressif de ce paradigme. De la mise en page des documents multimédia aux problèmes de planification, d'ordonnancement, de gestion de ressources ou d'emplois du temps, les contraintes ont été employées avec succès dans des situations complexes, difficiles à formaliser autrement que par un ensemble de conditions logiques sur les variables.

Le lecteur désirant approfondir ses connaissances sur la programmtion par contraintes pourra consulter les ouvrages de Tsang [Tsa93] ou Mariott & Stuckey [MS98].

6.2 Contraintes et informatique musicale

La programmation par contraintes entretient depuis une vingtaine d'années des liens étroits avec l'informatique musicale[1], et en particulier la composition assistée par ordinateur, en raison de la facilité avec laquelle la musique symbolique se prête à une formalisation sous forme de règles. L'utilisation de contraintes en CAO commence à la fin des années 80 avec des problèmes d'harmonisation automatique de mélodies. Avec son système CHORAL conçu pour l'harmonisation de chorals à quatre voix dans le style de Jean-Sébastien Bach, Kemal Ebcioglu [Ebc88] fut le précurseur d'un champ de recherche à part entière (pour état de l'art complet du domaine, voir la revue de Pachet & Roy [PR01]).

A un niveau plus général, le moteur *PWConstraints* de Mikaël Laurson [Lau93] permet de définir des contraintes sur des séquences dans l'environnement de CAO PATCHWORK [LD89]. Plus tard, Rueda & Bonnet [RLBA98] ont développé dans PATCHWORK la bibliothèque SITUATION, dans laquelle les structures musicales sont représentées sous forme d'objets bi-dimensionnels, permettant de prendre en compte contraintes harmoniques et contraintes temporelles au sein d'une même représentation. *Situation* est aujourd'hui intégré dans l'environnement OPENMUSIC [Ago98]. Enfin les travaux de Charlotte Truchet [Tru04] au sein de l'équipe Représentations musicales de l'IRCAM ont débouché sur la création dans OPENMUSIC de la bibliothèque *OMClouds*. Cette dernière tire parti du paradigme de programmation visuelle d'OPENMUSIC pour établir une correspondance entre un graphe de contraintes et la représentation visuelle d'un programme. L'originalité d'*OMClouds* est de proposer une collection d'opérateurs et de fonctions suffisamment génériques pour formaliser un grand nombre de problèmes musicaux.

La programmation par contraintes a également été utilisée dans le domaine temporel avec

[1] L'Association Française d'Informatique Musicale (AFIM) a d'alleurs encouragé en 2007 la création d'un groupe de travail *Contraintes, Musique et Interaction*, dont nous faisons partie. Ce groupe est également soutenu par l'Association française de programmation par contraintes. `http://www.lina.sciences.univ-nantes.fr/ContraintesMusique/`

l'environnement de CAO Boxes d'Antony Beurivé [Beu00]. Boxes est basé sur une représentation hiérarchique de type conteneurs/contenus assortis de paramètres temporels (début, durée, fin). Le compositeur peut ainsi facilement organiser dans le temps des objets sonores par la simple donnée de contraintes d'antériorité, de postérité, ou de synchronisation.

Antoine Allombert et Myriam Desainte-Catherine [AADC08] ont poursuivi et généralisé ce travail avec le concept de *partitions interactives*, permettant aux compositeurs de créer des pièces musicales dans lesquelles des libertés sont laissées aux interprètes quant aux déclenchements de certains événements musicaux. Les partitions interactives ont recours à un formalisme basé sur les relations temporelles de Allen[2] grâce auquel les compositeurs peuvent définir un ensemble de contraintes temporelles sur les événements.

Dans un tout autre registre, citons encore le programme MusicSpace d'Olivier Delerue [Del04], dans lequel les connaissances sur la spatialisation sonore sont modélisées par un ensemble de relations entre les positions des sources et de l'auditeur. MusicSpace allie à une représentation graphique des scènes sonores un moteur de contraintes permettant d'en respecter la cohérence spatiale.

6.3 Méthodes complètes de résolution

A l'instar de l'optimisation multicritère (voir chapitre 5), les méthodes de résolution en programmation par contraintes se distinguent en deux catégories :

- les *méthodes complètes*, qui garantissent la découverte d'une configuration consistante si le problème n'est pas surcontraint, mais au prix d'un temps de calcul souvent très long ;
- les *métaheuristiques*, qui permettent de trouver rapidement une solution dont la consistance peut être seulement partielle (i.e. toutes les contraintes ne sont pas satisfaites). Elle feront l'objet du paragraphe suivant.

6.3.1 Recherche systématique

Les méthodes de recherche systématique utilisent une représentation arborescente de l'espace de recherche. Chaque nœud correspond à l'instanciation d'une variable et chaque feuille représente une configuration. L'idée la plus simple pour résoudre un problème de satisfaction de contraintes consiste à parcourir l'arbre de recherche en profondeur jusqu'à l'obtention d'une configuration consistante : c'est la méthode dite *generate-and-test*. Une telle procédure est évidemment totalement inefficace en pratique, en raison du caractère $\mathcal{NP}$ de la plupart des problèmes.

L'alternative la plus courante est le *backtracking*, exploitant la notion d'« instanciation partielle ». Dans la plupart de problèmes, chaque contrainte est d'arité faible : elle n'implique qu'un nombre réduit de variables[3]. Il n'est donc pas nécessaire de connaître les valeurs de *toutes* les variables pour savoir si la contrainte est violée ou non. En pratique, un algorithme de backtracking instancie itérativement les variables (donc descend en profondeur dans l'arbre de recherche) jusqu'à ce qu'une contrainte soit violée : le sous arbre courant n'est alors pas

[2] Antériorité, postériorité, succession immédiate, recouvrement, synchronicité des débuts, synchronicité des fins, inclusion temporelle, égalité temporelle.

[3] Si ce n'est pas le cas, il est souvent possible d'exprimer une contrainte globale (portant sur toutes les variables) par un ensemble de contraintes locales. Par exemple, une contrainte *alldiff* (toutes les variables doivent avoir des valeurs différentes) sur un problème à trois variables A, B, C est équivalent à $A \neq B \wedge A \neq C \wedge B \neq C$.

exploré et on modifie la valeur de la variable dernièrement instanciée. Si ce n'est pas possible, on remonte alors à un niveau supérieur de l'arbre pour changer l'une des variables précédentes. L'algorithme termine lorsqu'une des feuilles est atteinte ou que l'arbre entier a été exploré.

Comme le generate-and-test, le backtracking est inefficace en pratique (sa complexité est exponentielle dans le pire cas), en raison d'une gestion tardive des conflits.

6.3.2 Méthodes de filtrage (ou de consistance)

Les méthodes de filtrage ont pour but de réduire les domaines des variables en détectant les valeurs *inconsistantes*, i.e. ne pouvant prendre part à une solution. Une valeur v d'une variable V est dit consistante si et seulement si pour toute contrainte C impliquant V, on peut trouver une instanciation des autres variables impliquées par C compatible avec $V = v$. Un problème est dit *localement consistant* si on peut trouver au moins une valeur consistante pour chaque variable.

La consistance la plus simple est la *consistance de nœud*, qui ne filtre que les variables associées à des contraintes *unaires* (n'impliquant qu'une seule variable). Elle est très rapide, mais le plus souvent inefficace : la réduction des domaines par consistance de nœud est très faible.

On parle de *consistance d'arc* — ou d'*arc-consistance* — pour les contraintes *binaires*. Celles-ci n'impliquent jamais que deux variables à la fois, et couvrent une grande proportion des problèmes de satisfaction de contraintes : de nombreuses contraintes peuvent en effet se décomposer en un ensemble de contraintes binaires. La consistance d'arc se généralise aux contraintes N-aires avec la notion de *consistance de chemin*. En pratique, les méthodes de consistance se limitent en général à la consistance d'arc : le temps de calcul d'une consistance de chemin est dans la plupart des cas rédhibitoire pour des contraintes dont l'arité dépasse trois variables. Aussi l'arc-consistance est-elle bien souvent le meilleur compromis entre le temps nécessaire au filtrage et l'efficacité de celui-ci.

Dans le cas de problèmes à variables continues ou pouvant prendre un très grand nombre de valeurs, la *consistance de bornes* est préférée à la consistance de domaines : elle consiste à raisonner sur les valeurs minimum et maximum que les variables peuvent prendre. Dans le cas de contraintes binaires d'inégalité, la consistance de bornes est équivalente à l'arc-consistance.

Les algorithmes de résolution par filtrage commencent généralement par réduire les domaines de chaque variable. Si l'un des domaines est vide, le problème est surcontraint et n'a pas de solution. Sinon, les variables dont les domaines filtrés sont des singletons sont instanciées à ces valeurs. Malheureusement, il reste dans la plupart des cas au moins une variable V non instanciée ; on choisit alors une valeur v possible (dans le domaine filtré) pour V et on recalcule la consistance avec la nouvelle contrainte $V = v$. Si le nouveau problème n'est pas consistant, v est retirée du domaine de V.

La résolution d'un problème de contrainte par calcul de consistance se base donc sur un algorithme de point fixe, dont le nombre d'itérations ne peut être connu à l'avance. Dans le cas de problème de grande taille ou en présence de contraintes d'arité élevées, ce processus paut être très long. En général, on implémente un algorithme de backtracking avec maintien de l'arc-consistance pour les variables encore non instanciées. De plus, afin d'alléger le calcul de consistance, on se limite fréquemment à une une seule itération de l'algorithme de poins fixe (chaque couple de variables n'est vérifié qu'une seule fois). Ces techniques sont dites *look-ahead*.

6.3.3 Méthodes de propagation

Les méthodes de propagation réalisent un compromis entre réduction des domaines et calcul de consistance, à travers la notion de *consistance partielle* : lors de l'instanciation d'une variable V, on vérifie d'abord que la nouvelle valeur ne génère pas de conflits avec les variables instanciées jusqu'alors (c'est le principe du backtracking), puis on « propage » l'information apportée par la nouvelle instanciation en filtrant les domaines de toutes variables en relation avec V dans le réseau de contraintes. Cette méthode est connue sous le nom de *forward-checking*. Elle est parfois plus efficace que de maintenir la consistance d'arc sur les variables non instanciées car elle ne filtre qu'une seule variable à la fois, là où les méthodes de look-ahead filtrent des paires de variables. En termes de parcours arborescent, le forward-checking visite davantage de nœuds que le look-ahead mais le coût de calcul à chaque nœud y est moins important. En vérité, les performances diffèrent selon le problème considéré.

6.3.4 Contraintes molles

En tant que relation logique sur les variables du problème, chaque contrainte n'apporte qu'une infomration binaire sur une configuration : soit la contrainte est vérifiée, soit elle ne l'est pas. Les contraintes molles (ou *soft constraints*) permettent davantage de souplesse dans la gestion des configurations inconsistantes. A l'instar de la logique floue, les contraintes molles sont exprimées au travers de fonctions non plus booléennes, mais à valeurs réelles. On parle alors de « fonctions de coût », traduisant à quel point les contraintes sont violées. Dans une modélisation en contraintes molles, le réseau de contraintes n'est pas considéré comme un ensemble de conditions à respecter, mais comme une structure sur la base de laquelle on peut donner une « mesure de qualité » des configurations.

Au cours de la résolution, on va chercher à minimiser les fonctions de coût associées aux contraintes. Si le problème est surcontraint, on essaiera d'obtenir la solution la plus consistante, i.e. qui viole le moins les contraintes. Les contraintes molles permettent ainsi de préférer certaines solutions (celles dont l'écart à la consistance est le plus faible) à d'autres.

D'un certain point de vue, cette approche permet donc d'aborder les contraintes sous l'angle de l'optimisation. On y a fréquemment recours pour les problèmes de grandes tailles, qu'on résout le plus souvent à l'aide de métaheuristiques.

6.4 Métaheuristiques

Les métaheuristiques diffèrent des méthodes complètes en ce sens qu'elles n'explorent pas l'arbre de recherche de façon systématique (elles ne peuvent donc assurer de toujours découvrir des solutions globalement consistantes avec le réseau de contraintes), mais procèdent à un « affinage » itératif d'une ou plusieurs solutions courantes, en général totalement instanciées.

les métaheuristiques couvrent une large classe de méthodes que nous détailleront pas intégralement ici. Nous nous contentons de présenter celles en rapport direct avec la programmation par contraintes.

6.4.1 Méthodes de recherche locale

Les méthodes de recherche locale s'appuient toutes sur l'amélioration itérative d'une unique configuration, en général totalement instanciée et initialisée de manière aléatoire. Elles mu-

nissent l'espace de recherche d'une fonction de coût globale, dérivée (par exemple par sommation) des fonctions associées à chaque contrainte, et que l'on cherche à minimiser. La fonction de coût peut donc être vue comme un critère qui « guide » la configuration courante vers des solutions consistantes avec l'ensemble des contraintes.

Le déplacement dans l'espace de recherche se fait par « exploration du voisinage » de la configuration courante. A chaque itération, on remplace la configuration courante par celle de son voisinage qui minimise la fonction de coût. En pratique, le voisinage est souvent défini comme l'ensemble des configurations dont seule une variable diffère de la configuration courante. Toutefois, l'exploration du voisinage peut rapidement devenir un processus fastidieux dans le cas de problèmes de grande taille, et on a alors généralement recours à une « heuristique de voisinage » pour se limiter à un nombre restreint de configurations.

Ce principe de résolution suppose une gestion efficace des minima locaux qui peuvent très facilement engendrer des « cycles » dans le parcours de l'espace de recherche : si y est la meilleure solution dans le voisinage de x, et x la meilleure solution dans le voisinage de y, alors la recherche va « boucler » sur le cycle $x \rightarrow y \rightarrow x \rightarrow y \rightarrow \ldots$ La méthode la plus utilisée en PPC pour éviter ce phénomène est la recherche tabou.

La recherche tabou, proposée par Glover [GL97], consiste à stocker dans une mémoire à court terme appelée « liste tabou »l'ensemble des configurations visitées. Le choix de la prochaine affectation se fait alors parmi les configurations n'appartenant pas à la liste tabou. Le coût en mémoire pouvant devenir prohibitif, on emploie en général une liste de taille limitée N. La recherche ne pourra alors échapper à un minimum local que si le nombre d'étapes pour obtenir une nouvelle amélioration est inférieur à N. Nous présentons en infra deux métaheuristiques pour les problèmes de satisfaction de contraintes utilisant la recherche tabou, la recherche adaptative et l'heuristique $\mathcal{CN}$-tabu.

6.4.2 Méthodes à population

Les méthodes à population diffèrent de la recherche locale par le maintien et l'actualisation d'un ensemble de configurations plutôt qu'une seule. L'évolution de la population n'est pas gouvernée par l'exploration de voisinages mais par des heuristiques de mouvement. En outre, ces méthodes ne proposent pas de stratégies explicites pour sortir des minima locaux, mais y parviennent implicitement par le recours à une population.

Les algorithmes génétiques (AGs — voir sections 5.5 et 5.6) constituent une large famille de méthodes pour les problèmes de satisfaction de contraintes, à condition le plus souvent d'adapter les opérateurs génétiques au problème à résoudre, sous peine d'inefficacité (voir Hao & al. [HGH98]). Eiben & van der Hauw [EvdH96] ont toutefois proposé un principe générique pour traiter les CSPs à l'aide d'AGs. La méthode SAW (*Stepwise Adaptation of Weights*) permet d'exprimer la fonction de coût comme une somme pondérée des pénalités associées à chaque contrainte. A chaque itération de l'algorithme, les poids relatifs sont réévalués en fonction des taux de satisfaction dans la population courante : aux contraintes les moins satisfaites sont attribués des poids plus élevés. SAW agit donc comme une fonction de coût adaptative qui permet de traiter en priorité les contraintes les plus difficiles. Cette méthode à été expérimentée avec succès sur de nombreux CSPs de référence.

Une autre métaheuristique couramment utilisée pour les CSPs est l'optimisation par colonies de fourmis (*Ant Colony Optimization* — ACO), inspirée du comportement réel des fourmis pour rechercher un chemin optimal entre la fourmillière et une source de nourriture. Les méthodes ACO s'appuient un principe de collaboration « stigmergique » au sein de la

population : la communication entre les agents se fait par l'intermédiaire de l'environnement. On sait en effet d'une part que les fourmis déposent sur leur passage des phéromones qui s'évaporent dans le temps, d'autre part que leurs choix de trajectoires sont déterminés par la quantité de phéromones qu'elles rencontrent. Au cours du temps, les trajets « peu intéressants » sont donc délaissés, tandis qu'« émerge » un unique trajet optimal. En termes d'optimation combinatoire, les méthodes ACO ont donc dans un premier temps été utilisées pour des problèmes de plus court chemin, avant d'être étendues à d'autres classes de problèmes. Quant à la programmation par contraintes, Christine Solnon [Sol02] a récemment proposé le moteur générique ANT-SOLVER qui permet de traiter les CSPs à l'aide de méthodes ACO. L'idée sous-jacente est de ramener la recherche de configurations consistantes à la découverte d'un plus court chemin dans un graphe dont les sommets sont des couples variable/valeur. Lors d'une itération de l'algorithme, chaque fourmi « visite » une instantiation complète du problème et dépose sur les arrêtes reliant les nœuds de la configuration visitée une quantité de phéromones proportionnelle au taux de satisfaction global des contraintes. Lorsque l'exécution se termine, le chemin marqué par le plus grand taux de phéromones permet de relier les valeurs de la meilleure configuration.

Il convient en dernier lieu de mentionner l'usage fréquent dans les recherches récentes de techniques « hybrides » qui combinent plusieurs métaheuristiques ou tirent parti de la complémentarité entre méthodes complètes et incomplètes. L'heuristique $\mathcal{CN}$-tabou de Vasquez & Dupont [VHD03] en est un exemple. Elle sera présentée au paragraphe 6.6.

6.5 Recherche adaptative

La recherche adaptative est une heuristique de recherche locale proposée par Philippe Codognet [CD01] [CDT02]. Elle part du principe que dans une configuration inconsistante, toutes les variables n'ont pas la même « responsabilité » dans la violation des contraintes, et que certaines d'entre elles sont à modifier en priorité afin de rétablir au plus vite la consistance.

Dans la plupart des CSPs, les contraintes possèdent la propriété essentielle de ne porter que sur des variables « fixes » : si une contrainte est violée, on est en général très souvent en mesure d'identifier les variables « responsables » de l'inconsistance. Soit par exemple un problème à quatre variables A, B, C, D, chacune à valeurs dans $\{0, 1, 2\}$, assorti des contraintes $C_1 : A \neq B$ et $C_2 : B < C$. Si C_1 est violée, il faut évidemment modifier les variables A ou B ; de même, si C_2 est violée, agir sur A ou D sera inutile. Allons plus loin : la configuration $(2, 2, 1, 0)$ viole les deux contraintes ; il faut donc au minimum modifier A, B ou C. Or B intervient dans C_1 *et* C_2 ; c'est donc peut-être la variable à changer en priorité. En effet, choisir $B = 0$ permet de vérifier toutes les contraintes en un seul changement de valeur, alors qu'agir sur une autre variable implique toujours au minimum deux mouvements.

La recherche adaptative tire parti d'un constat simple : dans une configuration inconsistante, on peut dans la plupart des cas identifier la variable dont la modification permet d'accroître le plus rapidement le niveau de consistance. En pratique, on commence par transformer les fonctions de coût associées aux contraintes pour obtenir une fonction de coût pour chaque variable. A chaque itération de l'algorithme, si la configuration courante viole au moins une contrainte, c'est la variable la plus « couteuse » qui est modifée. Charlotte Truchet [Tru04] préconise une méthode de « projection » des contraintes sur les variables permettant d'éviter

l'apparition de minima locaux artificiels[4] : le coût associé à une variable est simplement le nombre de contraintes violées dans lequel elle intervient. Ce type de fonction est appelé *min-conflict*. Dans l'exemple précédent, les coûts associés au variables A, B, C, D seraient alors respectivement $1, 2, 1, 0$; c'est donc bien sur B qu'il faut agir en premier.

Si nous rajoutons maintenant la contrainte $C_3 : A+B+C+D \leq 3$ au problème précédent, alors les deux configurations $x = (1, 0, 1, 2)$ et $y = (2, 1, 2, 0)$ vérifient toutes deux C_1 et C_2, mais pas C_3. Quelles sont les variables responsables ? A priori les quatre, car la contrainte C_3 les implique toutes. Comment alors choisir la variable à modifier ? Il est clair qu'une fonction *min-conflict* est ici inefficace. Etant donné le type de contrainte, il ne serait pas idiot d'agir sur les plus grandes valeurs, soit C pour x et A ou B pour y. Il apparaît que dans ce cas, le choix de la variable à changer dépend de la configuration, donc nécessite un calcul qui peut vite devenir très coûteux sur des problèmes de grande taille avec beaucoup de contraintes. En résumé, la recherche adaptative n'est envisageable que pour les contraintes d'arité faible (impliquant un petit nombre de variables) par rapport à la taille du problème. Ce n'est toutefois guère préjudiciable, car dans la plupart des CSPs, les contraintes sont unaires ou binaires, et quand ce n'est pas le cas, on peut souvent les décomposer en un ensemble de contraintes binaires.

L'algorithme de recherche adaptative [CD01] [CDT02] utilise une mémoire à court terme de type « tabou » (voir paragraphe 6.4.1. A chaque itération, le voisinage à explorer est obtenu en remplaçant, dans la configuration courante, la variable V_{max} non marquée « tabou » et de coût maximal, par toutes les valeurs de son domaine. La meilleure configuration remplace alors la configuration courante si elle en améliore le coût global. Sinon, V_{max} est marquée « tabou » pour un nombre fixé d'itérations.

Outre ses performances obtenues sur un grand nombre de problèmes musicaux, la recherche adaptative a prouvé son efficacité sur des CSPs classiques (problème des n-reines, carrés magiques...) en affichant des temps de calcul significativement inférieurs à ceux obtenus par des solveurs implémentant d'autres métaheuristiques. En outre, elle a permis de résoudre des instances jusqu'à 50 fois plus grandes que les limites actuelles des méthodes complètes.

6.6 L'heuristique $\mathcal{CN}$-tabou

L'heuristique $\mathcal{CN}$-tabou [VHD03] (*Consistent Neighborhood Tabu Search*) est une méthode hybride combinant recherche tabou et arc-consistance. L'idée maîtresse est de maintenir systématiquement la consistance au cours de la recherche, quitte à travailler sur des instantiations *partielles* (i.e. dont certaines variables ne sont pas instanciées). Là où les métaheuristques traditionnelles cherchent à satisfaire les contraintes en manipulant des configurations totalement instanciées, la méthode $\mathcal{CN}$-tabou opère sur des configurations partielles et localement consistantes (aucune contrainte n'est violée), et cherche à instancier toutes les variables. Les contraintes sont exclusivement binaires.

En pratique, la recherche est conduite par une succession d'instanciations et de désinstanciations. A chaque itération, un voisinage est construit en affectant à une variable non instanciée et non marquée « tabou » toutes les valeurs de son domaine. Dans chacune des

[4]La plupart du temps, une mesure d'inconsistance pour une configuration est obtenue par sommation des fonctions de coût associées à chaque contrainte. Si les fonctions sont hétérogènes, des minima locaux indésirables risquent d'apparaître, pouvant perturber la résolution. On s'en prémunit habituellement en choisissant pour mesure d'inconsistance le nombre de contraintes violées, ou le nombre de variables intervenant dans les contraintes non satisfaites.

configurations ainsi produites, les éventuelles violations sont alors « résolues » en désinstanciant toutes les variables en conflit avec la nouvelle affectation. Les contraintes étant binaires, un simple filtrage par arc-consistance suffit pour « propager » l'information apportée par nouvelle affectation. Dans le voisinage — localement consistant — ainsi obtenu, la configuration comptant le plus grand nombre de variables instanciées devient alors la configuration courante, et la variable modifiée est marquée « tabou ». Le statut « tabou » est révoqué après un nombre d'itérations égal au nombre de fois où la variable a été modifiée depuis le début de l'exécution. La liste tabou est ici gérée dynamiquement : plus une variable a été modifiée dans le passé, et plus il sera difficile de la changer dans le futur.

L'heuristique $\mathcal{CN}$-tabou a été employée avec succès pour résoudre des problèmes réels d'assignation de fréquence et de positionnement d'antennes, à très grand nombre de variables et de contraintes.

6.7 Contraintes et algorithmes génétiques

Si les algorithmes génétiques ont prouvé leur efficacité face à de nombreux problèmes d'optimisation (y compris multicritère) ou de satisfaction de contraintes, en revanche leur intérêt dans le cas de problèmes mixtes d'optimisation sous contraintes est moins établi. La raison principale est que les opérateurs génétiques sont « aveugles » aux contraintes : les rejetons obtenus par crossover (voir section 5.5.2) à partir de parents vérifiant les contraintes du problème n'offrent a priori aucune garantie de consistance. La même remarque s'applique aux mutations.

Comme nous le verrons aux chapitres 7 et 10, l'optimisation combinatoire multicritère sous contraintes est un cadre commode pour la formalisation du problème de l'orchestration assistée par ordinateur. Cela nous impose donc, si nous faisons le choix d'un AG pour l'optimisation (voir chapitre 9), de décider d'une méthode explicite pour la gestion des contraintes supplémentaires. Eiben [Eib01] recense pour cela deux types d'approches : la gestion *directe* et la gestion *indirecte*.

6.7.1 Gestion directe

La gestion directe des contraintes au sein d'un AG impose qu'à tout moment, la population ne contient que des configurations consistantes avec l'ensemble des contraintes. Ce maintien de la consistance peut s'effectuer de plusieurs manières :

Elimination systématique L'idée la plus simple consiste à retirer de la population tous les individus qui violent au moins une contrainte. Dans la plupart des applications toutefois, la probabilité d'obtenir une configuration consistante par hasard ou par le biais d'opérateurs génétiques est quasiment nulle. Dès lors, l'élimination systématique interdit toute possibilité d'évolution, car cette dernière n'engendre « naturellement » que des individus inconsistants.

Réparation Alternativement, les méthodes dites de « réparation » (*repair procedures*) permettent de rétablir la consistance chaque fois qu'apparait dans la population une configuration violant au moins une contrainte. Dans les problèmes de sac à dos (voir section 9.1), on utilise en général une heuristique de type « glouton » (*greedy repair procedure*) : les items les plus encombrants sont itérativement retirés de la configuration jusqu'à ce que les contraintes de

capacité soient de nouveau satisfaites (voir Ishibuchi & Kaige [IK03]). Dans le cas général, le choix d'une méthode de réparation nécessite une très bonne connaissance du problème, et débouche sur une procédure parfois coûteuse en temps de calcul : la plus grande partie de l'exécution est consacrée au maintien de la consistance plutôt qu'à l'évolution de la population. Dans le cas du sac à dos, les contraintes sont en faible nombre, les algorithmes gloutons de réparation sont de complexité acceptable, en général en $\mathcal{O}(N \log(N))$. Mais la réparation n'est pas toujours à préconiser pour des problèmes fortement contraints et dont la structure ne permet pas d'imaginer de méthodes de faible complexité.

Opérateurs ad-hoc Une troisième solution, fréquente dans la résolution de CSPs au moyen d'AGs, consiste à imaginer des opérateurs génétiques qui garantissent (ou à défaut maximisent) la consistance des configurations qu'ils engendrent (voir par exemple Dorne & Hao [DJ98]). C'est dans cette direction que s'est orientée notre première version d'un algorithme génétique pour l'aide à l'orchestration [CTA+07], dans lequel l'opérateur de mutation agissait tantôt comme *déplacement horizontal* (la hauteur de note est conservée), tantôt comme *déplacement vertical* (l'instrument est conservé). Bien que donnant de bons résultats sur des problèmes faiblement contraints, ces opérateurs se sont vite révélés inefficaces face à un réseau de contraintes plus dense.

Décodage Enfin, le « décodage » permet de résoudre le problème dans un espace de recherche E' différent de E et libre de toute contrainte. On a alors recours à une procédure de « décodage » (*decoding procedure*) pour transformer les configurations de E' en configurations consistantes de E. Ce principe a notamment été expérimenté par Michalewicz [Mic96].

Eiben [Eib01] fait observer que si la gestion directe des contraintes peut s'avérer extrêmement efficace si elle est bien implémentée, elle est en revanche fortement dépendante du problème. Elle est donc à recommander pour des applications très ciblées, dans lesquelles la structure des problèmes rencontrés ne change pas ou peu.

6.7.2 Gestion indirecte

A l'opposé, la gestion indirecte permet souvent d'aborder une plus grande variété de problèmes. Une de ses différences majeures avec la gestion directe est d'autoriser la présence de configurations inconsistantes dans la population. Ces dernières étant en effet souvent partiellement consistantes (seules certaines contraintes sont satisfaites), elles recèlent une part d'information importante pour le « guidage » de la population à la fois vers les zones d'optimalité et de consistance.

Transformation en objectifs La méthode la plus simple consiste à transformer chaque contrainte en un critère supplémentaire à optimiser, à travers par exemple une fonction de coût. Seul problème, un grand nombre de contraintes implique une augmentation importante du nombre de critères, donc une complexité plus grande et une plus grande probabilité d'efficience pour une configuration aléatoire[5] : le risque est alors d'obtenir un ensemble de solutions de très grande taille, au sein duquel il sera difficile de repérer les configurations vraiment pertinentes.

[5]Dans le cas général, il est évident que plus on augmente le nombre de critères, plus une configuration aléatoire à de chance d'être efficiente : il y aura toujours un critère sur lequel elle sera meilleure que les autres. Dans notre cas, une configuration est efficiente sitôt qu'elle satisfait au moins une contrainte.

On peut toutefois limiter l'augmentation du nombre de critères en agrégeant l'ensemble des fonctions de coût associées aux contraintes en une valeur unique, mais la perte d'information qui en résulte peut rendre le problème difficile à résoudre.

Pénalisation de la fitness La pénalisation de fitness est l'approche la plus courante. Une mesure de consistance[6] est ajoutée à l'évaluation de fitness pour pénaliser les configurations non consistantes. La population est alors évaluée à l'aide d'une fitness « augmentée » :

$$g(x) = f(x) + \mu z(x)$$

où f est la fitness d'optimisation et $z(x) = 0$ si est seulement si x satisfait l'ensemble des contraintes. Seul problème, le paramètre μ est extrêmement délicat à fixer, et cependant déterminant dans l'efficacité de l'algorithme : si la valeur de μ est petite, l'évolution n'est pas encouragée vers les configurations consistantes ; si μ est trop grand, l'algorithme ne « voit » plus que les contraintes et se détourne de l'optimisation. En outre, Deb [Deb00] fait observer que la pénalisation introduit une distorsion dans le paysage de fitness, susceptible d'engendrer de nouveaux optima locaux.

Incorporation dans la relation de dominance Deb [Deb00] a proposé une méthode générique pour la gestion des contraintes dans les AGs, incorporant la notion d'inconsistance au sein même de la relation de dominance. L'auteur recommande l'utilisation d'un tournoi binaire dans le mécanisme de sélection (voir sections 5.5 et 5.6), au cours duquel deux individus sont comparés de la manière suivante :

- Une configuration consistante est toujours préférée à une configuration violant au moins une contrainte.
- Entre deux configurations consistantes, celle de meilleure fitness est préférée.
- Entre deux configurations inconsistantes, celle qui viole le moins les contraintes est préférée.

Dans chacun de ces trois scénarii, les solutions ne sont jamais comparées simultanément sur leur fitness ou sur leur consistance. A l'inverse de la méthode précédente, il n'y a donc pas de paramètre supplémentaire à fixer. En outre, aucune hypothèse n'est formulée quant à la nature du problème. Nous pouvons dès lors étendre la définition de la dominance de Pareto à tout problème d'optimisation sous contraintes :

Définition 6.1. *Etant donné un problème de minimisation multicritère sous contraintes, soient x et y deux points de l'espace de recherche E et $z : E \to \mathbb{R}$ une mesure d'inconsistance des configurations ($z(x) = 0$ si est seulement si x satisfait l'ensemble des contraintes). On dira alors que « x domine y au sens de Deb » si et seulement si :*

$$z(x) < z(y) \ \vee \ (z(x) = z(y) \wedge x \prec y)$$

[6] Le plus souvent, elle est définie comme le nombre de contraintes violées, ou le nombre de variables impliquées dans l'ensemble des violations.

Conclusion

La programmation par contraintes permet de définir un problème par un ensemble de conditions que doivent satisfaire ses variables. Selon la complexité des situations, les métaheuristiques peuvent s'avérer nettement plus efficaces que les méthodes complètes. La recherche locale — notamment la recherche adaptative — a prouvé son efficacité dans bien des cas et permet souvent de résoudre en un temps acceptable des instances de très grande taille. En revanche, la gestion des contraintes dans un problème d'optimisation pose problème à bien des égards, en raison notamment de l'hétérogénéité des formalisations. En outre, la résolution de problèmes d'optimisation par algorithmes génétiques se prête mal aux problèmes contraints, car les opérateurs génétiques engendrent très facilement des configurations inconsistantes. Une solution élégante consiste alors à introduire la consistance dans la relation de dominance entre les configurations.

Résumé de la première partie

Aujourd'hui, la richesse et l'extension permanente du vocabulaire instrumental ne permettent plus aux compositeurs contemporains de maîtriser l'ensemble des possibilités timbrales offertes par l'orchestre. L'art de l'orchestration, dont l'empirisme gouverne la pratique comme la transmission, souffre cruellement de l'absence d'un traité contemporain qui tienne compte de l'héritage du XX^{e} siècle. Nous pensons que l'ordinateur, par sa puissance de calcul et sa capacité à explorer rapidement un grand nombre de combinaisons instrumentales, peut apporter un éclairage nouveau sur cette pratique, et assister les compositeurs dans l'exploration et le contrôle du timbre orchestral.

La conception d'un système d'aide à l'orchestration se situe à la convergence de plusieurs domaines de recherche et impose que soient maîtrisées un éventail de techniques adaptées à la complexité de cette discipline.

Du côté du timbre, les résultats de la psychoacoustique en perception auditive et leur mise en relation avec les techniques de description du signal audio permettent de modéliser le potentiel sonore des instruments à partir d'échantillons enregistrés.

Du côté de l'optimisation, les méthodes multicritères offrent la possibilité de rechercher des solutions selon plusieurs objectifs, sans faire d'hypothèses de priorité ou de hiérarchie sur les critères. C'est donc un cadre formel opportun pour aborder le problème du timbre, dont la multidimensionnalité est une des causes de la complexité.

Le caractère $\mathcal{NP}$ des problèmes d'optimisation combinatoire interdisant le recours à des méthodes complètes, la recherche opérationnelle s'est depuis une vingtaine d'années tournée vers les métaheuristiques. Ces dernières fournissent un ensemble de techniques génériques pour aborder des problèmes complexes, et permettent de trouver en un temps raisonnable des solutions approchées.

Enfin, la programmation par contraintes est un paradigme idéal pour les problèmes dont les solutions sont définies par un ensemble de relations logiques entre leurs variables, comme il en va de nombreuses situations musicales. Là encore, les métaheuristiques sont d'un grand secours dans le cas de problèmes difficiles.

Deuxième partie

Algorithme de recherche d'orchestrations

Chapitre 7

Formulation du problème

Timbre ciblé et connaissance instrumentale
Concepts fondateurs pour un outil d'aide à l'orchestration
Formalisation comme recherche multicritère sous contraintes
Entre écriture et optimisation : les descripteurs du signal

Nous abordons dans cette thèse le problème de l'orchestration assitée par ordinateur d'un point de vue algorithmique. Nous nous ramenons à un problème combinatoire dans lequel il s'agit de trouver une combinaison de sons instrumentaux qui, joués simultanément, produisent le timbre désiré par le compositeur. Des questions préliminaires surgissent aussitôt : Comment caractériser ce timbre ciblé ? Comment représenter la connaissance instrumentale dans un outil informatique ? Comment évaluer la qualité d'un mélange (i.e. sa "distance" à la cible) ? Sur quels critères se baser ? Ce chapitre aborde ces questions. Nous y exposons nos propres conceptions et les choix méthodologiques qui en découlent.

7.1 Données du système

Un outil d'orchestration doit permettre d'assister le compositeur dans le contrôle du timbre orchestral. A partir de la formalisation d'une « idée de timbre » le système doit pouvoir proposer un « mélange instrumental » qui corresponde à l'intention (ou l'intuition) du compositeur. Cela suppose deux conditions préalables :

1. Un vocabulaire précis doit être défini pour permettre au compositeur de formaliser de manière objective son intention musicale.
2. Le système doit connaître les possibilités sonores des intruments.

7.1.1 Le concept de « cible »

D'un point de vue scientifique, l'intention du compositeur doit être pensée comme une « cible », un objectif à atteindre par une méthode appropriée. Si l'objectif ne peut être atteint, il faut alors trouver la solution la plus proche. La notion de « distance » est donc primordiale, et notre travail doit consister dans un premier temps à définir cette distance, dans un second temps à imaginer une méthode appropriée pour la minimiser. On pourrait bien sûr décider de se passer de la cible, et prendre le parti d'explorer l'espace des timbres de manière itérative. L'orchestre serait alors un simple réservoir de sons dans lequel on puiserait pour « construire »

un timbre, à travers un processus interactif d'essais/erreurs. Bien sûr, cette démarche est tout à fait légitime d'un point de vue esthétique, mais elle n'a *rien* de scientifique.

Il nous faut donc définir une cible. Dans notre cas, elle serait un ensemble de caractéristiques décrivant les particularités du timbre souhaité. Or, caractériser le timbre nécessite une collection de termes établis et univoques — chacun d'entre eux se référant à une caractéristique précise du son — partagée par les compositeurs, les chercheurs et les instrumentistes. Hélas, la construction d'un tel vocabulaire est illusoire, car la manière dont la perception auditive est verbalisée implique toujours une grande part de subjectivité. Certains champs lexicaux conviennent certes davantage que d'autres. Erickson [Eri75] fait ainsi observer que « des termes du vocabulaire visuel ou tactile sont souvent appropriés pour décrire les sons ou les combinaisons de sons : tranchant, rugueux, sourd, doux, mordant, clair, éclatant, cassant, épais, fin, sec, vaporeux, aéré, minutieux, flasque, fluide, translucide, flamboyant, granuleux, cru, brumeux, lourd, glacial, ébréché, limpide, luxuriant, léger, ondulé, pour en citer quelques uns. » Mais qu'est-ce, en définitive, qu'un son fluide ? Qu'est-ce qu'un son glacial ? Interrogeons des compositeurs, ils donneront (s'ils le peuvent) des exemples différents, en rapport avec leur expérience musicale personnelle. Alors que tout le monde s'entend sur ce qu'est un accord majeur, ou un accelerando.

A la rigueur, ces termes peuvent être utilisés dans la comparaison des timbres ; on dira par exemple que tel son est plus clair que tel autre. On pourra alors tenter de relier le concept de clarté à un paramètre acoustique mesuré ou calculé sur le signal. Cette démarche est celle des psychoacousticiens qui cherchent à comprendre les mécanismes de différentiation et de catégorisation dans la perception auditive (cf. 2.5). Elle n'est d'aucun secours lorsqu'il s'agit de définir positivement les qualités perceptives d'un son.

On comprend alors pourquoi nos prédécesseurs ont toujours contourné le problème de la caractérisation du timbre souhaité par le compositeur en restreignant la cible à un son enregistré. Après tout, la meilleure façon de décrire un son est encore de le donner à entendre[1].

Nous ne prétendrons pas ici résoudre le problème de la verbalisation du timbre, d'autant moins que les problématiques psychoacoustiques ne sont que périphériques à notre sujet de recherche. Nous adoptons donc dans un premier temps l'approche de nos aînés, et supposons que le compositeur dispose du son enregistré qu'il désire reproduire avec l'orchestre. Après tout, « proposer un arrangement instrumental qui se rapproche le plus possible de la source d'origine » faisait partie de requêtes des compositeurs exposées au paragraphe 4.2. En outre, les questions sous-jacentes à cette idée sont déjà de taille : Comment analyser le son cible ? Comment évaluer la similarité entre la cible et une proposition d'orchestration ? Comment trouver l'orchestration optimale ?

Toutefois, la question d'un timbre que le compositeur aurait « seulement en tête » ne saurait être prématurément écartée. Il nous faut ici introduire un distingo entre ce que nous nommerons désormais l'orchestration imitative, à savoir la reproduction à l'orchestre d'un timbre défini par un son enregistré et l'orchestration générative, à savoir... tout le reste. Plus précisément, il s'agirait de trouver un moyen de contrôler en entrée notre outil d'orchestration avec une autre matière qu'un fichier son. L'idée est donc la suivante : puisqu'il semble illusoire, pour l'instant, de se passer de ce son, ne peut-on trouver un moyen de le générer à partir de données symboliques, via un processus de synthèse par exemple ? La figure 7.1 illustre ce principe. L'idée maîtresse est que le compositeur en passe toujours par le sonore

[1] Ce n'est pas nécessairement le cas de tous les éléments du langage musical. Il est sans doute plus facile de décrire un canon en expliquant le principe des entrées successives des voix que d'en faire écouter un exemple.

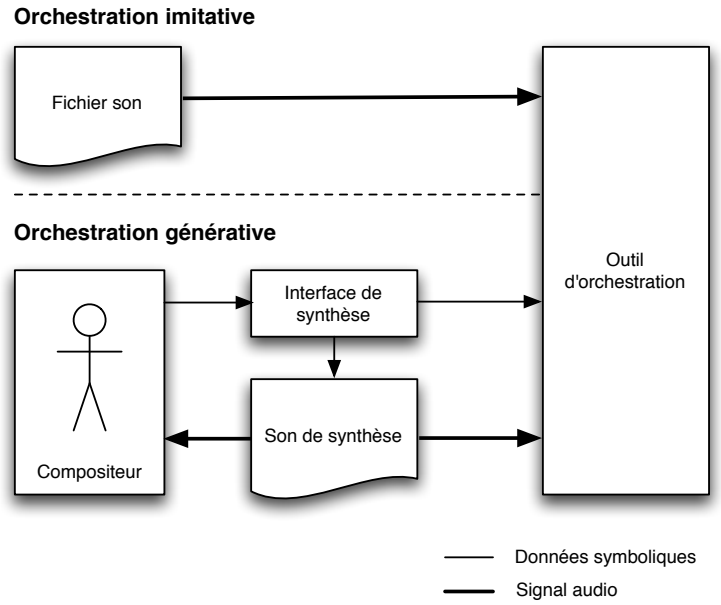

FIG. 7.1 – Données d'entrée en fonction du type d'orchestration (imitative ou générative)

pour communiquer avec l'ordinateur. Lorsque le timbre visé n'est pas un son enregistré, le compositeur dispose d'une interface de synthèse sonore pour le « construire » itérativement, à travers un processus d'essais/erreurs. Bien sûr, il y a fort à parier que le son ainsi créé soit très pauvre d'un point de vue perceptif, mais son rôle est uniquement de « capturer » l'ensemble des caractéristiques sonores souhaitées par le compositeur. Il n'est qu'un modèle, un « portrait-robot » dont l'orchestration finale proposée par le système possèdera les traits tout en y ajoutant la richesse du timbre instrumental. Comme l'illustre la figure 7.1, des données symboliques peuvent aussi être passées au système dans le cas d'une orchestration générative. Nous donnerons davantage de détails à ce sujet au chapitre 12.

Cette piste à été explorée de manière prospective en parallèle de nos travaux, en collaboration avec Jean Bresson (Chercheur à l'IRCAM, chargé du développement d'OPENMUSIC [Ago98]). Nous verrons au chapitre 14 un exemple d'orchestration utilisant un son de synthèse comme intermédiaire entre un accord écrit et un timbre. Nous exposerons également au chapitre 12 un ensemble d'interfaces et de méthodes permettant de contrôler le timbre de ce son de synthèse à partir de données symboliques.

7.1.2 Connaissance instrumentale

De même qu'il semble difficile de se passer du son dans la caractérisation d'un timbre à orchestrer, de même la modélisation des possibilités instrumentales en termes de timbre n'est envisageable que par l'analyse d'enregistrements de sons instrumentaux. Ces dernières années, le développement massif d'outils de production musicale et l'engouement des utilisateurs pour les systèmes basés sur des techniques d'échantillonnage (les « samplers ») ont grandement favorisé la mise à disposition de banques de sons instrumentaux. Outre les produits commerciaux (*Vienna Symphonic Library*, *Native Instruments*, *Plugsound*, *Virtual Orchestra*...)

on trouve aujourd'hui de nombreuses banques d'échantillons créées à des fins de recherche, telles que *RWC Music Database* [GHNO03] [Got04], *University of Iowa Musical Instrument Samples*, *McGill University Master Samples* (Opolko & Wapnick, 1987) ou encore SOL (*Studio Online* [BBHL99], enregistrée à l'IRCAM.

Si ces bases de données sont loin d'être exhaustives, elle sont souvent complémentaires. Par exemple VSL (*Vienna Symphonic Library* [VSL]) contient tout les instruments de l'orchestre symphonique classique, en solo ou par ensembles[2], mais uniquement pour des modes de jeu classiques. A l'opposé, SOL inclut des instruments utilisés en musique contemporaine (comme l'accordéon, la guitare ou les saxophones), comprend les modes de jeu contemporains mais ne contient que des instruments solo.

On serait donc facilement tenté d'agréger plusieurs bases de données pour obtenir une palette sonore la plus large possible. Or, les conditions d'enregistrement et de post-production différant selon les banques d'échantillons, cette base hybride serait inexploitable pour des raisons d'hétérogénéité.

Le problème des dynamiques

Le plus gros problème inhérent aux bases de sons instrumentaux et principal responsable de leur incompatibilité est sans doute celui des dynamiques. La dynamique est une donnée symbolique inscrite sur la partition qui contrôle l'intensité du son. Dans les banques de sons, l'information de dynamique est en général incluse dans le nom du fichier. Un mezzo-forte de violon et un mezzo-forte de clarinette provenant de deux bases de données différentes doivent donc être perçus avec environ la même intensité. En pratique, c'est loin d'être le cas. Pour preuve, les échantillons de VSL sont normalisés, laissant à l'échantillonneur le soin d'ajuster l'intensité en fonction du contexte. Ceux de SOL ne le sont pas. L'utilisation croisée des deux bases est donc impossible.

A des fins de robustesse, il arrive parfois qu'on mélange des bases de données pour, par exemple, évaluer des algorithmes de reconnaissance d'instruments. Les incohérences de dynamique ne posent alors aucun problème, car les sons n'ont pas à être comparés entre eux, mais à un ensemble de modèles qui permettent une certaine souplesse dans le traitement de la dynamique. Les psychoacousticiens mélangent aussi des bases de sons pour mettre en place des tests peceptifs. En général, ces tests nécessitent un petit nombre de sons et on prend soin au préalable de les égaliser en sonie[3], ce qu'on ne pourrait faire dans notre cas qu'au sein d'un même groupe de dynamiques, et au prix d'un travail colossal.

Nous allons donc devoir nous restreindre à l'utilisation d'une unique banque de sons. Notre choix se portera sur SOL [BBHL99], pour les raisons suivantes :

1. La base SOL a été enregistrée à l'IRCAM et nous en disposons gratuitement.
2. Elle embrasse un vocabulaire sonore riche et représentatif des pratiques instrumentales actuelles ; de nombreux modes de jeu contemporains y sont disponibles.
3. Tous les échantillons de SOL on été enregistrés dans des conditions identiques, avec une prise de son indépendante de l'instrument. Ils sont « calibrés » de manière à restituer la perception d'un auditeur à une position fixe par rapport aux instrumentistes. On peut donc considérer que la base est cohérente par rapport aux dynamiques.

[2]Les sons d'ensemble sont des unissons d'instruments du même type : violons par 12, violoncelles par 8, cors par 4...

[3]Deux sons de même sonie sont perçus avec la même intensité. L'égalisation ne peut se faire que par l'écoute, car la sonie dépend non seulement de la puissance acoustique, mais également du timbre.

N.B. Si la cohérence des dynamiques est essentielle pour la capacité du système à évaluer la pertinence d'un mélange, elle n'empêche pas pour autant certaines surprises dans les résultats obtenus. Les musiciens interprètent la dynamique en fonction de plusieurs paramètres : hauteur, contexte musical, situation dans l'orchestre, dynamique de l'orchestre... Une même dynamique est jouée differemment en solo, en ensemble ou en grand orchestre. Un forte est joué différemment s'il est précédé d'un pianissimo ou d'un triple forte. Bref, les musiciens passent leur temps à équilibrer les dynamiques lorsqu'ils jouent en ensemble, afin de rendre au mieux les effets d'orchestre voulus par le compositeur. On ne sera donc pas surpris, parfois, d'entendre des simulations d'orchestration dont les dynamiques paraissent déséquilibrées, ou qui traduisent de façon peu réaliste la partition. Ce sera au compositeur, par son expérience, d'ajuster les dynamiques pour que la simulation d'un timbre soit fidèle à la partition que l'outil d'orchestration propose.

Approche par modèles d'instruments

Même avec une base cohérente en dynamiques telle que SOL, les problèmes de généricité et de robustesse de la connaissance instrumentale évoqués en 4.3 restent en suspens. Notre système d'orchestration ne connaît des potentialités du basson que ce que le bassonniste enregistré en a donné à entendre dans un studio de l'IRCAM. Les orchestrations proposées seront-elles valables pour un autre bassoniste dans une salle de concert ?

Nous n'avons malheureusement pas d'autre choix, dans un premier temps, que de s'en remettre à cette hypothèse. Cela dit, les travaux de Damien Tardieu — doctorant-chercheur dans l'équipe Analyse-Synthèse de l'IRCAM, avec qui nous collaborons sur le sujet de l'orchestration — permettent de dépasser cette limitation en remplaçant la base de données de sons instrumentaux par un ensemble de modèles d'instruments. Il s'agit de modèles probabilistes agrégeant la connaissance issues de plusieurs bases de données sonores à l'aide de mélanges de gaussiennes (GMM[4]). En utilisant plusieurs échantillons d'un même son, par exemple un La5 mezzo-forte de hautbois, on construit un modèle générique du La5 mezzo-forte pour le hautbois, plus robuste aux changements de conditions de jeu qu'un unique échantillon. Evidemment, un travail préalable sur la gestion des problèmes de dynamique a été nécessaire. Damien Tardieu a en outre mis en place une hiérarchie de modèles permettant de généraliser aisément la connaissance instrumentale et de construire des modèles pour des sons absents des bases de données. Par exemple, le modèle de l'alto vibré se décompose facilement en deux sous-modèles : l'alto non vibré et le vibrato de cordes. Si les bases de données utilisées pour l'apprentissage ne contiennent aucun son d'alto vibré, on peut quand même en construire un modèle à base de l'alto non vibré et du vibrato de cordes appris sur le violon. Le lecteur désirant davantage de précisions sur les modèles d'instruments pourra consulter [TR07].

7.2 Paradigmes

La conduite de nos travaux et le développement de notre outil d'aide à l'orchestration ont été conditionnés par un ensemble de paradigmes que semblaient imposer non seulement l'état de maturité actuelle des savoirs connexes à notre problème (voir 4.1), mais également un ensemble de scénarii d'utilisation de notre outil, imaginés lors du dialogue avec les compositeurs.

[4] *Gaussian Mixture Models.*

7.2.1 « Privilégier le son »

Etant donné la difficulté de définir un vocabulaire pour qualifier le timbre (voir section 7.1.1), nous pensons qu'il faut privilégier le son comme moyen de communication entre le compositeur l'outil d'orchestration, et ce à toutes les étapes du processus. C'est par l'intermédiaire du sonore que le compositeur doit être capable aussi bien de définir le timbre qu'il désire que d'orienter la recherche dans une direction particulière.

7.2.2 Approche descriptive

Nous avons vu au chapitre 2 que les recherches récentes en perception auditive, en psychoacoustique et en description du signal audio ont permis d'établir une correspondance entre la caractérisation du timbre et l'extraction automatique de descripteurs du signal. Cette approche qui consiste à décomposer le son en plusieurs dimensions significatives est d'un grand intérêt pour l'orchestration assistée par ordinateur. La question est ici de savoir si nous pouvons définir un espace muni d'une distance pertinente entre une cible sonore et des propositions d'orchestration.

Si descripteurs et attributs perceptifs peuvent être reliés, c'est qu'il existe une correspondance entre espaces de timbres et espaces de descripteurs. De même qu'il est donc possible de partir de jugements de dissimilarité pour rechercher les dimensions principales de la perception du timbre, ne peut-on pas, en plaçant des sons dans un espace de descripteurs, en déduire la distance perceptive qui les sépare ? Nous faisons cette hypothèse. Elle nous permet entre autre de définir un peu plus précisément la notion de « cible sonore » introduite au paragraphe 7.1.1 : si la cible reste pour l'utilisateur un son enregistré (orchestration imitative) ou généré par un processus de synthèse (orchestration générative), pour le système elle n'est en revanche rien de plus qu'un ensemble de descripteurs calculés sur ce son. La cible est un point dans l'espace des descripteurs, et les propositions d'orchestration en sont d'autant plus proches perceptivement que les distances entre leurs valeurs respectives de descripteurs sont moindres.

Gardons toutefois à l'esprit que les espaces de timbres sont construits sur des jugements perceptifs de dissimilarité de sons instrumentaux. Les dimensions qui y sont identifiées sont donc celles qui interviennent dans des mécanismes de différenciation ou de catégorisation, nous permettant d'identifier un piano, ou de distinguer une trompette d'un hautbois. En outre, tous les sons utilisés dans l'expérience de McAdams et al. [MWD+95] étaient égalisés en sonie, hauteur et durée perceptive, afin d'éviter toute interférence de ces paramètres dans les jugements de similiraté. Bien que le corpus utilisé contienne des timbres hybrides, il n'est donc pas certain que les dimensions identifiées par cette étude soient toujours valables pour l'orchestration, les compositeurs manipulant des mélanges instrumentaux complexes, composés de hauteurs et d'intensités différentes...

La caractérisation du timbre dans le cadre de l'orchestration assistée par ordinateur présente, par rapport à des tâches de ségrégation ou de classification instrumentale, au moins trois difficultés supplémentaires :

1. Les sons que nous décrivons sont aussi bien des sons instrumentaux individuels que des sons complexes, que ce soient les cibles sonores ou les mixtures instrumentales qui les approchent. Les descripteurs du timbre sont-ils toujours valables pour ces sons ?
2. Comment, à partir des positions individuelles de chaque composante d'un mélange dans un espace de timbres, déduire les coordonnées de la mixture dans le même espace ?

3. Quelle distance utiliser, au sein de cet espace, pour la comparaison des timbres ? La distance euclidienne est-elle la plus appropriée ? Les dimensions de l'espace de timbres ont-elles toutes la même importance dans le calcul de la distance ? Si non, comment déterminer les poids relatifs à chaque dimension ? Ces poids sont-ils invariants d'une comparaison à l'autre ?

En ce qui concerne le premier point, nous ferons l'hypothèse que nos descripteurs valent aussi bien pour les sons instrumentaux individuels que les cibles ou les mixtures. Cette hypothèse sera confirmée par notre expérience avec l'outil d'orchestration, lors du parcours, le long d'un descripteur, des solutions trouvées (voir chapitre 13). Le choix des descripteurs va dépendre conjointement des restrictions que nous nous imposons dans un premier temps de notre recherche (voir section 8.1.1), des hypothèses et paradigmes relatifs à la conception de notre outil, et en majeure partie des propriétés « additives » de ces descripteurs. Concernant ce dernier point, la question fondamentale est de savoir si, étant donnée une combinaison de sons de la base de données, on peut en estimer les descripteurs à partir des descripteurs individuels de chaque son. Elle renvoie à la deuxième des difficultés évoquées ci-dessus. Quant à la troisième, se pose d'abord le problème de définir une distance entre une mixture et la cible pour chaque descripteur, ensuite seulement de savoir s'il est possible de rapporter ce vecteur de distance à une valeur unique. Nous verrons en infra qu'une telle agrégation n'est pas possible a priori, sans connaissances supplémentaires sur les préférences d'écoute du compositeur.

Plutôt que de travailler directement sur les échantillons représentant la connaissance instrumentale, nous privilégions donc une vision descriptive du timbre instrumental. Outre les gains considérables en temps de calcul qui en résultent[5], cette approche nous permet à la fois de conserver un certain « recul » par rapport aux échantillons[6] et de définir des distances selon des directions indépendantes du timbre. Il devient alors possible de s'« approcher » de la cible en privilégiant une dimension ou une combinaison de dimensions : nous savons en effet que toutes ne sont pas équivalentes dans la perception du timbre et que les critères perceptifs utilisés dans les jugements de similarités perceptives dépendent à la fois de l'auditeur et des sons comparés (voir paragraphe 2.5). Nous exposerons au paragraphe 8.1.2 les dimensions du timbre que nous choisirons de retenir pour notre propre problème.

7.2.3 Accessibilité de la description

Nous considérons que la description du son doit être aussi simple que possible et accessible au compositeur non-scientifique. En d'autres termes, les dimensions du timbre doivent toujours être reliées à des paramètres perceptifs, afin de faciliter une manipulation intuitive de l'outil d'orchestration.

[5]Nous proposerons au chaptre 9 une méthode efficace pour la découverte en un temps raisonnable d'orchestrations pertinentes pour un problème donné. Au cours d'une exécution, cet algorithme évalue plusieurs milliers de configurations intermédiaires. Or aujourd'hui le temps d'extraction des descripteurs est du même ordre que la durée du signal. Sans compter le temps nécessaire à l'addition des signaux pour générer les mixtures, une simple exécution de l'algorithme d'orchestration nécessiterait donc plusieurs heures.

[6]Une même valeur de descripteur pouvant être partagée par plusieurs sons différents, le description se situe toujours à un niveau d'abstraction plus élevé que le signal. Il suffit pour s'en convaincre de considérer le descripteur de fréquence fondamentale perçue (pour les sons monophoniques).

7.2.4 Approche multicritère

Nous n'oublions pas que le timbre est un phénomène multidimensionnel, et que la perception organise ces dimensions en fonction du contexte sonore. Nous pensons qu'il est impossible de décider à priori d'une hiérarchie de ces dimensions ou de leur importance relative. Cadoz et McAdams rappellent qu'il faut comprendre comment l'oreille organise le sonore, déterminer quels sont, parmi les diverses composantes du son, les éléments porteurs de forme (voir 2.5). L'identification des dimensions privilégiées de l'écoute est certes une étape fondamentale, mais les proportions relatives dans lesquelles ces dimensions participent aux jugements de similarité ne sont pas décidables a priori. C'est par exemple l'harmonicité du son qui permettra de diffencier un trémolo de caisse claire d'un trémolo de violon, de même que le temps d'attaque sera déterminant pour discriminer un son de piano et un son de hautbois de hauteurs identiques. Dans ces cas triviaux, la dimension privilégiée de l'écoute est facile à prévoir, mais lorsqu'un compositeur cherche à produire un timbre avec l'orchestre, peut-il dire a priori quelle direction (ou combinaison de directions) guide son écoute ? Nous ne le pensons pas. Il nous faudra donc opter pour une approche multicritère, dans laquelle les différentes dimensions du timbre sont optimisées conjointement, sans que l'une d'entre elles ne soit jamais privilégiée.

7.2.5 Agnostisme

La connaissance de notre système se limite à la description de la base de sons instrumentaux. Nous choisissons de ne pas formaliser les éventuelles « règles » d'orchestration que l'on pourrait trouver dans les traités, et laissons le système découvrir par lui-même les bonnes solutions. Notre outil d'orchestration n'a de donc connaissances qu'en instrumentation. Cela doit lui suffire puisqu'il peut, en théorie, « imaginer » n'importe quelle mixture instrumentale.

7.2.6 Interaction et préférences de l'utilisateur

Enfin, nous ne perdons pas de vue que notre système est avant tout un outil de création musicale, et qu'il ne peut être utilisé comme tel qu'à travers un haut niveau d'interaction avec le compositeur. Au cours de ce « dialogue », le compositeur doit pouvoir émettre des choix qui vont conditionner le comportement du système, et le guider vers des solutions intéressantes.

Considérons la figure 7.2 comme le front de Pareto d'un problème d'orchestration dans lequel n'interviennent que deux critères perceptifs. Lorsque l'algorithme de recherche se termine, les configurations efficientes sont retournées au compositeur. Si ce dernier émet une préférence pour la solution 1 par exemple, alors le critère A est prédominant pour ce problème : il représente une dimension privilégiée de l'écoute. A l'opposé, si la solution 3 est préférée, c'est le critère B qui prime. Quant au choix de la solution 2, il indique que les deux critères sont mobilisés à plus ou moins égale importance.

Cette remarque est pour nous fondamentale. Elle nous montre de quelle manière on peut identifier les préférences d'écoute du compositeur, et révéler a posteriori une hiérarchie de critères sousjacente à une situation d'écoute donnée. Dans un scénario d'utilisation interactive où le compositeur est régulièrement amené à faire des choix, ce mécanisme d'inférence des préférences implicites peut nous aider à privilégier certaines directions de l'espace de recherche. Le compositeur peut ainsi espérer rapidement découvrir, à travers un processus d'exploration interactif et supervisé, une proposition d'orchestration qui réponde à ses exigences. Nous pensons ainsi qu'une approche multicritère couplée à un système interactif permet donc de

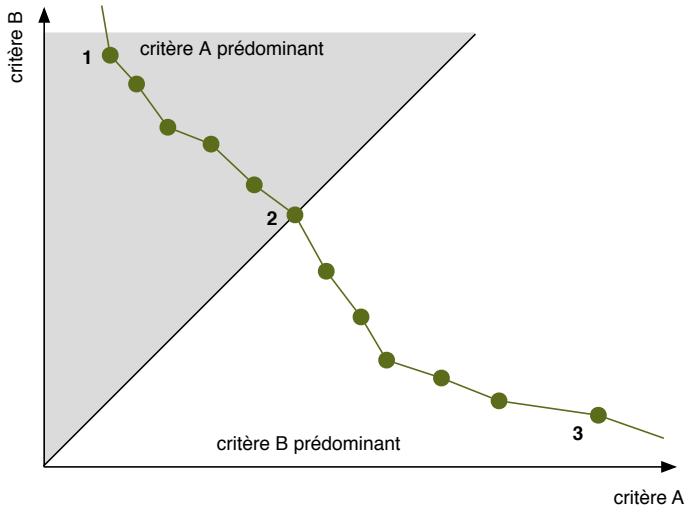

FIG. 7.2 – Zones de prédominance dans un espace à deux critères

construire un système expert guidant le compositeur vers des solutions pertinentes, et ce sans avoir à en passer par une caractérisation verbale du timbre.

Nous proposerons au chapitre 8 une validation de cette approche sur une description « réduite » du timbre instrumental, et présenterons au chapitre 9 un algorithme efficace pour la découverte de solutions efficientes.

7.3 Définitions

Nous rappelons ou introduisons dans cette section les définitions des concepts clés de notre approche, articulée autour des notions de « cible sonore » et de « descripteur ».

Définition 7.1. *Un **descripteur** d est une valeur numérique scalaire ou vectorielle décrivant un aspect du timbre perceptivement identifiable.*

Définition 7.2. *Une **méthode d'extraction** associée à un descripteur d est une méthode de calcul permettant d'obtenir la valeur $d(S)$ de ce descripteur à partir du signal audio S.*

Définition 7.3. *Une **cible** T est un ensemble de descripteurs audio $\{d_1(T), \ldots, d_N(T)\}$ caractérisant un timbre recherché, obtenus par l'analyse d'un son soit enregistré (orchestration imitative), soit de synthèse (orchestration générative).*

Définition 7.4. *Une **configuration** est un sous-ensemble de sons de la banque d'échantillons représentant la connaissance instrumentale, constituant une proposition d'orchestration.*

Nous définissons alors les concepts de fonction d'agrégation et fonction de comparaison permettant de passer en deux étapes d'une configuration à un ensemble de distances à la cible selon chaque descripteur. La figure 7.3 illustre cette transformation.

Définition 7.5. *Etant donné une configuration, une* ***fonction d'agrégation*** f_d *est une fonction n-aire associée à un descripteur d permettant d'estimer la valeur de d pour la configuration en fonction des valeurs individuelles prises par d pour chacune de ces composantes.*

Définition 7.6. *Etant donné une cible à orchestrer T, une configuration, un descripteur d et une valeur de d retournée par la fonction d'agrégation f_d associée à d pour cette configuration, la* ***fonction de comparaison*** D_d^T *associée à d permet de calculer la distance entre la configuration et la cible T selon ce descripteur d.*

Les fonctions d'extraction, d'agrégation et de comparaisons relatives à chacun des descripteurs introduits en 8.1.2 seront explicitées en annexe A et formalisées au paragraphe 7.4. une évaluation de la qualité de ces fonctions est proposée au chapitre 8.

7.4 Formalismes

Nous désignons respectivement par Ω l'ensemble des sons possibles (qu'ils soient enregistrés, produits par un orchestre, ou le résultat d'une synthèse), $(s_i)_{1\leq i\leq N_s}$ la partie de Ω constituant la connaissance instrumentale de notre outil d'orchestration et $\oplus$ l'opération interne sur Ω qui à un ensemble de sons de Ω associe leur mélange[7].

Formulation 7.1. *Le problème de l'orchestration consiste alors, pour une cible $T \in \Omega$, à trouver le sous-ensemble $I \subset \{1, \ldots, N_s\}$ tel que*[8] *:*

$$\bigoplus_{i\in I} s_i \equiv T \tag{7.1}$$

où $\equiv$ est l'opérateur d'équivalence sur Ω, tel que $\omega_1 \equiv \omega_2$ si et seulement si ω_1 et ω_2 ne peuvent être perceptivement différenciés.

En général un tel sous-ensemble n'existe pas. Il s'agit alors de trouver celui qui représente la configuration la plus proche de T. Nous avons introduit pour cela le concept de descripteur. Formellement, un descripteur est une fonction d de Ω dans $\mathbb{R}^p$, où p est la dimension du descripteur.

Formulation 7.2. *Soit $\{d_1, \ldots, d_N\}$ un ensemble de descripteurs sur Ω et $T \in \Omega$ le timbre recherché. La cible (au sens d'un ensemble de descripteurs) est alors définie par :*

$$T \Leftrightarrow \{d_1(T), \ldots, d_N(T)\} \tag{7.2}$$

Remarque 7.1. *Ce formalisme doit nous rappeler que le système d'orchestration ne peut « connaître » de la cible à orchestrer uniquement la description qu'il en a. Il ne cherchera donc des configurations qu'en fonction de ce qu'il entend de la cible, à travers le prisme de la description. Il ne faut donc pas s'attendre à trouver dans les orchestrations proposées des particularités de la cible que le système ne peut pas repérer. A titre d'exemple, la rugosité due à la modulation d'amplitude dans un son flatterzunge ne pourra pas être considérée à l'heure actuelle.*

[7] En traitement du signal cette opération pourrait s'apparenter à une somme de signaux.

[8] Nous rappelons que T est fourni au système sous forme de son enregistré dans le cas d'une orchestration imitative et représenté par un avatar de synthèse dans le cas d'une orchestration générative.

Les fonctions d'agrégation, quant à elles, peuvent être formalisées de la façon suivante :

Formulation 7.3. *Soit d un descripteur et $I = \{i_1, \ldots, i_I\}$ un sous-ensemble de $\{1, \ldots, N\}$. La fonction d'agrégation f_d associée à d vérifie alors :*

$$d\left(\bigoplus_{i \in I} s_i\right) = f_d\left(d\left(s_{i_1}\right), \ldots, d\left(s_{i_I}\right)\right) + \epsilon \tag{7.3}$$

où ϵ est une erreur d'estimation dont on montrera à la section 8.3 qu'elle est raisonnable.

De façon similaire, une fonction de comparaison est une application de Ω dans $\mathbb{R}$ vérifiant l'ensemble des propriétés suivantes :

Propriété 7.1. *Soit d un descripteur et T la cible d'un problème d'orchestration. Soit D_T^d la fonction de comparaison associée à d pour la cible T. On a alors :*

1. $\forall x \in \Omega, D_T^d(x) \geq 0$
2. $\forall x \in \Omega, D_T^d(x) = 0 \Leftrightarrow d(x) = d(T)$
3. *Pour tout couple (x, y) de Ω, $D_T^d(x) \leq D_T^d(y)$ si et seulement si x est perceptivement plus proche de T que x relativement au descripteur d.*

Remarque 7.2. *Les fonctions de comparaison utilisées dans notre outil et explicitées en annexe A vérifient par construction les points (1) et (2) de la propriété 7.1. En ce qui concerne le troisième point, nous sommes contraints d'en faire une hypothèse. Bien qu'elle semble raisonnable, elle n'est pas systématiquement vérifiée, en grande partie à cause de l'erreur d'estimation introduite dans la formulation 7.3. Nous montrerons au chapitre suivant que cette erreur est résonnable en pratiqu et ne remets pas en cause la propriété (3).*

Remarque 7.3. *Les travaux de Damien Tardieu [TR07] sur les modèles d'instruments permettront bientôt de se passer à la fois des fonctions d'agrégation et des fonctions de comparaison. L'évaluation d'une combinaison x sera alors une simple mesure de probabilité conditionnelle. La valeur de critère associée à un descripteur d et une cible T sera définie comme la probabilié d'observer la valeur $d(T)$ sachant la combinaison x :*

$$D_T^d(x) = P\left(d(T) \mid x\right)$$

Nous pouvons maintenant exposer, dans sa formalisation la plus générale, le problème de l'orchestration assistée par ordinateur vu comme une optimisation multi-objectif. Le caractère éventuellement contraint du problème (discuté en 7.5) ne remet pas en cause cette définition.

Formulation 7.4. *Soient Ω l'ensemble des sons possibles muni de l'opération interne $\oplus$, $(s_i)_{1 \leq i \leq N_s}$ la partie de Ω constituant la connaissance instrumentale du système et E le sous-ensemble de $(\Omega, \oplus)$, engendré par $(s_i)_i$, c'est-à-dire l'ensemble des configurations possibles. Soient T une cible à orchestrer, $(d_1, \ldots, d_N)$ un jeu de descripteurs sur Ω et $\left(D_T^{d_1}, \ldots, D_T^{d_N}\right)$ les fonctions de comparaison associées à $(d_1, \ldots, d_N)$ pour T. La découverte d'orchestrations de T se ramène alors au problème de minimisation multicritère suivant :*

$$\begin{cases} \min \ D_T^d(x) = \left(D_T^{d_1}(x), \ldots, D_T^{d_N}(x)\right) \\ s.t. \ x \in E \end{cases} \tag{7.4}$$

Nous avons commencé à l'évoquer à la section 4.3, le problème ainsi posé est « $\mathcal{NP}$-difficile » : sa complexité en temps est exponentielle. Nous verrons qu'il ne peut être abordé par des méthodes complètes[9] et qu'il nous faudra avoir recours à des heuristiques. Nous introduirons au chapitre 9 un algorithme évolutionnaire permettant d'obtenir en un temps raisonnable une bonne approximation du front de Pareto.

7.5 Contraintes

Nous avons identifié le problème de l'orchestration à une recherche multicritère de combinaisons de sons, dont la pertinence est jugée sur un ensemble de descripteurs. Si le formalisme introduit à la section précédente autorise l'évaluation de n'importe quelle configuration, il ne permet pas en revanche de modéliser d'éventuelles contraintes sur l'orchestration recherchée, ni donc de pénaliser les configurations qui ne satisferaient pas ces contraintes.

Il ne faut toutefois par perdre de vue que les combinaisons identifiées par le système comme étant les plus pertinentes relativement à un ensemble de critères doivent avant tout être exécutables par l'orchestre. Quel serait par exemple l'intérêt d'une solution nécessitant huit violoncelles pour un petit orchestre comme L'*Ensemble Intercontemporain* n'en comptant au maximum que deux ? En d'autres termes, l'effectif instrumental de l'orchestre impose des contraintes de cardinalité sur les instruments que les solutions trouvées par le système se doivent de respecter.

En outre, de même que le compositeur peut, par son interaction avec l'outil, « orienter » la recherche dans une direction privilégiée (cf. 7.2), de même la spécification de contraintes peut faciliter la découverte de solutions pertinentes en éliminant systématiquement certaines régions de l'espace de recherche. Dans notre système, les contraintes portent sur la caractérisation symbolique des sons, c'est-à-dire sur leurs attributs : hauteur, dynamique, instrument, mode de jeu, sourdine... Les contraintes sont donc mesurées directement dans l'espace de décisions. Le compositeur peut par exemple exiger des configurations proposées par le système qu'elles mobilisent obligatoirement certains instruments, que leurs composantes aient une dynamique commune ou encore qu'elles ne sollicitent qu'une partie de l'orchestre.

Définition 7.7. *Une* ***contrainte*** *est une relation logique que doivent satisfaire les attributs d'une configuration, c'est-à-dire l'ensemble de ses caractéristiques symboliques, en lien direct avec l'écriture musicale. Un ensemble de contraintes est appelé* ***réseau de contraintes****. une configuration est dite* ***consistante*** *si et seulement si elle satisfait toutes les contraintes du réseau.*

Définition 7.8. *Une contrainte est caractérisée par une fonction de coût, à valeurs réelles positives, qui à une configuration donnée associe le niveau de violation de la contrainte. La contrainte est satisfaite si et seulement si sa fonction de coût est nulle.*

Les fonctions de coût sont en lien direct avec ce que le monde de la programmation par contraintes (PPC) désigne par « contraintes molles » (*soft constraints*). En règle générale, une contrainte n'a que deux états possibles : soit elle est satisfaite, soit elle et violée. Les contraintes molles offrent davantage de souplesse en permettant de quantifier le degré de

[9] Les méthodes complètes, en procédant implicitement à une exploration totale de l'espace de recherche, garantissent l'optimalité des solutions trouvées. En revanche, elles nécessitent souvent des temps de calcul rédhibitoires pour les problèmes de grande taille.

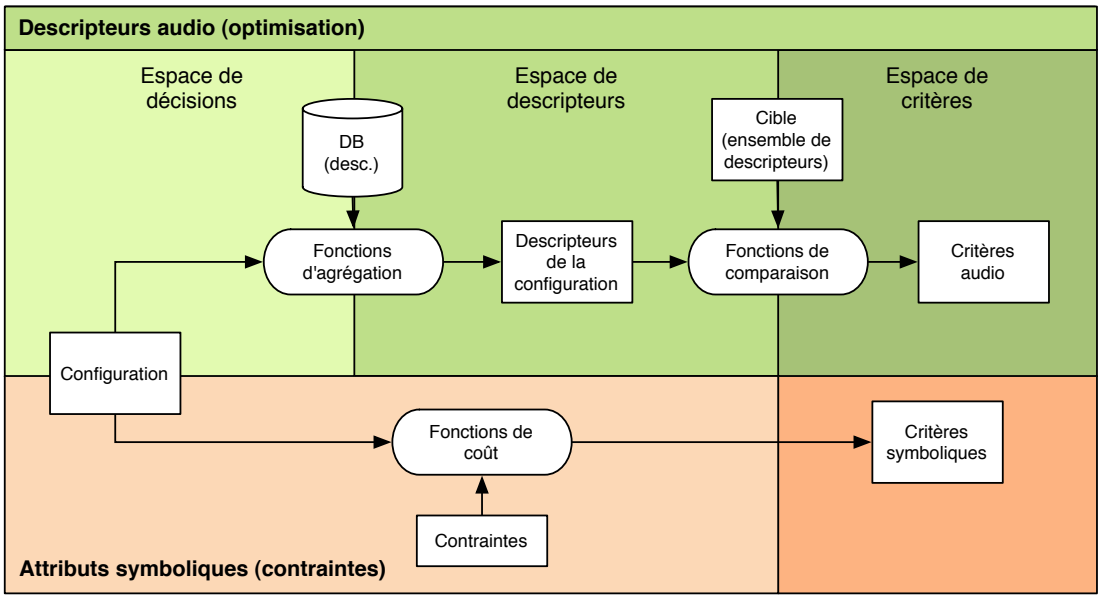

FIG. 7.3 – Les « trois mondes » de l'orchestration assistée par ordinateur

violation. Considérons par exemple un orchestre ne comprenant qu'un seul hautbois et qu'une seule flûte. Une proposition d'orchestration A à deux hautbois et une proposition B à trois flûtes sont toutes deux inconsistantes du point de vue de l'effectif orchestral. Pourtant, la fonction de coût sera inférieure pour A que pour B.

Le problème de l'orchestration devient donc un problème d'optimisation sous contraintes. La figure 7.3 illustre de quelle manière les problèmes peuvent être séparés. L'optimisation concerne les descripteurs d'une configuration, soit l'ensemble des paramètres du signal en rapport avec la caractérisation du timbre, tandis que les contraintes s'adressent aux attributs symboliques, c'est-à-dire les variables de l'écriture musicale.

Nous introduirons au chapitre 9 une modélisation du problème de l'orchestration permettant de s'affranchir des contraintes liées à l'effectif orchestral. Nous définirons au chapitre 10 un langage élémentaire de contraintes sur les attributs, et présenterons un algorithme efficace de réparation des solutions inconsistantes.

7.6 D'un espace à l'autre

L'orchestration assistée par ordinateur a donc été identifiée comme un processus d'optimisation multicritère. Dans ce type d'approche, la pertinence d'un point de l'espace de recherche (ou « espace de décisions ») est mesurée à l'aide d'une fonction multi-valuée à valeurs dans $\mathbb{R}^K$, où K est le nombre de critères considérés. Cette fonction d'évaluation transforme donc l'espace de recherche en un espace métrique. En d'autres termes, c'est un moyen de munir l'espace de recherche d'une topologie sans laquelle il n'y a pas d'optimisation possible.

Les méthodes mutli-critères, couramment utilisées dans les outils d'aide à la décision, permettent à l'utilisateur de choisir une solution de compromis entre un ensemble de critères

la plupart du temps inconciliables. En général, la signification des critères est accessible au décideur. Par exemple, un chef de projet industriel devant planifier la fabrication d'un nouveau produit considèrera conjointement le temps de fabrication, le coût des matières premières, la qualité du produit, les besoins des utilisateurs, l'horizon de rentabilité..., et prendra sa décision en fonction des *valeurs de critères* sans se précoccuper des paramètres de contrôle du processus industriel (c'est-à-dire des coordonnées dans l'*espace de décisions*) qui conduisent à ces valeurs. En d'autres termes, le décideur s'intéresse davantage aux objectifs à atteindre sans véritablement se soucier de la manière dont on les atteint. S'il utilise un outil d'aide à la décision pour déterminer par exemple cinq solutions différentes, il attend du logiciel cinq solutions de compromis dispersées dans l'espace des critères, rendant compte de la diversité des valeurs de critères possibles, et le choix final ne dépendra que des valeurs de ces critères.

En orchestration, le problème est différent. L'approche multicritère a été retenue non seulement pour explorer l'éventail des possibilités sonores mais surtout afin de tenir compte du caractère multidimensionnel de la perception du timbre, phénomène complexe pour lequel l'importance relative des composantes ne peut être connue à l'avance. Et c'est justement dans le but de découvrir ce qui importe au compositeur dans la cible à orchestrer que ce dernier est amené à choisir une solution particulière dans l'espace de critères. Or, ce choix ne saurait être motivé par les valeurs de critères elles-mêmes, qui ne sont qu'une mesure arbitraire de similarité selon telle ou telle dimension du timbre, mais par les qualités perceptives de la solution et par l'écriture (au sens musical) de cette solution, c'est-à-dire par sa position dans l'espace des décisions. La capacité de l'orchestre à « réaliser » un certain timbre importe certes au compositeur, mais sans doute autant que les moyens nécessaires pour y parvenir, à savoir, l'ensemble des paramètres de l'écriture musicale : hauteurs, dynamiques, modes de jeu, etc. Aussi le primat des critères semble-t-il moins évident ici. Allons plus loin : une fois les préférences du compositeur identifiées par le choix d'une solution dans l'espace des critères, ce dernier sera intéressé par connaître l'ensemble des points de l'espace des décisions qui « mènent » à ces valeurs de critères, c'est-à-dire l'ensemble des possibilités d'écriture du timbre recherché.

La particularité du problème de l'orchestration tient peut-être au type d'espace dans lequel se « situe » l'utilisateur. Dans un problème classique de décision multicritère, le décideur « vit » dans l'espace des critères, il connaît la signification de chaque critère et sait en interpréter les valeurs. Le compositeur, lui, vit dans l'espace des décisions, l'espace de la partition et de la symbolique de l'écriture. Les valeurs de critères ne lui parlent pas et il n'a d'ailleurs pas à les connaître. Or, un processus d'optimisation est toujours opéré dans un espace de critères, où se manifestent les effets des déplacements dans l'espace des décisions. D'où l'intérêt de l'approche descriptive : elle introduit un espace intermédiaire, l'*espace des descripteurs*, qui sert de « correspondance » entre l'écriture et l'optimisation.

Avant d'aller plus loin, et de chercher comment un problème d'optimisation sous contraintes peut-être résolu dans un espace de critères, il nous faut tout d'abord étudier les propriétés de chacun de ces espaces, et voir si les relations de similitude et de dissemblance entre leurs éléments sont transmissibles d'un espace à l'autre.

7.6.1 Espaces des décisions

C'est le monde du compositeur. Les coordonnées de ses éléments correspondent à des instruments, des hauteurs, des dynamiques, des modes de jeu, etc. On peut munir cet espace d'une distance qui permet de comparer deux éléments en fonction de leurs caractéristiques communes. Pour reprendre le formalisme introduit au paragraphe 7.4, si Ω est l'ensemble

des sons possibles muni de l'opération interne $\oplus$, $(s_i)_{1\leq i\leq N_s}$ la partie de Ω constituant la connaissance instrumentale et E le sous-ensemble de $(\Omega, \oplus)$, engendré par $(s_i)_i$, alors E est l'espace de décisions. Tout élément de E s'écrit comme la « somme » d'un sous-ensemble de $(s_i)_i$:

$$E = \left\{ \bigoplus_{i=1}^{N_s} x_i s_i \mid x_i \in \{0;1\} \right\} \tag{7.5}$$

On serait donc tenté de munir E une distance de type « Hamming », définie comme le nombre de coordonnées différentes entre deux vecteurs :

$$\|x - y\|_E = \sum_{i=1}^{N_s} |x_i - y_i| \tag{7.6}$$

Il n'est toutefois pas certain que ce choix soit très pertinent, car la distance sera la même entre une clarinette en si bémol et une clarinette en mi bémol jouant la même note à la même dynamique d'une part, et entre un violon col legno-tratto et un cor brassy d'autre part.

En revanche, on peut très bien utiliser une distance de Hamming sur les attributs, et comparer deux configurations selon leurs instruments, leurs hauteurs, leurs familles d'instruments... Ce principe sera utilisé au chapitre 8 pour comparer des solutions du point de vue du compositeur et au chapitre 13 pour explorer l'ensemble des solutions selon un critère symbolique.

7.6.2 Espace des descripteurs

Dans cet espace les coordonnées des sons dépendent directement des valeurs de descripteurs associés à chaque dimension du timbre. On peut y comparer toute paire de solutions, mais il est impossible d'y définir une distance mono-valuée, car les poids relatifs à chaque dimension du timbre changent selon le contexte (voir paragraphe 7.2), donc d'une comparaison à l'autre. Par défaut, on utilisera donc si nécessaire une distance euclidienne sur des descripteurs normalisés, en prenant toutes les précautions nécessaires pour l'interprétation.

7.6.3 Espace des critères

C'est le monde de l'optimisation. Les coordonnées de ses éléments sont des distances relatives à une cible sonore selon diverses dimensions de la perception du timbre. Il permet de notamment de deviner les préférences de l'utilisateur (cf. 7.2). Seule restriction (de taille), les distances dans cet espace ne s'interprètent que par rapport à la cible, située à l'origine du repère. Deux points sont voisins dans l'espace des critères si leurs distances relatives à la cible sur chacune des dimensions sont proches ; cela n'implique par pour autant une proximité perceptive entre les deux solutions. Par analogie avec la cartographie, cet espace s'apparente à la projection du globe établie au XVI$^{\text{e}}$ siècle par Postel. Dans cette projection, seuls sont respectés les distances et les angles calculés à partir du pôle, et la distance entre les continents paraît beaucoup plus petite qu'en réalité. Elle n'a d'intérêt que pour la navigation en zone polaire, de même que l'espace des critères n'a d'autre dessein que de positionner les solutions de l'espace de recherche par rapport à la cible. Seules les relations de dominance au sens de Pareto introduites à la section 7.2 n'ont de sens dans l'espace des critères. Nous verrons toutefois au chapitre 9 comment le choix d'une solution peut induire un ordre total sur l'espace des critères et permettre la comparaison de toute paires de solutions.

Le schéma de la figure 7.3 indique comment passer d'un espace à l'autre. Etant donné un point de l'espace de décision (c'est-à-dire une mixture de sons instrumentaux), les fonctions d'agrégation permettent d'estimer, à partir des descripteurs individuels de chaque son, les descripteurs de la mixture selon chaque dimension du timbre. Dans un second temps, les fonctions de comparaison permettent de calculer les distances relatives à la cible pour chaque descripteur.

Ces fonctions ne sont pas monotones. Il suffit de considérer la valeur absolue des fonctions de comparaisons des moments spectraux pour s'en persuader (voir annexe A). Ainsi, réduire la distance entre deux configurations dans l'espace des critères n'implique pas que cette distance diminue dans l'espace des décisions et vice-versa.

A la non-monotonie des ces fonctions transformant l'espace des décisions en espace des critères s'ajoute leur non-injectivité. Les deux espaces extrêmes de la figure 7.3 ont donc des topologies *très* différentes. Nous ne devons pas perdre du vue que l'approche multicritère se justifie avant tout par l'aspect multidimensionnel de la perception du timbre et que l'espace des critères n'est là que pour distinguer les solutions efficientes des solutions dominées, et pour estimer l'importance relative des dimensions de l'écoute, ainsi qu'exposé au paragraphe 7.2. Le compositeur, de son côté, juge la qualité des résultats obtenus dans l'espace des décisions. Nous aurons donc à nous poser la question fondamentale suivante : Les solutions optimales du problème d'optimisation sous contraintes, à savoir les configurations à la fois *efficientes* et *consistantes*, correspondent-elles à des orchestrations intéressantes pour le compositeur ? En d'autres termes, le problème de l'orchestration formulé en 7.4 est-il bien posé ? Ce sera le propos du chapitre 8.

Chapitre 8

Validation sur une description réduite du timbre

Description spectro-harmonique du timbre
Base de test : problèmes de petite taille
Evaluation des fonctions d'agrégation
Le problème est-il bien posé ?

Nous proposons dans ce chapitre une validation expérimentale des concepts et formalismes introduits au chapitre précédent. Nous commençons par introduire un ensemble réduit de descripteurs du timbre puis montrons qu'ils permettent de traiter une classe particulière de problèmes, sans remettre en cause le cadre théorique du chapitre 7.

Nous évaluons dans un premier temps la qualité des fonctions d'agrégation en comparant, pour un ensemble conséquent de mixtures de test, l'estimation des descripteurs au calcul par extraction directe. Dans un second temps, nous prouvons que le problème de l'orchestration assisté par ordinateur, reformulé en 7.4 comme un problème d'optimisation multicritère sous contraintes, est « bien posé » : nous calculons pour cela les fronts de Pareto pour un ensemble de problèmes de petite taille et montrons qu'ils contiennent des configurations pertinentes pour le compositeur.

8.1 Description « réduite » du timbre instrumental

8.1.1 Hypothèses et limitations

Nos recherches sur l'orchestration assistée par ordinateur sont nées d'une demande de compositeurs formulée au sein d'un groupe de travail de l'IRCAM. Les suggestions émises par les membres de ce groupe (voir 4.2) fournirent alors une matière suffisante pour vingt ans de recherche. La nécessité de disposer rapidement d'un outil d'orchestration a donc nécessité certaines hypothèses simplificatrices. Elles ont été brièvement introduites dans [CTA+06]. Nous les rappelons ici de manière plus détaillée :

Aspects temporels : Tous les sons manipulés par notre système (aussi bien la cible que les échantillons de la connaissance instrumentale) sont supposés entretenus et sans évolution temporelle. Les non-entretenus (piano, guitare, harpe, glockenspiel, pizzicati...), de même que

les sons présentant une variation temporelle (trémolos, vibratos, crescendos, decrescendos, glissandi...) sont absents de la banque de sons instrumentaux.

Composantes harmoniques, composantes bruitées : La cible et les sons de la base sont supposés contenir des composantes sinusoïdales. Les sons uniquement bruités ne peuvent être décrits, ni donc traités. Les percussions à sons potentiellement entretenus (caisse claire, cymbales...) sont donc absentes de la base. En outre, la modélisation du bruit n'est pas supportée. Les aspects transitoires du son sont négligés.

Connaissance instrumentale : Nous supposons que la base de sons constituant la connaissance instrumentale est représentative des possibilités de jeu dans le cadre des restrictions exposées ci-dessus. Nous supposons également que la base est cohérente du point de vue des dynamiques (voir 7.1.2) : nous faisons l'hypothèse que les niveaux d'enregistrement reflètent fidèlement la puissance acoustique des instruments et permettent d'estimer avec précision la qualité des mélanges. Nous restreignons en outre la connaisance instrumentale aux sons harmoniques monophoniques, pour lesquels la hauteur est clairement identifiée.

Effets d'espace : En dernier lieu, nous négligeons les qualités acoustiques du lieu d'écoute ainsi que les positions des instrumentistes et de l'auditeur. L'hypothèse sous-jacente est que le timbre obtenu par un mélange instrumental ne dépend que des paramètres d'écriture et non des conditions d'exécution de la musique.

En regard de l'expérience musicale et des souhaits des compositeurs (voir 4.2) ces hypothèses pourront paraître extrêmement réductrices voire irréalistes. Elles ont toutefois permis, après trois années de recherche, de construire un outil d'orchestration dont les potentialités dépassent déjà ce que l'état de l'art propose aujourd'hui. En outre, la considération de phénomènes tels que les effets d'espace n'a de sens qu'une fois levées les imprécisions de la connaissance instrumentale. Il serait en effet absurde de calculer une réverbération avec une grande précision sans parfaitement maîtriser les caractéristiques timbrales des sources sonores. Il nous semble donc légitime de considérer que de telles hypothèses n'affectent pas, aujourd'hui, la qualité des résultats de façon significative.

8.1.2 Descripteurs retenus pour l'orchestration

Nous introduisons ici les descripteurs utilisés par notre système. Compte tenu des hypothèses formulées en infra, les aspects temporels et spectro-temporels du timbre ne sont pas pris en compte. Le signal sera donc toujours un segment de son correspondant à une partie statique du timbre ; l'attaque, en particulier, n'y figurera jamais.

Nos descripteurs se divisent en deux catégories : harmoniques, multidimensionnels (principaux partiels résolus) et spectraux, scalaires (centroïde spectral et étendue spectrale). Nous ne donnerons pas ici de détails techniques quant au calcul et à la manipulation de ces descripeurs. Le lecteur désireux d'approfondir ces notions est invité à consulter l'annexe A.

Principaux partiels résolus (MRP)

Rappelons que l'espace de timbres de McAdams et al. [MWD+95] ne concerne que des sons de même hauteur et de même intensité perceptive. Une description harmonique de la cible nous

a donc paru nécessaire. Elle consiste en un ensemble des principaux partiels « résolus » par le système auditif. On sait en effet que la membrane basilaire de l'oreille interne se comporte comme un ensemble de filtres passe-bande [Zwi61]. Un partiel est « résolu » lorsqu'il est seul à l'intérieur d'une bande critique. Si une même bande contient plusieurs partiels, seul le plus élevé en amplitude est considéré comme résolu.

Nous désignerons désormais par MRP (*Main Resolved Partials*) les principaux partiels résolus. Dans le cas d'un son monophonique (de hauteur unique) ou polyphonique (perçu comme un accord), les MRP jouent un rôle déterminant dans la perception de la (des) hauteur(s). Dans le cas d'un son complexe (une cloche par exemple), ils déterminent la « couleur harmonique » du son.

Centroïde spectral

Le centroïde spectral est souvent défini comme le centre de gravité spectral du son. Les psychoacousticiens le relient à la brillance : les sons brillants ont un centroïde élevé, les sons sourds un centroïde bas. Son rôle dans la perception et les jugements de similarité de timbre n'a plus à être démontré : le centroïde spectral figure toujours parmi les dimensions principales des espaces de timbres (voir section 2.5).

Le centroïde peut être calculé en considérant le spectre comme la densité d'une variable aléatoire dont les valeurs sont les fréquences du spectre et les probabilités d'observation sont les amplitudes normalisées [Pee04] :

$$sc = \int_0^\infty f\ a(f)\ df \tag{8.1}$$

Etant donné que les fréquences peuvent être exprimées sur une échelle linéaire ou logarithmique, et que le spectre peut être calculé en amplitude, en energie ou en décibels, il y a en tout six façons de calculer le centroïde spectral. Pour des raisons d'additivité des descripteurs, nous préférons le centroïde linéaire, calculé avec des fréquences linéaires et un spectre d'amplitude.

La formule 8.1 permet de calculer le centroïde instantané, associé à une fenêtre temporelle unique. Le centroïde global (caractérisant la brillance générale du son) est obtenu par une moyenne du centroïde instantané, pondérée par l'intensité perceptive sur chaque fenêtre temporelle (voir A).

Etendue spectrale (*spread*)

De même que le centroïde est la moyenne du spectre (ou premier moment spectral), l'étendue spectrale correspond à l'écart-type (second moment spectral), appelé couramment *spread*. D'un point de vue perceptif, il est relié à l'épaisseur spectrale du son (à ne pas confondre avec l'épaisseur spatiale de Koechlin [Koe43]). L'épaisseur spectrale est en quelque sorte la bande passante ; une voix humaine sera plus épaisse en conditions naturelles d'émission que sortant d'un vieil appareil de radio. Pour donner un exemple musical, à hauteur et centroïde équivalents, une trompette a un son très épais avec une sourdine Harmon et très étroit avec une sourdine Cup. En termes probabilistes, l'étendue spectrale s'écrit [Pee04] :

$$ss = \int_0^\infty (f - sc)^2\ a(f)\ df \tag{8.2}$$

où sc est le centroïde spectral du son. De même que le centroïde, il y a six calculs possibles pour l'étendue. Toujours pour des raisons d'additivité des descripteurs, nous travaillerons avec un

spread linéaire, calculé avec des fréquences linéaires et un spectre d'amplitude. La formule 8.2 renvoie également une valeur instantanée. Le calcul de l'étendue globale fait intervenir l'intensité perceptive instantannée (voir A).

8.1.3 Extensibilité de la description

Nous nous limitons donc a une vision spectrale, ou spectro-harmonique, du timbre, et sommes conscients que cette simplification néglige de nombreux aspects du son. Helmoltz déjà, pour qui les différences de perception sonore n'ont longtemps tenu qu'à la fréquence et à l'amplitude des partiels, s'était rendu compte de l'importance des facteurs temporels et des parties transitoires. « La caractéristique des sons la plus significative d'un point de vue musical n'est pas qu'ils soient identifiables en tant qu'entités multidimensionnelles cohérentes, mais qu'ils présentent des variations temporelles significatives, et conservent toutefois leur identité. » (Erickson, *Sound Structure in Music* [Eri75]).

On sait aujourd'hui que ces variations temporelles jouent un rôle majeur dans notre perception du timbre. Schouten [Sch68] a proposé un ensemble de dimensions pour la caractérisation du timbre, adaptées aux préoccupations contemporaines : rapport entre composantes tonales et bruitées, enveloppe spectrale, enveloppe temporelle, glissement de formants, variation de fréquence fondamentale, forme de l'attaque. Tous ces aspects sont désormais accessibles à la description des sons [PMH00] [Pee04] et peuvent être extraits directement du signal audio à l'aide du programme *ircamdescriptor* de Geoffroy Peeters. Or, nous avons vu en 7.2.2 que disposer d'un modèle descriptif n'est pas suffisant pour le problème de l'orchestration ; la difficulté principale réside dans la prédiction des descripteurs d'un mélange de sons. Il serait en effet impensable — en termes de temps de calcul — d'évaluer la qualité des propositions d'orchestration en générant un « mixage » des composantes, puis en extrayant les descripteurs de la mixture (voir section 7.2.2).

Nous avons donc besoin pour chaque descripteur utilisé, non seulement d'une méthode d'extraction, mais aussi d'une méthode d'addition (cf. 7.3). Les descripteurs introduits en 8.1.2 satisfont cette exigence sans toutefois nécessiter une expertise approfondie du domaine. En outre, cette caractérisation réduite du timbre nous a déjà permis d'obtenir des résultats significatifs auprès des compositeurs. Après tout, notre description recouvre une bonne partie de l'espace de timbres de McAdams et al. [MWD$^+$95][1]...

Rappelons également que la conception et la manipulation des descripteurs du timbre ne constituent pas le cœur de nos travaux. En attendant de disposer d'un ensemble de descripteurs « additifs » du timbre, nous avons avancé dans nos recherches avec une description simple mais facilement manipulable.

8.2 Création d'une base de test

Le problème d'une évalutation quantitative de nos approches ou de nos méthodes s'est présenté à plusieurs reprises au cours de nos travaux : mesures de la précision des fonctions d'agrégation (section 8.3), pertinence de la description et de l'approche multicritère (section 8.4 et 8.5), performances de l'algorithme de réparation des configurations inconsistantes

[1] Il n'en effet pas aberrant de considérer l'irrégularité spectrale comme « incluse » dans les principaux partiels résolus (MRP), les différences d'amplitude entre les partiels d'importance moindre pouvant être considérées comme moins significatives dans le calcul de l'irrégularité spectrale.

(chapitre 10), évaluation de l'algorithme d'orchestration (chapitre 11). Pour l'ensemble de ces évaluations, une base commune d'instances de test a été générée. Chaque cas de test est un son résultant d'un mélange d'échantillons de la connaissance instrumentale, qui peut servir de cible pour un problème d'orchestration. Nous détaillons dans cette section le processus de création de cette base.

8.2.1 Génération de mixtures

La connaissance instrumentale de notre outil d'orchestration provient de la base *Studio Online* (SOL) [BBHL99], enregistrée à l'IRCAM (voir section 7.1.2). Seuls sont utilisés par notre système les échantillons de sons harmoniques, entretenus et sans variations temporelles, au sens large — les échantillons présentant des modulations d'amplitude (trémolos) et de fréquences (vibratos) sont exclus. Les raisons de ces restrictions ont été exposées en 8.1.1. Notre base de données compte ainsi 4763 échantillons, soit environ un tiers de SOL, et regroupe les instruments suivants : flûte, hautbois, clarinette en Si bémol, basson, trompette en Do, trombone ténor, cor, violon, alto, violoncelle et contrebasse.

Enregistrés en solo, en dehors de tout contexte musical, les échantillons de SOL sont la plupart du temps très légèrement faux. Lorsqu'on les mélange pour générer des mixtures de test ou pour simuler des orchestrations, de pénibles phénomènes de dissonance et de modulation d'amplitudes apparaissent. Nous avons donc « repitché » automatiquement les échantillons de notre propre base à l'aide une analyse en fréquence fondamentale et d'un algorithme de « stretch » dans *SuperVP* [DP91].

A partir de ces sons individuels, nous avons dans un second temps généré des mixtures instrumentales de deux types. Les premières, que nous qualifions de « monophoniques », n'incluent que des sons de même hauteur, et leur cardinalité varie entre 1 et 4 sons. Les secondes, « polyphoniques », sont formées à partir de 2, 3 ou 4 sons de la base, de hauteurs toutes différentes. Pour ces dernières, l'ambitus de l'accord (c'est-à-dire l'écart entre la note la plus haute et la plus basse) n'excède jamais deux octaves, afin de favoriser la fusion des timbres — on sait en effet que deux sons harmoniques de hauteurs trop éloignées sont perçus comme appartenant à deux « plan sonores » distincts. Dans tous les cas, les mixtures ne peuvent impliquer deux fois le même instrument, et leur tessiture est limitée à Do2 [2] dans le grave et Ré5 dans l'aigu.

La méthode d'addition utilisée est une simple somme de signaux, chaque signal étant préalablement convolué avec la réponse impulsionnelle d'une petite salle, différente à chaque fois et puisée aléatoirement parmi un ensemble de réponses générées numériquement avec le programme de spatialisation *Spat* [Jot97]. Cette opération permet de simuler un effet de salle en créant une réverbération artificielle. Son but est d'une part d'éviter d'invraisemblables effets de phase et de modulation d'amplitude, fréquents lors de l'addition de signaux harmoniques de même hauteur. D'autre part, elle vise à s'abstraire de la spécificité de l'échantillon liée à l'instrumentiste et aux conditions d'enregistrement (voir section 7.1.2).

500 mixtures de chaque classe ont été générées, soit une base de test rassemblant 3500 instances de cardinalité variable.

[2] Dans l'intégralité de ce document, Do4 désigne le Do central. C'est la convention de notation adoptée par la base SOL.

8.2.2 Construction de cibles : description et contraintes

Les descripteurs audio introduits au paragraphe 8.1.2 ont été extraits de chacune des mixtures selon le processus exposé en annexe A. Cet ensemble de descripteurs va servir dans un premier temps à l'évaluation des fonctions d'agrégation (voir paragraphe 8.3). Ensuite, chaque jeu de descripteurs sera considéré comme la cible d'un problème d'orchestration imitative dont le but sera de retrouver la mixture originale.

Chaque cible est en outre complétée par trois contraintes permettant de réduire la taille de l'espace de recherche (rappelons que ce dernier ne comporte que des configurations consistantes) :

- Contraintes de hauteur : L'espace de recherche est limité aux configurations comprenant *toutes* les hauteurs de la mixture cible, et *uniquement* ces hauteurs-là. Pour les mixtures monophoniques, l'espace est donc restreint à des combinaisons de sons de hauteurs identiques ; pour les cibles polyphoniques, le nombre de hauteurs différentes dans une configuration est égal au degré de polyphonie de la mixture.
- Contraintes d'effectif : L'espace de recherche est limité aux configurations n'utilisant qu'une seule fois le même instrument.
- Contraintes de cardinalité : L'espace de recherche est limité aux configurations impliquant le même nombre d'instruments que la mixture cible. Pour les mixtures à trois sons, on ne cherchera donc que des solutions à trois sons.

Les mixtures cibles vérifient systématiquement les trois contraintes de hauteur, effectif et cardinalité. Elle font donc à chaque fois partie de l'espace de recherche. L'objet de la section 8.4 sera justement d'y repérer leur position.

8.2.3 Espaces de recherche, fronts de Pareto

Chacune des 3500 mixtures de la base de test a donc été considérée comme la cible d'un problème d'orchestration particulier, assortie de contraintes de hauteur, d'effectif et de cardinalité. Pour chaque problème, l'intégralité des configurations consistantes a été générée. Le tableau 8.1 donne un aperçu de la cardinalité des espaces de recherche. Pour les mixtures à 4 sons, le temps de calcul est déjà de l'ordre de plusieurs minutes. L'exploration systématique (*brute force*) des combinaisons possibles dans la résolution de problèmes de grande taille sera donc à proscrire. Nous verrons au chapitre 9 de quelle manière le problème combinatoire peut être traité en un temps raisonnable.

Les espaces de critères sont alors obtenus par applications successives des fonctions d'agrégation et fonctions de comparaison sur les espaces de décisions. Nous extrayons de chaque espace de critères les fronts de Pareto à l'aide de l'algorithme 1. Afin de prouver la validité de l'approche multicritère et l'absence de redondance dans l'emploi de critères exclusivement spectro-harmoniques, nous considérons pour chaque problème plusieurs combinaisons de critères. De chaque espace de critères nous extrayons donc 7 fronts de Pareto correspondant aux différents jeux de critères du tableau 8.2.

La complexité de l'algorithme 1 est dans le pire des cas en $\mathcal{O}(n^2K)$, où n est la taille de l'espace de recherche et K le nombre de critères. Afin de traiter les problèmes à 4 sons en un temps raisonnable nous créons une partition de l'espace de recherche en sous-ensembles de l'ordre de 10^5 configurations, et utilisons l'algorithme 2 de mise à jour du front de Pareto. Sa complexité est dans le pire cas en $\mathcal{O}(npK)$, où K est le nombre de critères, p la taille du front courant et n la taille du sous-ensemble de l'espace des critères utilisé pour la mise à jour.

TAB. 8.1 – Tailles moyennes des espaces de recherche pour des problèmes de petite taille (seules sont comptabilisées les configurations consistantes avec les contraintes de hauteur, d'effectif et de cardinalité).

Type de cible	Taille de l'espace
monophonique - 1 son	82
monophonique - 2 sons	$2,85.10^3$
monophonique - 3 sons	$7,29.10^4$
monophonique - 4 sons	$1,01.10^7$
polyphonique - 2 sons	$5,70.10^3$
polyphonique - 3 sons	$3,65.10^5$
polyphonique - 4 sons	$2,00.10^7$

Algorithme 1 Extraction du front de Pareto (`extract_pareto_front`)

Arguments: Une partie $C = \{c_1, c_2, \ldots\}$ de l'espace de critères.

1: $P \leftarrow \{c_1\}$
2: **pour** $i = 2 \ldots \#C$ **faire**
3: $P \leftarrow P \cup \{c_i\}$
4: **pour tout** $p \in P \wedge p \neq c_i$ **faire**
5: **si** $c_i \prec p$ **alors**
6: $P \leftarrow P \setminus \{p\}$
7: **sinon si** $p \prec c_i$ **alors**
8: $P \leftarrow P \setminus \{c_i\}$
9: **aller** à la ligne 2
10: **fin si**
11: **fin pour**
12: **fin pour**

TAB. 8.2 – Différentes combinaisons de critères testées

$k_{(1)}$	centroïde seul
$k_{(2)}$	spread seul
$k_{(3)}$	MRP seuls
$k_{(12)}$	centroïde + spread
$k_{(13)}$	centroïde + MRP
$k_{(23)}$	spread + MRP
$k_{(123)}$	centroïde + spread + MRP

Algorithme 2 Mise à jour du front de Pareto (`update_pareto_front`)

Arguments: Une partie $C = \{c_1, c_2, \ldots\}$ de l'espace de critères, un ensemble P de solutions non dominées, à mettre à jour avec les éléments de C.

1: **pour** $i = 1 \ldots \#C$ **faire**
2: $P \leftarrow P \cup \{c_i\}$
3: **pour tout** $p \in P \wedge p \neq c_i$ **faire**
4: **si** $c_i \prec p$ **alors**
5: $P \leftarrow P \setminus \{p\}$
6: **sinon si** $p \prec c_i$ **alors**
7: $P \leftarrow P \setminus \{c_i\}$
8: **aller** à la ligne 1
9: **fin si**
10: **fin pour**
11: **fin pour**

Les tableaux 8.3 recensent les tailles moyennes des fronts de Pareto pour chaque type de mixture et chaque combinaison de critères. De façon évidente, la taille des fronts augmente toujours avec le nombre de critères. En effet, une configuration efficiente dans un espace réduit de critères l'est encore dans un espace plus large, alors que l'inverse n'est pas vrai. C'est un des effets de la multidimensionnalité, qui veut que les distances diminuent quand la dimension des espaces augmente. Nous aurons l'occasion de rediscuter ces phénomènes et les conséquences qu'ils engendrent pour notre problème. Ce qui nous semble remarquable à ce stade, c'est que la taille relative des fronts de Pareto par rapport aux espaces de recherche chute considérablement avec la taille du problème, comme en atteste le dernier tableau de 8.3. Seule une configuration sur 10000 est efficiente dans un espace à trois critères pour une cible polyphonique de 4 sons ; le problème est donc d'autant plus difficile que la cardinalité de l'espace de recherche est élevée. L'augmentation du nombre de critères ne se traduit pas par une explosion du nombre de solutions efficientes.

Dans la situation idéale où les fonctions d'agrégation prédiraient exactement les valeurs issues de l'analyse, il est clair que les distances relatives entre la cible analysée et la configuration composée des éléments de la cible serait nulle pour chacun des critères. Corrélativement, le front de Pareto serait alors réduit à un unique point, l'origine du repère. D'une manière plus générale, on peut dire qu'en optimisation multicritère la taille de fronts de Pareto diminue d'autant plus qu'il est possible d'obtenir simultanément de « bonnes » valeurs sur chacun des critères, comme cela semble être le cas pour nos problèmes de test. Il est donc normal que les fronts soient de faible cardinalité.

La figure 8.1 résume le processus de création d'une instance de test. A partir d'un jeu de hauteurs et d'un orchestre ne contenant qu'un seul instrument de chaque type, on déduit dans un premier temps un ensemble de contraintes de hauteur, d'effectif et de cardinalité. Ces contraintes vont servir d'une part à délimiter l'espace de recherche, d'autre part à générer une mixture instrumentale consistante formée d'échantillons de la connaissance instrumentale (les boîtes « RI » désignent les réponses impulsionnelles d'une salle de petite taille, voir 8.2.1). Les descripteurs extraits de la mixture constituent alors la cible d'un problème d'orchestration pour lequel plusieurs fronts de Pareto sont extraits de l'espace des critères. Une instance de test est donc composé d'une cible, d'un jeu de contraintes, d'un espace de recherche (et de

TAB. 8.3 – Tailles moyennes des fronts de Pareto

Mixtures monophoniques

	1 son	2 sons	3 sons	4 sons
$k_{(1)}$	1	1	1	1
$k_{(2)}$	1	1	1	1
$k_{(3)}$	1	1	1	1
$k_{(12)}$	4.3	7.9	10.9	13.7
$k_{(13)}$	5.0	22.5	326.4	81.4
$k_{(23)}$	5.1	23.1	329.4	90.6
$k_{(123)}$	9.0	44.3	402.6	244.3

Mixtures polyphoniques

	2 sons	3 sons	4 sons
$k_{(1)}$	1	1	1
$k_{(2)}$	1	1	1
$k_{(3)}$	1	1	1
$k_{(12)}$	8.6	12.7	16.6
$k_{(13)}$	25.5	34.7	28.5
$k_{(23)}$	25.8	35.3	27.5
$k_{(123)}$	50.8	115.2	196.9

Proportion de l'espace de recherche (seuls les fronts à 3 critères sont reportés ici)

Type de cible	Ratio
monophonique - 1 son	11.11%
monophonique - 2 sons	1.55%
monophonique - 3 sons	0.55%
monophonique - 4 sons	0.002%
polyphonique - 2 sons	0.89%
polyphonique - 3 sons	0.03%
polyphonique - 4 sons	0.001%

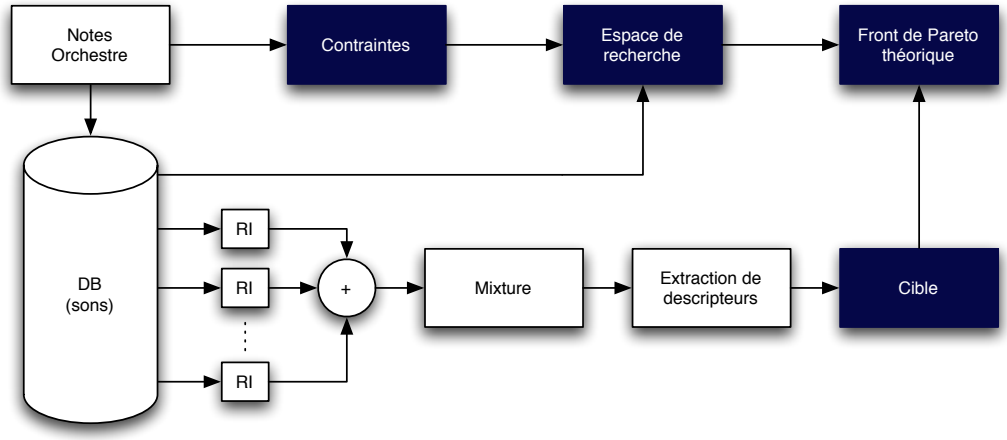

FIG. 8.1 – Processus de création d'une instance de test (un cas de test est constitué de l'ensemble des éléments en bleu).

l'espace de critères associé) et de fronts de Pareto pour différentes combinaisons de critères.

8.3 Evaluation des fonctions d'agrégation et de comparaison

Si $(s_i)_{1\leq i\leq N_s}$ la partie de Ω constituant la connaissance instrumentale, alors à chaque instance de la base de test correspond une mixture M un sous-ensemble $I \subset \{1,\ldots,N_s\}$ tel que :

$$\bigoplus_{i\in I} (s_i \oplus r(s_i)) = M$$

où $r(s)$ désigne réverbération ajoutée à une composante s.

Définition 8.1. *Nous appellons* ***solution théorique*** *associée à une instance de test représentée par une mixture M, la configuration $(s_i)_{i\in I_T}$ dont les éléments sont les composantes de la mixture M. Théoriquement, c'est la configuration de l'espace de recherche la plus proche de M. Nous la notons K_T, et on a :*

$$K_T = \bigoplus_{i\in I_T} s_i$$

Remarque 8.1. *Si les fonctions d'agrégation sont de bons prédicteurs des descripteurs d'une configuration, peu sensibles à la réverbération, on doit alors observer les relations suivantes :*

$$\forall d,\ D_M^d(K_T) \simeq 0\ ,$$

où $D_M^d(K_T)$ est la fontion de comparaison entre la cible M et la solution théorique K_T pour le descripteur d (voir définition 7.6 et propriété 7.1).

Nous cherchons dans cette section à évaluer l'« erreur » intrinsèque ϵ_d des fonctions d'agrégation relatives à chaque descripteur d (voir définition 7.5 et formulation 7.3), mentionnée au paragraphe 7.4 :

$$d\,(M) = f_d\left(d\,(s_{i_1}),\ldots,d\left(s_{i_{I_T}}\right)\right) + \epsilon_d\ , \tag{8.3}$$

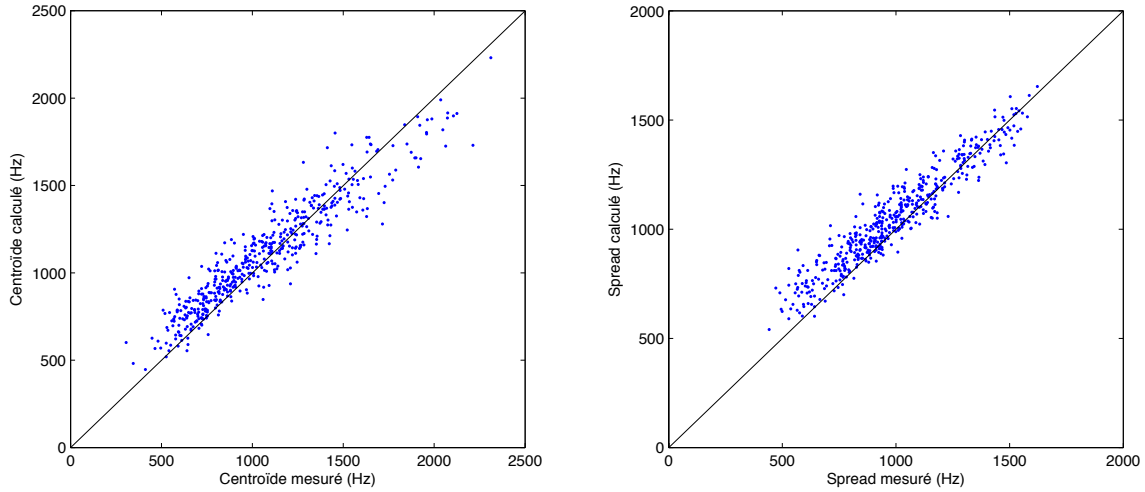

FIG. 8.2 – Estimation des moments spectraux des cibles polyphoniques à 4 sons

La figure 8.2 permet de se faire une idée de la qualité de l'estimation des moments spectraux pour les cibles polyphoniques à 4 sons. Des nuages de points similaires sont obtenus pour les autres types de mixtures. En ce qui concerne les MRP (rappelons qu'il s'agit d'un descripteur multidimensionnel), la représentation de l'estimation est plus problématique, car l'erreur d'estimation définie en 8.3 est de même dimension que le descripteur associé. Afin de comparer des erreurs exclusivement scalaires, nous choisissons donc de mesurer, pour les MRP, l'erreur « propagée » dans la fonction de comparaison :

$$D_M^{MRP}(K_T) = \epsilon'_{MRP}$$

Nous évaluons donc, pour chaque instance de test, les descripteurs associés à la solution théorique, puis nous calculons les erreurs d'estimation des moments spectraux et des MRP. L'erreur relative aux moments est définie par $\epsilon = (\widehat{d}_{K_T} - d_M)/d_M$ et peut donc être négative en cas de sous-évaluation. L'erreur relative au MRP s'écrit $\epsilon_{MRP} = D_M^{MRP}(K_T)$, elle est donc toujours positive. Les moyennes et écarts-type de ces erreurs sont reportées dans les tableaux 8.4. La figure 8.3 permet en outre de se faire une idée de la distribution des fonctions de comparaison pour les cibles polyphoniques à 4 sons.

Les tests sur les sons isolés (mixtures monophoniques à une seule composante) permettent de quantifier l'effet de la réverbération sur les valeurs de descripteurs. Il semble que dans notre cas l'effet de salle (ou la méthode utilisée pour le simuler) diminue en moyenne les valeurs des moments spectraux (donc rende le son moins brillant et moins large), tout en conservant bien l'enveloppe spectrale dans la partie basse du spectre. Nous expliquons ces phénomène en rappelant que la réverbération se comporte en général comme un filtre passe-bas.

Pour l'ensemble des instances de test, le biais d'estimation est d'environ +11% pour le centroïde et +8,5% pour l'étendue spectrale. Même si l'on peut considérer ces valeurs comme relativement élevées, leur constance doit nous rassurer : la qualité de l'estimation ne se dégrade pas avec la taille de la mixture. Il y a donc fort à parier que l'erreur d'estimation des moments spectraux soit uniquement due à la réverbération pour les mixtures considérées dans ces tests.

TAB. 8.4 – Erreurs moyennes d'estimation des descripteurs (écart-type de l'erreur d'estimation)

Mixtures monophoniques

	1 son	2 sons	3 sons	4 sons
centroïde	10.4% (9.7%)	10.7% (9.5%)	11.1% (9.2%)	11.2% (9.3%)
spread	8.8% (7.6%)	8.2% (8.5%)	7.7% (8.0%)	7.5% (7.7%)
MRP	5.0% (5.8%)	5.1% (5.5%)	5.0% (5.1%)	4.7% (5.5%)

Mixtures polyphoniques

	2 sons	3 sons	4 sons
centroïde	11.0% (10.4%)	11.2% (9.9%)	11.2% (10.4%)
spread	9.0% (8.8%)	9.5% (9.1%)	9.1% (9.2%)
MRP	5.9% (6.1%)	6.3% (6.9%)	6.6% (7.3%)

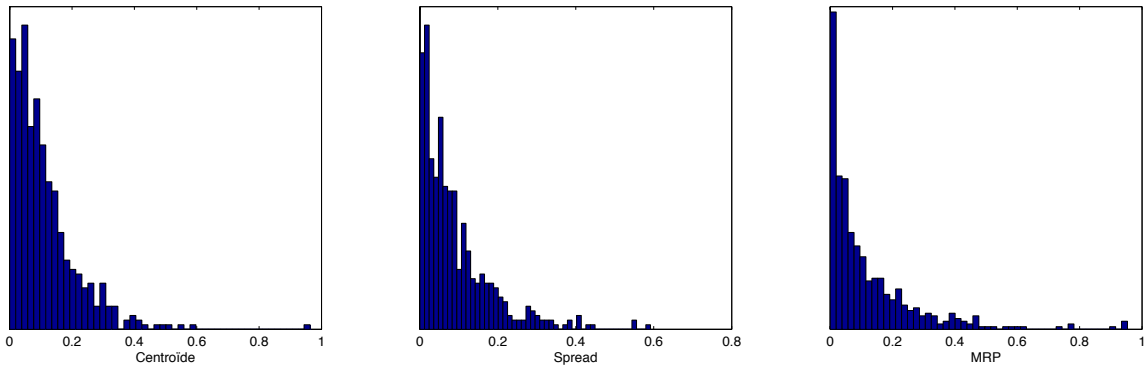

FIG. 8.3 – Distribution des fonctions de comparaison pour les cibles polyphoniques à 4 sons

En ce qui concerne les MRP, les faibles valeurs de la distance en cosinus indiquent une forte corrélation entre les vecteurs d'amplitude des partiels les plus importants, donc une grande similitude entre les enveloppes spectrales mesurées et calculées.

N.B. Rappelons encore une fois que si les deux premiers moments spectraux correspondent bien à des dimensions perceptives (la brillance pour le centroïde et l'épaisseur pour l'étendue), ces relations n'ont jamais été établies que pour des sons monophoniques. Nous ne savons donc a priori pas comment interpréter une valeur de centroïde ou d'étendue pour des mixtures dont les éléments ont des hauteurs différentes. La seule conclusion que nous pouvons tirer de notre test est que l'estimation des moments spectraux par le calcul et leur extraction par un processus d'analyse conduisent approximativement aux mêmes valeurs. Nous verrons toutefois à la section 8.5.4 que les moments spectraux contribuent grandement à l'identification de solutions perceptivement pertinentes.

8.4 Position de la solution théorique dans l'espace des critères

Nous avons affirmé au paragraphe 7.4 qu'en général l'équation 7.1 n'a pas de solutions. Exceptionnellement ici, dans le cadre de notre base de test, chaque problème admet au moins sa solution théorique (voir définition 8.1) comme solution :

$$K_T \equiv M$$

Dans l'hypothèse où l'erreur d'estimation des fonctions d'agrégation serait nulle, le front de Pareto associé à une instance de test serait alors réduit à la seule solution théorique (voir section 8.2.3). Or, les tableaux 8.3 prouvent bien que cette erreur ne peut être négligée, et que les fronts contiennent bien plus qu'une seule configuration. Mais contiennent-ils au moins la solution théorique ?

D'un point de vue perceptif, la mixture cible et la solution théorique sont par construction similaires. La question que nous soulevons dans cette section est de savoir si la seconde occupe une place privilégiée dans l'espace de critères, dont la topologie est « induite » par la première (cf. section 7.6). Nous montrons que la solution théorique est d'autant mieux placée dans l'espace des critères que le nombre de descripteurs est élevé.

8.4.1 Le front de Pareto contient-il la solution théorique ?

Le tableau 8.5 reporte les pourcentages des cas où la solution cible appartient au front de Pareto, en fonction du type de cible et du jeu de descripteurs. A chaque fois, ce pourcentage est à comparer avec le « random », c'est à dire la probabilité qu'une solution choisie au hasard appartienne au front. Cette probabilié est égale au rapport entre la taille du front et la taille de l'espace de recherche (soit l'ensemble des solutions consistantes avec les contraintes de hauteur, d'effectif et de capacité). Le tableau inférieur donne le rapport entre la fréquence observée et la probabilité due au hasard. Un rapport supérieur à 1 indique donc une performance supérieure au hasard.

L'interprétation de ces données est sujette à caution. Les principaux partiels résolus (MRP) apparaissent indéniablement comme le critère le plus déterminant, puisque la fréquence à laquelle la solution théorique est présente dans le front augmente significativement dès que

TAB. 8.5 – Fréquence à laquelle la solution théorique appartient au front de Pareto (tableau inférieur : rapport entre la fréquence observée et probabilité d'appartenance au front - un rapport supérieur à 1 indique une performance supérieure au hasard).

Fréquences observées

	mono-1	mono-2	mono-3	mono-4	poly-2	poly-3	poly-4
$k_{(1)}$	10.8 %	0.2 %	0 %	0 %	0.2 %	0 %	0 %
$k_{(2)}$	7.8 %	0.4 %	0 %	0 %	0 %	0 %	0 %
$k_{(3)}$	31.2 %	3.4 %	0.2 %	0 %	2.8 %	0 %	0 %
$k_{(12)}$	41.8 %	5.4 %	0.6 %	0 %	3.4 %	0.2 %	0 %
$k_{(13)}$	58.0 %	14.6 %	2.8 %	0.61 %	11.2 %	1.6 %	0 %
$k_{(23)}$	59.0 %	13.4 %	4.8 %	0.61 %	11.2 %	1.%	0 %
$k_{(123)}$	75.2 %	27.8 %	9.6 %	1.63 %	26.0 %	4.4 %	0.6 %

Proportion de la probabilité d'appartenance au front

	mono-1	mono-2	mono-3	mono-4	poly-2	poly-3	poly-4
$k_{(1)}$	8.8	5.7	-	-	11.4	-	-
$k_{(2)}$	6.3	11.4	-	-	-	-	-
$k_{(3)}$	25.4	97.2	145.8	-	159.7	-	-
$k_{(12)}$	7.9	19.6	40.3	-	22.5	56.9	-
$k_{(13)}$	9.5	18.5	6.3	759.3	25.1	168.4	-
$k_{(23)}$	9.4	16.6	10.6	682.3	24.7	103.5	-
$k_{(123)}$	6.8	17.9	17.4	674.5	29.2	139.5	610.1

ce critère est utilisé. En outre, le tableau inférieur montre que l'emploi simultané des trois critères est à préconiser pour les instances à quatre sons.

Cela dit, même si les performances par rapport au « random » semblent satisfaisantes en combinant les trois critères, la fréquence d'appartenance au front de Pareto diminue de façon drastique avec la taille du problème, et il y a fort à parier qu'étant donné la marge d'erreur intrinsèque aux fonctions d'agrégation et le caractère exclusivement spectro-harmonique des descripteurs utilisés, la solution théorique « sorte » systématiquement du front pour les problèmes de grande taille[3]. Que faut-il en conclure ? Peut-être simplement que mesurer une fréquence d'appartenance n'est sans doute pas pertinent pour évaluer notre approche. Nous préférons donc tenter de savoir si le front de Pareto, lorsqu'il n'inclut pas la solution théorique, contient au moins des solutions satisfaisantes pour l'utilisateur. Ce sera l'objet de la section 8.5. En attendant, nous cherchons à estimer à quelle « distance » du front se situe la solution théorique.

8.4.2 La solution théorique est-elle « éloignée » du front ?

Nous évaluons dans cette section la « distance » au sens de Pareto entre la solution théorique et le front. Nous calculons pour chaque instance de test et chaque combinaison de critères le nombre de solutions de l'espace de recherche qui dominent la solution cible. Plus ce nombre est faible, et plus la solution cible est proche du front au sens de la dominance de Pareto.

Les résultats de ce test sont reportés dans le tableau 8.6. La statistique r_{50} correspond au nombre médian de configurations dominant la solution théorique pour chaque classe de mixtures et chaque combinaison de critères. La mesure $dom_{\%}$ est le pourcentage moyen de solutions ne dominant pas la solution cible ; pour les solutions efficientes, on a : $dom_{\%} = 100$. Elle se calcule de la façon suivante :

$$dom_{\%} = 100\left(1 - \frac{\bar{r}}{\#E}\right) \tag{8.4}$$

où $\bar{r}$ est le rang moyen de la solution cible et $\#E$ la taille de l'espace de recherche.

Cette fois l'intérêt de l'approche multicritère nous semble plus évident. Quelle que soit la classe de mixtures considérée, le nombre de configurations dominant la solution théorique diminue quasi-systématiquement d'un facteur 10 dès que l'on passe de 2 à 3 critères, et d'un facteur 20 lorsqu'on passe du seul critère sur les MRP (le plus performant) à l'ensemble des trois critères. Même si le rang de la solution théorique reste relativement élevé pour les plus grandes instances, elle est en moyenne dominée par moins de 1% de l'espace. Nous sommes donc plutôt satisfaits de ces résultats au regard de la description limitée du timbre à laquelle nous nous restreignons.

Nous devons toutefois nous méfier des conséquences inévitables de la multidimensionnalité, évoquées au paragraphe 8.2.3 : la taille du front de Pareto augmente avec le nombre de critères. En grande dimension, les solutions ont naturellement tendance à être moins dominées. Les résultats du tableau 8.6 doivent donc être interprétés en connaissance de cause. Un test d'écoute présenté au paragraphe 8.5.3 nous confortera toutefois dans l'approche multicritère.

[3]Sans doute l'extension des critères à des descripteurs de variation temporelle, de bruit, de rugosité, etc., permettrait-elle de « ramener » la solution théorique dans le front.

TAB. 8.6 – Nombre médian de configurations dominant la solution théorique ($dom_{\%}$: pourcentage de l'espace ne dominant pas la solution théorique)

Mixtures monophoniques

	mono-1		mono-2		mono-3		mono-4	
	r_{50}	$dom_{\%}$	r_{50}	$dom_{\%}$	r_{50}	$dom_{\%}$	r_{50}	$dom_{\%}$
k_1	9	84	270	83	7149	81	1215486	79
k_2	10	83	277	84	6611	84	982176	82
k_3	4	88	110	89	2059	88	216987	87
k_{12}	8	86	233	83	6959	81	1064636	80
k_{13}	6	85	201	84	6593	82	879714	80
k_{23}	7	85	192	85	4524	85	661301	83
k_{123}	1	97	5	99	104	99	11114	99

Mixtures polyphoniques

	poly-2		poly-3		poly-4	
	r_{50}	$dom_{\%}$	r_{50}	$dom_{\%}$	r_{50}	$dom_{\%}$
k_1	582	83	38346	83	1909090	83
k_2	624	83	38493	83	1852530	83
k_3	157	93	5591	94	203606	95
k_{12}	542	84	34644	84	1656381	84
k_{13}	331	87	21212	86	864101	88
k_{23}	324	87	17476	88	708116	89
k_{123}	7	99	239	99	8772	99

8.5 Qualité des solutions du front de Pareto

Dans cette section, nous montrons que la similartié de certaines solutions du front avec la solution théorique augmente avec le nombre de critères, et que l'utilisation simultanée des trois critères garantit une pertinence plus élevée des solutions efficientes. Nous introduisons au paragraphe 8.5.2 trois heuristiques permettant d'identifier des solutions du front a priori pertinentes. L'une de ces heuristiques utilise la notion de *norme induite* par une configuration. Nous l'explicitons au paragraphe suivant.

8.5.1 Norme induite par une configuration

Etant donné un point x de l'espace de critères dans un problème de minimisation, les points dominant x se situent dans l'hyperparallélogramme de diagonale $[0; x]$ (voir figure 5.2). On se propose ici d'identifier cette région de l'espace par une propriété métrique.

En considérant l'espace de N critères comme l'hyperquadrant positif d'un espace métrique de dimension N, un hyperparallélogramme de diagonale $[0; x]$ n'est rien d'autre qu'une boule engendrée par une norme de Tchebycheff pondérée.

Définition 8.2. *Soit Λ la partie de $[0; 1]^N$ telle que :*

$$\forall \lambda = (\lambda_1, \ldots, \lambda_N) \in \Lambda,\ \sum_i \lambda_i = 1 \tag{8.5}$$

Soit $\lambda = (\lambda_1, \ldots, \lambda_N) \in \Lambda$. La norme définie sur un espace de critères C, de dimension N, par :

$$\forall x \in C,\ \|x\|_\lambda = \max_i\ \lambda_i x_i \tag{8.6}$$

*est appelée **norme de Tchebycheff pondérée** par le jeu de poids $\lambda = (\lambda_1, \ldots, \lambda_N)$.*

Définition 8.3. *Soit $x = (x_1, \ldots, x_N) \in C$ un point de l'espace des critères correspondant à une configuration $K \in E$ de l'espace de recherche. On appellera **norme induite** par x (ou par K) et on notera $\|\cdot\|_x$ (ou $\|\cdot\|_K$) la norme de Tchebycheff vérifiant :*

$$\forall (i, j) \in \{1; N\}^2,\ \lambda_i x_i = \lambda_j x_j \tag{8.7}$$

Remarque 8.2. *Si toutes les coordonnées de x sont strictement positives, alors l'unicité de la norme induite par x est assurée par l'équation 8.5. Le calcul des poids associés à la norme induite est donné en annexe B.*

Nous pouvons alors énoncer les deux propriétés essentielles de la norme induite :

Propriété 8.1. *Soit $x \in C$ un point de l'espace des critères et $\|\cdot\|_x$ la norme induite par x sur C. On a alors :*

$$\forall y \in C,\ y \preceq x \Leftrightarrow \|\ y\|_x \leq \|x\|_x \tag{8.8}$$

Démonstration. Par construction de la norme induite, on a :

$$\forall i,\ \|x\|_x = \lambda_i x_i$$

Par conséquent :

$$\begin{aligned} \| y\|_x \leq \|x\|_x &\Leftrightarrow \max_i \lambda_i y_i \leq \|x\|_x \\ &\Leftrightarrow \forall i,\ \lambda_i y_i \leq \|x\|_x \\ &\Leftrightarrow \forall i,\ \lambda_i y_i \leq \lambda_i x_i \\ &\Leftrightarrow \forall i,\ y_i \leq x_i \\ &\Leftrightarrow y \preceq x \end{aligned}$$

□

La propriété 8.1 est fondamentale. C'est elle qui nous permettra, au chapitre 9, de transformer localement un problème multicritère en un problème uni-critère.

Propriété 8.2. *Soit $x \in C$ un point de l'espace des critères et $\| \cdot \|_x$ la norme induite par x sur C. $\| \cdot \|_x$ vérifie :*

$$\forall \lambda \in \Lambda, \| x\|_\lambda \geq \|x\|_x \tag{8.9}$$

Démonstration. Soit $\lambda = (\lambda_1, \ldots, \lambda_N)$ le jeu de poids associé à $\| \cdot \|_x$. Soit $l = (l_1, \ldots, l_N) \in \Lambda$ et $\| \cdot \|_l$ la norme de Tchebycheff associée à l. La condition 8.5 impose :

$$\exists j \in \{1; N\},\ l_j \geq \lambda_j$$

On a alors :

$$\begin{aligned} &\Rightarrow l_j x_j \geq \lambda_j x_j \\ &\Rightarrow l_j x_j \geq \|x\|_x \\ &\Rightarrow \max_i l_i x_i \geq \|x\|_x \\ &\Rightarrow \|x\|_l \geq \|x\|_x \end{aligned}$$

□

En d'autres termes, $\| \cdot \|_x$ est, de toutes les normes de Tchebycheff pondérées, celle qui attribue à x la plus petite valeur, donc celle qui permet d'obtenir le meilleur rang pour x. Cette propriété sera pour nous d'un grand intérêt dans l'inférence des préférences de l'utilisateur évoquée au paragraphe 7.2.4. En effet, si le compositeur émet une préférence pour une configuration K de l'espace de recherche, alors la norme induite par K est celle qui reflète le mieux cette préférence. Le calcul des poids associés à $\| \cdot \|_K$ peut donc nous renseigner sur l'importance relative des critères perceptifs qui a conduit au choix de K.

Remarque 8.3. *La démonstration de la propriété 5.1 établissant l'équivalence entre un problème multicritère et un ensemble de problèmes mono-critère (voir section 5.7) est alors immédiate. Dans le sens direct, il suffit de choisir $\| \cdot \|_\lambda = \| \cdot \|_x$ et d'invoquer la propriété 8.1. La réciproque est évidente par définition de la norme de Tchebycheff.*

8.5.2 Trois heuristiques

Nous cherchons maintenant à savoir si au moins une solution du front théorique peut être considérée comme « satisfaisante », c'est-à-dire à la fois proche de la solution cible sur le plan sonore (dans l'espace des descripteurs) et sur le plan symbolique (dans l'espace de décisions). Nous définissons dans ce but trois heuristiques pour choisir automatiquement dans le front théorique trois solutions supposés pertinentes :

Heuristique H_e : Optimum de Salukwadze. La première idée qui vient à l'esprit consiste à choisir le point du front le plus proche de la solution théorique au sens de la norme euclidienne dans l'espace des critères. Ce point, introduit par Salukwadze dans [Sal79], est appelé « optimum de Salukwadze ». Bronstein [BBBK07] l'a récemment utilisé pour convertir des distances multi-valuées en distances mono-valuées. Nous le notons K_e. Nous verrons par la suite que les solutions découvertes par cette heuristique sont rarement pertinentes.

Heuristique H_λ : Classement selon la norme induite. Une autre manière de choisir une solution du front consiste à « tirer un rayon » depuis l'origine du repère (dans l'espace des critères) vers la solution théorique et de choisir la solution du front à l'endroit où le rayon intercepte ce dernier. L'hypothèse sous-jacente est que les descripteurs de la solution théorique, bien qu'ayant été mal évalués (sinon le front de Pareto la contiendrait), peuvent néanmoins nous informer sur une direction privilégiée de l'espace. Cette direction est liée aux préférences implicites aux « choix » de la solution théorique comme meilleure configuration (voir paragraphe 8.5.1). Nous faisons l'hypothèse qu'elle permet d'identifier une « bonne » solution dans le front. En pratique, nous choisissons la solution du front qui minimise la norme de Tchebycheff induite par la solution théorique (voir annexe B). Nous la notons K_λ.

Heuristique H_s : Proximité symbolique. Enfin, la troisième solution retenue est celle qui minimise la *distance symbolique* avec la solution théorique. Cette distance est calculée sur des attributs hiérarchiquement ordonnés selon des niveaux de dissemblances décroissants. Pour mesurer la distance entre deux sons, on commence par comparer leurs hauteurs, puis, en cas d'égalité, leur dynamiques, leurs familles d'instruments, leurs instruments en cas de nouvelle égalité, et enfin leurs modes de jeu. En outre, deux sons de hauteurs différentes seront toujours distants d'une valeur supérieure à la distance entre deux sons de même hauteur, la propriété se transmettant par récurrence aux niveaux de dissemblance inférieurs. Pour comparer deux configurations, nous procédons tout d'abord à un alignement (au sens d'une distance d'édition) de leurs éléments respectifs puis sommons les distances symboliques entre les éléments appairés. Le point obtenu par cette méthode sera noté K_s.

La figure 8.4 illustre ces trois heuristiques dans un cas bi-critères. La distance symbolique étant calculée dans l'espace de décisions, K_s a donc été placé aléatoirement dans l'espace des critères sur cette figure. Pour chaque classe de problèmes et chaque combinaison de critères, nous identifions dans les fronts de Pareto les solutions K_λ, K_e et K_s à l'aide des heuristiques H_λ, H_e et H_s, et évaluons leur qualité à l'aide des mesures suivantes :

MAD **(mean audio distance)** Distance euclidienne moyenne dans l'espace des descripteurs entre la solution théorique et la solution du front trouvée par l'heuristique. Les distances associées à chaque descripteur sont les fonctions de comparaison. Ainsi que mentionné à la partie 7.6, il est impossible d'agréger ces distances sans connaître les préférences de l'utilisateur. Faute de mieux, nous exprimons donc la statistique MAD comme la norme $\mathcal{L}_2$ du vecteur des distances associées à chaque descripteur[4].

[4]Pour calculer la statistique MAD nous utilisons systématiquement *toutes* les fonctions d'agrégation, et ce quels que soient les critères utilisés. La distance ainsi obtenue est alors une distance euclidienne sur l'espace des descripteurs normalisés, dont la dimension est indépendante du nombre de critères choisis, ce qui permet en outre des comparaisons entre des espaces de critères de dimensions différentes.

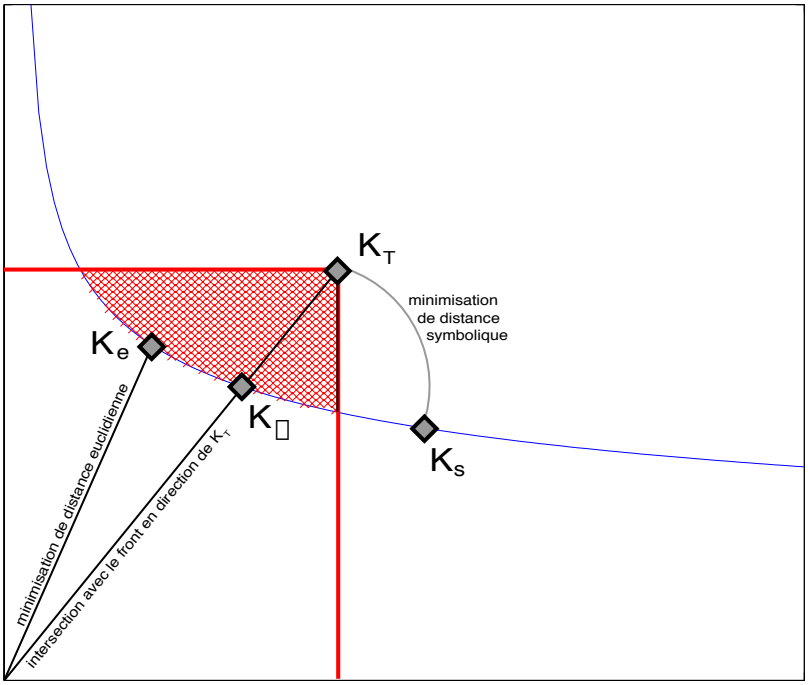

FIG. 8.4 – Trois heuristiques pour trouver des solutions pertinentes dans le front de Pareto lorsque ce dernier ne contient pas la solution théorique K_T. K_λ (heuristique H_λ) est obtenue par classement du front selon la norme de Tchebycheff induite par K_T ; K_e (heuristique H_e) par minimisation de la norme euclidienne, et K_s (heuristique H_s) par minimisation de la distance symbolique entre K_T et le front (la position de K_s n'est pas significative sur cette figure).

PFM **(perfect family match)** Fréquence à laquelle la solution du front trouvée par l'heuristique comporte *exactement* les mêmes familles d'instruments que la solution théorique.

MFM **(mean family match)** Nombre moyen de familles communes à la solution du front trouvée par l'heuristique et à la solution théorique.

PIM **(perfect instrument match)** Fréquence à laquelle la solution du front trouvée par l'heuristique comporte *exactement* les mêmes instruments que la solution théorique.

MIM **(mean instrument match)** Nombre moyen d'instruments communs à solution du front trouvée par l'heuristique et à la solution théorique.

Nous reportons dans les tableaux 8.7 et 8.8 les mesures moyennes relatives aux mixtures monophoniques et polyphoniques à quatre sons.

Ces résultats confirment tout d'abord que les principaux partiels résolus (MRP) constituent le critère le plus pertinent dans l'évaluation des configurations, puisque la distance moyenne dans l'espace des descripteurs et les mesures de proximité symbolique entre la solution du front trouvée par l'heuristique et la solution théorique sont systématiquement plus faibles quand les partiels sont utilisés. Mais le résultat principal est le suivant : aussi bien pour les cibles monophoniques que polyphoniques, la « qualité » des solutions trouvées par les heuristiques H_λ, H_e et H_s augmente avec le nombre de critères ; la distance audio moyenne et les valeurs de similarité symbolique atteignent leurs optima lorsque les trois critères sont considérés simultanément.

Toutefois, si cette tendance est globalement observée pour les trois heuristiques, elle est moins marquée pour la seconde d'entre elles, qui choisit la solution du front la plus proche de l'origine au sens de la norme euclidienne. Ses performances sont systématiquement inférieures aux deux autres heuristiques, et à trois critères ne sont que de peu supérieures aux cas mono- ou bi-critères. L'optimum de Salukwadze choisi par cette heuristique est la solution d'un problème d'optimisation où tous les critères sont agrégés par une norme euclidienne. La qualité moindre de cette solution prouve donc la supériorité d'une véritable approche multicritère par rapport à ce que Talbi [Tal99] appelle la transformation vers l'uni-objectif.

Les mesures employées pour évaluer la qualité des solutions trouvées par les heuristiques se divisent en deux catégories : l'évaluation « symbolique » (PFM, MFM, PIM et MIM) et l'évaluation « audio » (MAD). Les solutions K_s obtiennent toujours de meilleurs résultats sur le critère symbolique et les solutions K_λ sont toujours plus performantes sur le critère audio, sans qu'aucune catégorie de solutions ne surpasse l'autre sur les critères audio et symboliques simultanément. Il est donc impossible a priori de décider que l'une des heuristiques est meilleure que l'autre. En revanche les solutions découvertes par l'heuristique H_e sont systématiquement battues sur les critères audio et symboliques. Il est donc permis de douter de la pertinence de H_e, ce qui sera d'ailleurs confirmé par la suite.

Nous avons donc prouvé d'une part que l'utilisation simultanée de trois critères spectro-harmoniques pour l'évaluation des configurations garantit des solutions efficientes plus proches de la solution théorique, d'autre part que ces trois critères doivent être considérés conjointement dans une approche multicritère, et non agrégés en une seule valeur à optimiser.

Il reste maintenant à savoir si les solutions trouvées par les heuristiques représentent véritablement — d'un point de vue perceptif — des propositions intéressantes pour l'utilisateur.

TAB. 8.7 – Statistiques MAD (mean audio distance), PFM (perfect family match), MFM (mean family match), PIM (perfect instrument match), MIM (mean instrument match) pour les trois heuristiques H_λ, H_e et H_s et pour toutes les combinaisons de critères dans le cas de mixtures monophoniques à 4 sons.

Heuristique H_λ

	MAD	PFM	MFM	PIM	MIM
$k_{(1)}$	0.217	16.9 %	2.89	5.1 %	2.22
$k_{(2)}$	0.250	15.5 %	2.87	6.3 %	2.21
$k_{(3)}$	0.243	23.9 %	3.02	10.0 %	2.41
$k_{(12)}$	0.123	18.6 %	2.90	5.9 %	2.24
$k_{(13)}$	0.081	23.3 %	3.00	10.0 %	2.33
$k_{(23)}$	0.075	24.3 %	3.05	8.6 %	2.41
$k_{(123)}$	0.046	26.9 %	3.08	12.9 %	2.49

Heuristique H_e

	MAD	PFM	MFM	PIM	MIM
$k_{(1)}$	0.217	16.9 %	2.89	5.1 %	2.22
$k_{(2)}$	0.250	15.5 %	2.87	6.3 %	2.21
$k_{(3)}$	0.243	23.9 %	3.02	10.0 %	2.41
$k_{(12)}$	0.183	9.8 %	2.67	4.7 %	1.97
$k_{(13)}$	0.164	14.3 %	2.77	6.5 %	2.15
$k_{(23)}$	0.167	14.5 %	2.80	6.3 %	2.16
$k_{(123)}$	0.168	9.8 %	2.59	6.1 %	1.95

Heuristique H_s

	MAD	PFM	MFM	PIM	MIM
$k_{(1)}$	0.217	16.9 %	2.89	5.1 %	2.22
$k_{(2)}$	0.250	15.5 %	2.87	6.3 %	2.21
$k_{(3)}$	0.243	23.9 %	3.02	10.0 %	2.41
$k_{(12)}$	0.143	36.7 %	3.26	12.9 %	2.60
$k_{(13)}$	0.122	53.1 %	3.44	26.3 %	2.89
$k_{(23)}$	0.126	53.5 %	3.46	25.1 %	2.91
$k_{(123)}$	0.094	72.0 %	3.70	41.2 %	3.22

TAB. 8.8 – Statistiques MAD (mean audio distance), PFM (perfect family match), MFM (mean family match), PIM (perfect instrument match), MIM (mean instrument match) pour les trois heuristiques H_λ, H_e et H_s et pour toutes les combinaisons de critères dans le cas de mixtures polyphoniques à 4 sons.

Heuristique H_λ

	MAD	PFM	MFM	PIM	MIM
$k_{(1)}$	0.378	3.4 %	1.69	0.0 %	0.63
$k_{(2)}$	0.421	2.8 %	1.63	0.2 %	0.61
$k_{(3)}$	0.299	6.0 %	2.04	1.4 %	1.03
$k_{(12)}$	0.310	3.4 %	1.76	0.4 %	0.71
$k_{(13)}$	0.133	6.4 %	1.96	1.6 %	0.99
$k_{(23)}$	0.133	7.2 %	2.00	0.4 %	0.99
$k_{(123)}$	0.084	7.8 %	2.06	2.6 %	1.08

Heuristique H_e

	MAD	PFM	MFM	PIM	MIM
$k_{(1)}$	0.378	3.4 %	1.69	0.0 %	0.63
$k_{(2)}$	0.421	2.8 %	1.63	0.2 %	0.61
$k_{(3)}$	0.299	6.0 %	2.04	1.4 %	1.03
$k_{(12)}$	0.305	5.0 %	1.87	0.2 %	0.95
$k_{(13)}$	0.190	4.8 %	2.09	1.2 %	1.21
$k_{(23)}$	0.216	6.0 %	2.09	1.4 %	1.28
$k_{(123)}$	0.239	4.4 %	1.87	1.4 %	1.07

Heuristique H_s

	MAD	PFM	MFM	PIM	MIM
$k_{(1)}$	0.378	3.4 %	1.69	0.0 %	0.63
$k_{(2)}$	0.421	2.8 %	1.63	0.2 %	0.61
$k_{(3)}$	0.299	6.0 %	2.04	1.4 %	1.03
$k_{(12)}$	0.279	10.6 %	2.31	1.0 %	1.07
$k_{(13)}$	0.154	16.8 %	2.59	3.4 %	1.51
$k_{(23)}$	0.175	19.0 %	2.62	4.0 %	1.54
$k_{(123)}$	0.114	36.8 %	3.11	11.2 %	2.03

TAB. 8.9 – Test d'écoute 1 : Pour chaque combinaison de critères, fréquence à laquelle la meilleure solution du front de parmi K_λ, K_e et K_s est jugée satisfaisante.

	mono-4	poly-4
$k_{(1)}$	10 %	10%
$k_{(2)}$	10 %	0%
$k_{(3)}$	20 %	10%
$k_{(12)}$	25 %	20%
$k_{(13)}$	40 %	40%
$k_{(23)}$	75 %	45%
$k_{(123)}$	80 %	60%

8.5.3 Tests d'écoute

Ayant épuisé la connaissance que nous pouvons tirer des descripteurs et des attributs dans l'évaluation de la qualité des solutions K_λ, K_e et K_s, le seul moyen d'aller plus avant dans la validation de notre approche est un test perceptif. Nous avons mis en place deux tests qui visent d'une part à confirmer la supériorité de l'approche multicritère pour le problème de l'orchestration, d'autre part à vérifier la qualité des solutions trouvées par les heuristiques introduites à la section précédente.

Ces tests n'ont pas la prétention d'être de véritables tests d'écoute réalisés selon un protocole scientifique rigoureux. Leur but n'est pas de découvrir un mécanisme perceptif ou cognitif particulier, mais seulement de confirmer une intuition. Ils été subis par un petit nombre de sujets, et leurs résultats sont à interpréter dans un sens exclusivement qualitatif.

Test 1 : validation de l'approche multicritère

Pour ce premier test nous avons, pour chaque combinaison de critères, généré les trois solutions trouvées par les heuristiques H_λ, H_e et H_s, et ce pour 40 instances de test choisies aléatoirement parmi les mixtures monophoniques et polyphoniques à 4 sons. Pour chaque combinaison de critères nous avons retenu parmi les solutions K_λ, K_e et K_s celle que nous jugions la plus proche de la cible d'un point de vue perceptif. Nous avons ainsi obtenu 7 propositions par instance de test, correspondant aux 7 combinaisons de critères du tableau 8.2. Pour chaque instance, les sujets devaient choisir la ou les (éventuellement aucune) solutions qui lui semblaient la ou les plus proches de la cible sonore. Le but de ce test est de montrer la supériorité de l'approche multicritère.

Les résultats sont regroupés dans le tableau 8.9. Comme on pouvait s'y attendre, les similarités sont maximales lorsque les trois critères sont considérés. Ces chiffres confirment les statistiques de proximité audio et symboliques des tableaux 8.7 et 8.8.

Test 2 : évaluation des heuristiques H_λ, H_e et H_s

Dans ce second test ont été générées les solutions trouvées par les heuristiques H_λ, H_e et H_s, uniquement dans un cadre tri-critères, et ce pour 40 instances de test choisies aléatoirement parmi les mixtures monophoniques et polyphoniques à 4 sons. Pour chaque instance, les sujets devaient choisir la ou les (éventuellement aucune) solutions parmi les trois qui lui semblaient

TAB. 8.10 – Test d'écoute 2 : Les trois critères étant considérés simultanément, fréquence à la quelle chacune des solutions K_λ, K_e et K_s est jugée satisfaisante. *Proche* : fréquence à laquelle au moins une des trois solutions est proche de la cible. *Très proche* : fréquence à laquelle au moins une des trois solutions est très proche à la cible.

	mono-4	poly-4
H_λ	55 %	30%
H_e	25 %	15%
H_s	65 %	45%
Proche	100 %	80%
Très proche	75 %	35%

les plus proches de la cible sonore, puis évaluer cette similarité sur une échelle très simple : peu similaire, proche ou très proche. Le but de ce test est d'évaluer la pertinence des solutions du front en multicritère.

Ce test prouve la supériorité des heuristiques H_λ et H_s sur H_e. Là encore, nous retrouvons les résultats du paragraphe précédent. La solution K_e, qui résulte d'une transformation du problème vers l'uni-objectif, est la moins pertinente. Par ailleurs, notons qu'au moins l'une des solutions proposées par ces heuristiques est jugée proche de la cible dans une grande proportion de cas (100 % des cas en monophonique, 80 % en polyphonique).

Nous n'irons pas plus loin dans l'exploitation de ces résultats. Bien que très informels et réalisés sur un petit nombre de sujets, ces tests nous semblent suffisants pour justifier aussi bien l'approche multicritère que la qualité des solutions trouvées.

8.5.4 Contributions relatives des critères

Dorénavant convaincus que de meilleures solutions peuvent être trouvées en considérant simultanément les trois critères dans le processus d'optimisation, nous nous proposons ici de justifier a posteriori notre approche en évaluant pour différentes instances de test l'« information » apportée par chacun des critères. Le tableau 8.10 indiquant que les heuristiques H_λ et H_s permettent d'identifier dans le front de Pareto des solutions pertinentes, nous calculons donc, pour les fronts à trois critères et pour chaque instance de test, les jeux de poids associés à la norme induite (voir section 8.5.1 et annexe B) par les solutions K_λ et K_s. Le poids associé à un critère peut être interprété comme l'importance de ce dernier dans le choix de la solution. Ces poids calculés, nous estimons, sur la base des fréquences auxquelles ils surviennent, les probabilités des événements suivants :

- A : $w_{sc} > w_{MRP}$ Le poids associé au centroïde spectral est supérieur au poids associé aux partiels.
- B : $w_{sc} > 0.5$ Le poids associé au centroïde spectral est supérieur à 0.5 (son rôle est déterminant dans le choix de la solution).
- C : $w_{ss} > w_{MRP}$ Le poids associé à l'étendue spectrale est supérieur au poids associé aux partiels.

- $D : w_{ss} > 0.5$ Le poids associé à l'étendue spectrale supérieur à 0.5 (son rôle est déterminant dans le choix de la solution).
- $E : w_{MRP} < 0.5$ Le poids associé aux partiels est inférieur à 0.5 (le rôle des moments spectraux est déterminant dans le choix de la solution).

A partir des probabilités observées, nous pouvons déduire les intervalles de confiance pour les probabilités réelles de ces événements en inversant un test de Student. Ce test permet de confronter une probabilité observée $\tilde{P}(X) = p$ à une probabilité théorique $P(X) = p_t$. Sous l'hypothèse nulle $H_0 : P(X) = p_t$, alors la statistique :

$$t = \frac{|p - p_t|}{\sqrt{p(1-p)}}\sqrt{n} \tag{8.10}$$

suit une loi normale centrée réduite. Etant donné une probabilité observée $\tilde{P}(X) = p$, nous en déduisons l'intervalle de confiance au niveau α pour la probabilité théorique :

$$p_t = p \pm t_\alpha \sqrt{\frac{p(1-p)}{n}} \tag{8.11}$$

Les bornes inférieures des intervalles de confiance au niveau 0.01 sont reportées dans le tableau 8.11. Globalement, les probabilités des événements A, C, D, E et dans une moindre mesure B sont loin d'être négligeables. Les cas sont donc nombreux dans lequels les partiels n'interviennent pas de façon majeure dans l'évaluation des solutions. Nous justifions ainsi l'emploi simultané de plusieurs critères spectro-harmoniques. Une analyse plus fine mérite toutefois de distinguer les heuristiques employées.

Lorsque la solution de l'heuristique H_λ est retenue, les probabilités des événements sont indépendantes de la taille du problème et ne dépendent que du caractère monophonique ou polyphonique des cibles. Si la solution obtenue par l'heuristique H_λ est la plus pertinente, cela signifie que les coordonnées de la solution théorique dans l'espace des critères donnent la « bonne direction » d'optimisation (voir figure 8.4). Autrement dit, les erreurs dans l'estimation des descripteurs de la solution théorique sont toutes du même ordre de grandeur, a fortiori faible. En conséquence, il y a fort a parier que cette dernière est relativement proche de la frontière de Pareto. Dans ce cas, si l'estimation des descripteurs est correcte pour tous les critères, il est normal que les partiels apportent l'essentiel de l'information dans 60 % des cas, comme en atteste les probabilités moyennes de l'événement E.

En revanche, lorsque l'heuristique H_s donne une meilleure solution, alors la position de la solution cible dans l'espace des critères ne donne pas la « bonne direction » d'optimisation. Nous pensons que ce « faux cap » est dû à des erreurs d'estimation différentes selon les descripteurs. La difficulté du calcul des principaux partiels résolus (notamment à cause des phénomènes de masquage) permet de faire l'hypothèse que les MRP de la solution théorique ont été mal évalués. En conséquence, leur importance relative dans l'identification d'une solution pertinente dans le front est diminuée. C'est bien ce que confirme la lecture du tableau inférieur, en particulier pour les mixtures polyphoniques, pour lesquelles la probabilité que le poids associé aux partiels soit inférieur à 0.5 augmente fortement avec la taille du problème. Dans les cas où K_s est une solution meilleure que K_λ, les MRP ne permettent donc pas de l'identifier, et sont peu à peu relayés par les moments spectraux qui interviennent alors à des fins de robustesse dans l'évaluation des solutions. Pour les cibles monophoniques toutefois, l'importance des partiels augmente avec la taille du problème, sans doute car leur évaluation est plus aisée (les phénomènes de masquage interviennent peu dans ces cas).

TAB. 8.11 – Bornes inférieures des intervalles de confiance au niveau 0.01 pour les probabilités théoriques des événements A : $w_{sc} > w_{MRP}$, B : $w_{sc} > 0.5$, C : $w_{ss} > w_{MRP}$, D : $w_{ss} > 0.5$ et E : $w_{MRP} < 0.5$.

Heuristique H_λ

	$P_{inf}(A)$	$P_{inf}(B)$	$P_{inf}(C)$	$P_{inf}(D)$	$P_{inf}(E)$
mono-1	0.209	0.098	0.237	0.114	0.419
mono-2	0.213	0.101	0.228	0.110	0.398
mono-3	0.183	0.072	0.240	0.109	0.405
mono-4	0.174	0.056	0.225	0.107	0.416
poly-2	0.152	0.055	0.176	0.082	0.325
poly-3	0.164	0.075	0.155	0.056	0.308
poly-4	0.163	0.055	0.201	0.094	0.333

Heuristique H_s

	$P_{inf}(A)$	$P_{inf}(B)$	$P_{inf}(C)$	$P_{inf}(D)$	$P_{inf}(E)$
mono-1	0.262	0.154	0.270	0.126	0.476
mono-2	0.333	0.179	0.343	0.215	0.502
mono-3	0.377	0.241	0.407	0.255	0.606
mono-4	0.174	0.056	0.225	0.107	0.416
poly-2	0.305	0.180	0.335	0.199	0.509
poly-3	0.454	0.298	0.480	0.289	0.695
poly-4	0.590	0.382	0.541	0.327	0.808

Conclusion

Nous concluons ici quant au caractère « bien posé » du problème de l'orchestration formulé comme une tâche d'optimisation multicritère avec trois critères spectro-harmoniques a priori non indépendants : le centroïde spectral linéaire, l'étendue spectrale linéaire, et les principaux partiels résolus. Afin de prouver la validité de cette formulation, nous avons généré 3500 mixtures monophoniques et polyphoniques comprenant entre 1 et 4 sons. Chacune de ces mixtures a été considérée comme la cible d'un problème d'orchestration particulier, avec contraintes de hauteurs, de capacité et de cardinalité. Pour chaque problème, le front de Pareto (les solutions du problème multicritère) a été extrait après un parcours exhaustif de l'espace de recherche, et ce pour différentes combinaisons de critères. L'enjeu était de vérifier si les solutions théoriques de chaque problème étaient pertinentes, à savoir « proches » de la mixture cible. Cette étude à notamment montré :

- qu'augmenter le nombre de critères augmente la probabilité de trouver la solution théorique dans le front de Pareto ;
- qu'augmenter le nombre de critères diminue la portion de l'espace dominant la solution théorique ;
- que les solutions du front théorique sont d'autant plus proches de la cible que le nombre de critères est élevé, et ce aussi bien en termes symboliques que descriptifs et perceptifs ;
- enfin, que malgré une sélectivité inégale des critères spectraux, les critères a priori mineurs peuvent apporter une information prévalante dans l'identification d'une « bonne » solution, et que l'emploi simultané de critères non indépendants permet souvent d'évaluer de manière plus robuste les solutions.

Nous nous autorisons donc désormais à poser le problème de l'orchestration comme une tache d'optimisation combinatoire multicritère. Convaincus que les solutions des fronts de Pareto sont pertinentes pour le compositeur, nous pouvons dès lors nous consacrer à la recherche d'algorithmes permettant de découvrir ces solutions en un temps raisonnable.

Chapitre 9

Algorithme d'orchestration

Modélisation « orchestre » pour la gestion des contraintes de capacité
Crossover et mutation pour l'orchestration
Poids aléatoires, fonctions scalarisantes de Tchebycheff
Directions d'optimisation et préférences d'écoute

Nous présentons dans ce chapitre un algorithme génétique adapté au problème de l'orchestration. Notre choix s'est porté sur ce type d'approche — à l'exclusion de métaheuristiques plus récentes — pour au moins trois raisons :

1. Les AGs font l'objet d'une littérature abondante et on été employés avec succès dans des situations très variées. En outre, la théorisation des AGs constitue, depuis le début des années 90, un courant de recherche majeur (voir notamment Cerf [Cer94] et Amiot & Durand [AD05]).
2. Les opérateurs de croisement (voir paragraphe 5.5) répondent à une vision intuitive de la recherche de combinaisons, en manipulant les sous-ensembles. Ils permettent ainsi de générer, à partir d'orchestrations existantes, de nouvelles combinaisons en « mélangeant » les « plans sonores ». Soient par exemples deux orchestrations $\{A, B\}$ et $\{C, D\}$ affichant de « bonnes valeurs » de fitness, où A et C (respectivement B et D) désignent des mélanges impliquant les mêmes groupes d'instruments. Il semble alors naturel de « tester » les configurations $\{A, D\}$ et $\{C, B\}$, déductibles des premières par application d'un crossover. Si seule la configuration $\{A, D\}$ est bien évaluée, on peut alors en conclure que les plans sonores B et C ne sont pas d'un grand intérêt. En revanche, émergent et persistent au cours de la recherche les sous-ensembles localement optimaux. En outre, le compositeur satisfait avec une orchestration $\{a, b, c, d\}$ mais convaincu de pouvoir « mieux faire » cherchera naturellement dans un voisinage : $\{a', b, c, d\}$, $\{a, b', c, d\}$, etc. Dans le langage des AGs, il réalisera donc une mutation.
3. Enfin, la non-linéarité et la non-différentiabilité de certaines fonctions d'agrégation (voir annexe A) ne permettent pas d'avoir recours à des méthodes traditionnelles d'optimisation telles que la descente de gradient par exemple.

9.1 Un problème de sac à dos ?

Nous avons vu, à la section 7.6, que l'espace de recherche E du problème de l'orchestration était isomorphe à $\{0; 1\}^{N_s}$, où N_s est la taille de la connaissance instrumentale. Il nous faut

donc trouver un sous-ensemble I de $\{0;1\}^{N_s}$ tel que la mixture composée des éléments d'indices $i \in I$ maximise la similarité perceptive avec la cible à orchestrer, et ce dans la limite des capacités instrumentales.

Ainsi formulé, le problème de l'orchestration s'apparente fort à un problème de sac à dos binaire [MT90] (*Binary Knapsack Problem — KP-O/1*), dans lequel il s'agit de remplir un sac à l'aide d'objets de poids (p_i) et d'utilités (u_i) en maximisant l'utilité totale, sans excéder la capacité C du sac :

$$\text{(KP-0/1)} \begin{cases} \max & z(x) = \sum\ x_i u_i \\ \text{s.t.} & x_i \in \{0;1\} \\ & \sum\ x_i p_i \leq C \end{cases} \tag{9.1}$$

Si la similitude apparente entre les deux problèmes[1] laisse supposer le caractère $\mathcal{NP}$ de l'orchestration telle que définie en 7.4 (le problème du sac-à-dos étant lui-même $\mathcal{NP}$), elle ne permet pas en revanche de ré-utiliser pour la résolution du premier les méthodes propres au second. Trois raisons à cela :

1. Dans le cas de l'orchestration, l'utilité d'un élément dépend des éléments déjà présents dans la configuration. L'ajout d'un trombone « brassy » n'a par exemple pas le même effet dans un alliage mezzo-fote de cordes que dans un accord de cuivres joué fortissimo.
2. L'utilité d'une configuration ne peut pas être définie comme la somme des utilités de ses composantes. Pour s'en convaincre, remarquons simplement que l'ajout d'un élément dans une configuration peut très bien en dégrader la qualité.
3. Les contraintes instrumentales dans le cas de l'orchestration ne se ramènent pas à une unique capacité C, mais à un vecteur de capacités dont chaque élément représente la quantité de chaque instrument dans l'orchestre considéré.

Notons toutefois qu'il existe une variante du problème du sac à dos, connue sous le nom de *Multiple Container Knapsack Problem*, dans laquelle le sac est divisé en plusieurs conteneurs de capacités variables. Nous tirons parti de la modélisation de ce problème dans la section suivante.

9.2 Modélisation et encodage

Soit $(s_i)_{1\leq i\leq N_s}$ la connaissance instrumentale du système dont une orchestration représente un sous-ensemble. Soit $(c_j)_{1\leq j\leq N_I}$ le vecteur des effectifs instrumentaux, où N_I est le nombre total d'instruments présents dans (s_i), et c_j le nombre d'occurrences de l'instrument j dans l'orchestre. Soit D_j la partie de $\{1,...,N_s\}$ associée à l'instrument j, i.e. $(s_i)_{i\in D_j}$ sont les sons de la base associés à l'instrument j. Soit enfin R la matrice des allocations instrumentales, définie par :

$$\forall i \in \{1,...,N_s\}, \forall j \in \{1,...,N_I\}, R_{ji} = \begin{cases} 1 \text{ si } i \in D_j \\ 0 \text{ sinon} \end{cases}$$

[1] Le caractère uni-critère de problème 9.1, en regard de l'aspect multicritère de l'orchestration, ne doit pas remettre en cause cette similitude. Il existe d'ailleurs une formulation multicritère du problème du sac à dos.

Le problème d'orchestration multicritère (POMC) se formule alors comme un problème de minimisation sous contraintes linéaires :

$$\text{(POMC)} \begin{cases} \min & D_T^{d_k}(x) = D_T^{d_k}(x_1, \ldots, x_{N_s}) \\ & k = 1, \ldots, K \\ \text{s.t.} & x_i \in \{0; 1\} \\ & R \cdot x \leq C \end{cases} \tag{9.2}$$

Nous rappelons que $D_T^{d_k}(x)$ est la fontion de comparaison entre la cible T et un configuration x pour le descripteur d_k (voir définition 7.6 et propriété 7.1).

Encore une fois, la formulation 9.2 incite à considérer le problème de l'orchestration comme un problème de sac à dos multicritère à conteneurs multiples. Mais, comme nous l'avons dit au paragraphe précédent, la non-linéarité (et plus particulièrement la non-additivité) des fonctions objectifs à minimiser ne peut que nous en dissuader. Aussi séduisant soit-il, le cadre formel du sac à dos n'est pas assez souple pour supporter la complexité des mélanges de timbres.

Les variables de décision du problème 9.2 sont binaires : $x_i = 1$ si et seulement si l'échantillon s_i de la base de données est retenu dans l'orchestration, 0 sinon. On serait donc facilement tenté d'avoir recours à la technique d'encodage la plus répandue dans les algorithmes génétiques, qui consiste à représenter les solutions par des chaînes binaires (*bit-string encoding*), ici de longueur N_s. Trois raisons s'opposent à ce choix :

1. **La taille du problème.** Nous travaillons actuellement avec un réservoir d'environ 5000 sons. Pour une population de 1000 individus, l'encodage du type bit-string implique donc le stockage et la mise à jour récurrente d'une matrice à 5 millions d'éléments. Même si cela reste encore acceptable, la taille mémoire deviendra vite une limite sitôt que nous travaillerons avec davantage de sons et des populations potentiellement plus grandes.
2. **Le petit nombre d'items impliqués dans une solution.** Dans notre problème d'orchestration au maximum une petite centaine de sons (dans le cas de très grands orchestres) seront choisis parmi plusieurs milliers, occasionnant une fréquence très faible de valeurs égales à 1 dans la matrice représentant la population. On pourrait certes choisir d'avoir recours à une représentation parcimonieuse des données, mais cela ne résout pas le problème des contraintes multiples de capacité (cf. infra).
3. **La multiplicité des contraintes de capacité.** Dans le problème du sac à dos, la seule contrainte est la capacité globale du sac. La gestion des solutions inconsistantes produites au cours des générations fait en général appel à une méthode de réparation de type « glouton » qui retire récursivement les éléments les plus volumineux jusqu'à ce que la contrainte de capacité soit satisfaite ; éventuellement, des items plus petits peuvent alors être insérés dans le sac s'il reste de la place. En orchestration, il y a autant de contraintes de capacité que de type d'instruments, soit onze contraintes pour l'instrumentarium actuellement à disposition, et leur respect exige le recours à autant de procédures de réparation. Autant donc chercher une représentation qui permette de s'en affranchir.

Compte tenu de ces remarques nous optons pour un encodage en N-uplets d'entiers. Soit $(s_i)_{1 \leq i \leq N_s}$ un ensemble de sons instrumentaux représentatifs d'un orchestre de taille N. Une solution $K \in E$ est alors représentée par un N-uplet $i = (i_1, ..., i_N)$ à valeurs dans $\bar{D}_1 \times ... \times \bar{D}_N$. Chaque domaine $\bar{D}_k$ est défini par :

$$\bar{D}_k = D_k \cup \{e\} \ , \tag{9.3}$$

où e est l'élément neutre, indiquant que l'instrument k n'est pas impliqué dans l'orchestration, et D_k la partie de $\{1, ..., n\}$ associée à l'instrument k, i.e. $(s_i)_{i \in D_k}$ sont les sons de la base associés à l'instrument k.

En outre, afin d'éviter les doublons d'encodage (i.e. deux génomes différents exprimant la même solution) nous imposons la condition suivante, très facilement satisfaite à l'aide d'un tri par parties après chaque génération d'une solution :

$$\forall (k,l) \in \{1, ..., N\}^2 \text{ t.q. } \bar{D}_k = \bar{D}_l,\ (k < l) \Rightarrow (i_k < i_l) \tag{9.4}$$

Formulation 9.1. *Le problème de l'orchestration formulé en 7.4 et 9.2 se réécrit alors en un problème d'optimisation combinatoire sans contraintes :*

Variables : $i_1, ..., i_N$

Domaines : $\bar{D}_1, ..., \bar{D}_N$

Critères : $\forall k \in \{1, ..., K\},\ \min D_T^{d_k}(i) = D_T^{d_k}(i_1, \ldots, i_N)$

Remarque 9.1. *Grâce à cette représentation des solutions (qui se confond strictement avec leur encodage génétique), une configuration est donc définie comme un ensemble de couples variable/valeur dont chaque variable correspond à un instrument j de l'orchestre, à valeurs dans $\bar{D}_j$. On notera $D = \bar{D}_1 \times ... \times \bar{D}_N$, équivalent à E et à $\{0;1\}^{N_s}$*

Remarque 9.2. *Comme nous le verrons au paragraphe suivant, les opérateurs génétiques sont stables sur D. Toute configuration générée aléatoirement ou produite par une opération génétique est donc, par définition, consistante avec les contraintes de capacité (i.e. liée aux effectifs instrumentaux de l'orchestre).*

En d'autres termes, le problème de l'orchestration ainsi reformulé est non contraint. Nous verrons au chapitre 10 de quelle manière les éventuelles contraintes autres que celles de capacité peuvent être prises en compte au sein de l'algorithme de recherche.

9.2.1 Opérateurs génétiques

Dans notre algorithme d'orchestration, nous utilisons le *crossover uniforme* et la *mutation monopoint*, ou *1-mutation*. La figure 9.1 donne un exemple de ces opérations dans le cas d'un quatuor à cordes.

Définition 9.1. *Soient $i = (i_1, \ldots, i_N)$ et $j = (j_1, \ldots, j_N)$ deux chromosomes parents de $D = \bar{D}_1 \times ... \times \bar{D}_N$. Les rejetons k et l du* crossover uniforme *sont des N-uplets vérifiant :*

$$\forall n \in \{1, \ldots, N\},\ \left\{ \begin{array}{ll} k_n = i_n \wedge l_n = j_n & ou \\ k_n = j_n \wedge l_n = i_n & \end{array} \right. \tag{9.5}$$

En pratique, les rejetons sont engendrés par interversion aléatoire des coordonnées des chromosomes parents.

Remarque 9.3. *L'équation 9.5 certifie que si $(i,j) \in D^2$ (i.e. i et j représentent deux configurations consistantes avec les contraintes de capacité), alors $(k,l) \in D^2$ (le crossover uniforme est stable sur D).*

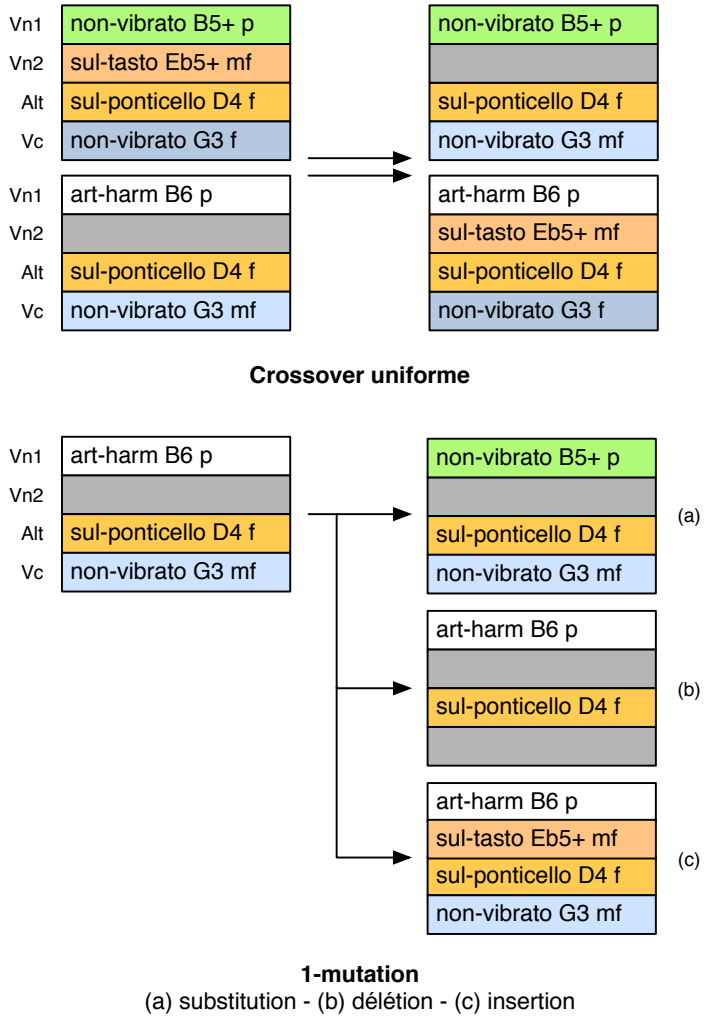

FIG. 9.1 – Opérateurs génétiques pour l'orchestration (cas d'un quatuor à cordes). La case vide (en gris) désigne l'élément neutre.

Lorsque deux chromosomes parents sont recombinés non pas par un crossover uniforme, mais, comme c'est souvent le cas, par un crossover monopoint (voir figure 5.5, les gènes éloignés ont plus de chances d'être séparés par le crossover que les gènes voisins dans la chaîne (voir Whitley [Whi93]). Cette propriété est parfois souhaitable dans le cas de représentations binaires échantillonnant un espace de décisions à valeurs réelles, pas dans le cas de l'orchestration. Considérons par exemple un quatuor à cordes représenté par des quadruplets (violon1, violon2, alto, violoncelle), comme il en va sur la figure 9.1. Le crossover monopoint séparera toujours le premier violon du violoncelle, et maintiendra toujours l'un des duos (violon1, violon2) ou (alto, violoncelle). En d'autres termes, le potentiel exploratoire du crossover monopoint dépend de l'ordre dans lequel les instruments de l'orchestre sont encodés dans les chromosomes. Le crossover uniforme n'implique quant à lui aucune de ces restrictions, raison pour laquelle nous le préférons pour l'orchestration.

Définition 9.2. *Soit* $i = (i_1, \ldots, i_N)$ *un chromosome parent de* $D = \bar{D}_1 \times ... \times \bar{D}_N$. *La* mutation monopoint *(ou* 1-mutation*) de* i *engendre un chromosome* j *défini par :*

$$\exists! m \in \{1, \ldots, N\}, \ \begin{cases} \forall n \in \{1, \ldots, N\}, \ i_n = j_n \\ j_m \in \bar{D}_m \setminus \{i_m\} \end{cases} \tag{9.6}$$

En pratique, j *est engendré par modification d'une seule variable* m*, la nouvelle valeur étant choisie dans le domaine associé* $\bar{D}_m$*. La mutation monopoint est donc stable sur* D*.*

Les domaines associés à chaque variable contenant l'élément neutre e (voir section 9.2), la mutation monopoint peut donc prendre la forme soit d'une substitution, soit d'une instanciation de variable (insertion) soit d'une désinstanciation (délétion). La figure 9.1 illustre ces trois types de mutation.

9.3 Calcul de *fitness*

Dans notre algorithme d'orchestration la *fitness* (i.e. l'adaptabilité d'un individu à son environnement) est calculée à l'aide de fonctions scalarisantes de Tchebycheff, car cette approche est en lien direct avec notre problématique d'inférence des préférences d'écoute du compositeur (voir sections 7.2.4 et 8.5.1). A chaque itération de l'algorithme, la population est évaluée avec un jeu de poids différent, donc selon des préférences d'écoute différentes. Si $\lambda = (\lambda_1, \ldots, \lambda_K) \in \Lambda$ est le jeu de poids courant, la fitness courante d'un point x de l'espace des critères est donnée par :

$$f(\lambda, x) = \max_{1 \leq k \leq K} \lambda_k \bar{x}_k \ ,$$

où $(\bar{x}_1, \ldots, \bar{x}_K)$ sont les coordonnées réduites de x, permettant de s'affranchir des différentes échelles de valeurs prises par chacun des critères (voir section 5.7).

Lorsque l'algorithme se termine, le compositeur peut, s'il le désire, « relancer » la recherche à partir d'une solution efficiente courante. La direction de recherche est alors « figée » par la norme induite par la solution choisie (cf. section 8.5.1). Le même algorithme est utilisé, cette fois-ci avec les poids associés à la norme induite, déterministes et invariants. La direction d'optimisation fixée, le problème devient alors mono-critère. Grâce à l'approche par fonctions scalarisantes, la méthode permettant d'« affiner » une solution intermédiaire n'est donc pas conceptuellement différente de l'algorithme de « recherche globale ». Seul change le caractère des poids : ils n'y sont plus aléatoires, mais fixés par le choix intermédiaire du compositeur. Il

y a équivalence entre le choix d'une solution et le choix d'une fonction scalarisante (ou d'une direction d'optimisation).

En un sens, le but de la recherche multicritère est ici de trouver la « bonne » fonction scalarisante pour un problème d'orchestration donné. Notons que grâce à cette approche, le seul fait de rendre les poids aléatoires ou déterminés permet de commuter entre un problème multicritère et un problème mono-critère.

9.4 Maintien de la diversité

Dans notre algorithme d'orchestration, le calcul de fitness n'est pas affecté par la densité locale. En revanche, cette dernière est utilisée pour réduire la taille de la population dès que celle-ci dépasse une valeur limite N_{max}. Les individus sont alors itérativement retirés des zones les plus denses jusqu'à ce que la taille de la population atteigne de nouveau N_{max}. Une fois identifiée le sous-ensemble de densité maximale, le retrait d'un individu peut se faire de trois manière différentes :

1. *"random"* : on retire un individu au hasard.
2. *"pareto"* : on retire au hasard un individu dominé au sein du sous-ensemble. Si aucun élément du sous-ensemble n'en domine un autre, le retrait est aléatoire.
3. *"fitness"* : on retire l'individu de plus faible fitness selon la fonction scalarisante courante.

La densité locale est évaluée par la méthode PADE (*Population size Adaptive Density Estimation*) présentée à la section 5.8 : l'espace de critère est « découpé » par une hypergrille dont les caractéristiques (dimensions, nombre de cellules) sont déterminées par les valeurs de critères de la population courante, et la densité de chaque individu est égale au nombre de configurations qui partagent la même cellule que lui.

Nous obtenons ainsi la procédure `decrease_PADE` (voir algorithme 3) qui permet de ramener une population à une cardinalité inférieure tout en préservant sa diversité :

Algorithme 3 Réduction de population par préservation de la diversité (`decrease_PADE`)

Arguments: Une population $X = \{x_1, x_2, \ldots\}$ associée à une partie $C = \{c_1, c_2, \ldots\}$ de l'espace de critères, une taille maximale N_{max}, une méthode d'élimination *method* (random, pareto, ou fitness)

tant que $\#X > N_{max}$ **faire**
    $D \leftarrow$ calculer la densité PADE de C
    $I \leftarrow \text{argmax}(D)$
    si *method* = random **alors**
        $i \leftarrow$ choisir aleatoirement un élément dans I
    sinon si *method* = pareto **alors**
        $i \leftarrow$ choisir aleatoirement un élément dominé dans I
    sinon si *method* = fitness **alors**
        $i \leftarrow$ choisir l'élément de plus faible fitness dans I
    fin si
    $X \leftarrow X \setminus \{x_i\}$
    $C \leftarrow C \setminus \{c_i\}$
fin tant que

9.5 Pseudo-code

L'algorithme que nous présentons ici permet d'obtenir, en un temps raisonnable, une approximation du front de Pareto associé à tout problème pouvant être ramené à la formulation 9.1. Grâce à la modélisation proposée au paragraphe 9.2, les contraintes de capacité (i.e. les restrictions imposées par les effectifs instrumentaux de l'orchestre) sont toujours satisfaites. Cette version de l'algorithme aborde donc des problèmes exclusivement non contraints. Son extension à des problèmes assortis de contraintes supplémentaires est l'objet du chapitre 10. La qualité de l'approximation retournée par notre algorithme est évaluée au chapitre 11.

L'algorithme 4 est un AG inspiré de l'algorithme MOGLS (*Multi-Objective Genetic Local Search*) de Jaszkiewicz [Jas01] [Jas02] pour son approche de l'optimisation multicritère par tirage aléatoire de fonctions sclarisantes (voir sections 5.7 et 9.3). Dans l'algorithme MOGLS, une seule configuration est obtenue à chaque génération par crossover, avant d'être améliorée par une méthode de recherche locale (il s'agit en l'occurence d'une simple descente) selon la fonction scalarisante en cours. La nouvelle configuration remplace alors, dans la population, la configuration de moins bonne fitness. La partie génétique de MOGLS peut donc être considérée comme un moyen de générer la configuration initiale d'un algorithme de descente pour lequel la fonction à optimiser est chaque fois différente.

Ishibushi et Murata [IY02] font observer que dans cette classe d'algorithmes — dit « hybrides » — la partie génétique tient un rôle d'*exploration* alors que la recherche locale est dévolue à l'*intensification* de la recherche. Les auteurs constatent par ailleurs qu'en général la majorité du temps de calcul est absorbée par la recherche locale, et recommandent, pour une performance optimale des méthodes hybrides, d'éviter ce déséquilibre. Soucieux de privilégier dans un premier temps l'exploration de l'espace de recherche au profit de l'intensification dans une zone particulière, nous avons donc choisi délibérément de ne pas implémenter de recherche locale dans notre algorithme d'orchestration. Ce parti pris va à l'encontre des résultats de Ishibushi et Murata [IY02], selon lesquels la méthode MOGLS sans recherche locale est moins performante que le SPEA de Zitzler & Thiele [ZT98b] [ZT98a] [ZT99]. Rappelons toutefois que le but premier de notre algorithme est de découvrir une configuration « intéressante » pour le compositeur, révélatrice de ses préférences en termes d'écoute du timbre (voir section 7.2). Si cette configuration n'est pas pleinement satisfaisante, elle permet au moins d'identifier la fonction scalarisante la plus pertinente pour un problème donné, selon laquelle une intensification de la recherche (voir paragraphe 9.3) pourra toujours être menée par la suite. Pas de recherche locale dans cet algorithme donc, mais, comparativement à MOGLS, un plus grand nombre de configurations engendrées par les opérateurs génétiques à chaque génération.

L'algorithme 4 prend en arguement une cible à orchestrer T. Une population initiale de taille N_{init}^{pop} est générée par instanciation aléatoire des variables, avec une probabilité d'instanciation égale à 0.55. Ainsi, chaque configuration de la population initiale mobilise en moyenne un peu plus de la moitié de la ressource orchestrale.

A chaque génération, un sous-ensemble de la population (de taille N_{mate}) est sélectionné par tournoi binaire, dont la supériorité sur les autres méthodes de sélection a été évoquée au paragraphe 5.6. Deux individus sont tirés aléatoirement dans la population courante, et le meilleur des deux selon la fonction scalarisante en cours est retenu dans le bassin de reproduction. Cette opération est répétée jusqu'à ce que le bassin atteigne la taille N_{mate}. Les opérations de crossover et mutations s'appliquent alors sur les éléments du bassin, et les nouveaux individus sont intégrés à la population courante.

Algorithme 4 Algorithme d'orchestration

Arguments: Une cible d'orchestration T, les paramètres de population N_{init}^{pop}, N_{max}^{pop}, N_{mate}, N_{max}^{pareto}, un nombre maximal d'itérations $iter_{max}$.

$X \leftarrow$ générer une population aléatoire de taille N_{init}^{pop}
$C \leftarrow$ **évaluer_population**(X, T)
$P \leftarrow$ `extract_pareto_set`(C) // (*algorithme 1*)
$iter \leftarrow 1$
tant que $iter \leq iter_{max}$ **faire**
    $C \leftarrow$ **évaluer_population**(X, T)
    $\lambda \leftarrow$ tirer un jeu de poids // (*selon la formule 5.8 de Jaszkiewicz*)
    $f \leftarrow \|C\|_\lambda$
    $M \leftarrow$ sélectionner N_{mate} individus dans X par tournoi binaire selon f
    $O \leftarrow$ crossover(M)
    $O \leftarrow$ 1-mutation(O)
    $C_O \leftarrow$ **évaluer_population**(O, T)
    $X \leftarrow X \cup O$
    $P \leftarrow$ `update_pareto_set`(C_O, O) // (*algorithme 2*)
    si $\#P > N_{max}^{pareto}$ **alors**
        $P \leftarrow$ `decrease_PADE`$(P, N_{max}^{pareto}, \text{random})$ // (*gestion de la taille de l'approximation par l'algorithme 3*)
    fin si
    si $\#X > N_{max}^{pop}$ **alors**
        $X \leftarrow X \setminus P$
        $C \leftarrow$ **évaluer_population**(X, T)
        $f \leftarrow \|C\|_\lambda$
        $X \leftarrow$ `decrease_PADE`$(X, N_{max}^{pop} - \#P, f)$ // (*gestion de la taille de la population dominée : on retire les individus dominés les moins adaptés pour la fitness courante*)
        $X \leftarrow X \cup P$
    fin si
    $iter++$
fin tant que
retourner P

Si la taille de la population dépasse une valeur limite N_{max}^{pop}, la procédure `decrease_PADE` est invoquée pour éliminer les individus situés dans les zones les plus denses. La diversité de la population dans l'espace des critères est ainsi préservée (voir sections 9.4 et 5.8). En pratique, on retire itérativement les individus de densité locale maximale et de moins bonne fitness selon la fonction scalarisante en cours. Afin d'éviter de retirer des solutions de l'approximation courante, l'échantillonnage de la population par la procédure `decrease_PADE` ne s'applique qu'aux solutions dominées.

A chaque génération, l'approximation courante est mise à jour par l'algorithme 3 (`update_pareto_set`). Si la taille de cette dernière dépasse une valeur limite N_{max}^{pareto}, la procédure `decrease_PADE` est invoquée. L'algorithme se termine lorsque le nombre de générations atteint $iter_{max}$.

9.6 Extension aux modèles d'instruments

Dans l'algorithme 4, la procédure **évaluer_population** désigne l'application séquentielle, sur les configurations à évaluer, des fonctions d'agrégation et fonctions de comparaison (voir paragraphes 7.3 et 7.6). L'intégration future dans notre système des travaux de Damien Tardieu [TR07] sur les modèles d'instruments (introduits au paragraphe 7.1.2) ne remet pas en cause le déroulement de l'algorithme d'orchestration : la procédure **évaluer_population** renvoie alors une mesure probabiliste des configurations, entièrement calculée par les modèles instrumentaux. L'indépendance des méthodes d'évolution et d'évaluation permet de commuter aisément entre une approche par descripteurs (la nôtre) et une approche par modèles (celle de Damien Tardieu).

Chapitre 10

Gestion des contraintes

Tous les attributs supportent des contraintes
CDCSolver : une méthode de réparation des configurations inconsistantes
Un nouvel algorithme d'orchestration sous contraintes
Contrôle de l'évolution temporelle à l'aide de contraintes

Nous présentons dans ce chapitre une méthode efficace pour la gestion, dans un problème d'orchestration, de contraintes autres que celles liées à l'effectif instrumental (voir chapitre 9). Nous commençons par énumérer un ensemble de contraintes N-aires portant sur les attributs des configurations, puis montrons qu'elles se répartissent en deux catégories : contraintes de *design* et contraintes de *conflit*. Nous introduisons alors l'heuristique de recherche locale *CDCSolver*, qui s'appuie sur cette dichotomie pour trouver en un temps acceptable une configuration consistante avec l'ensemble du réseau de contraintes. *CDCSolver* est évalué par comparaison avec un algorithme aléatoire sur des problèmes de petite taille et de difficulté variable. Nous terminons avec l'intégration de *CDCSolver* dans l'algorithme d'orchestration.

10.1 Contraintes en orchestration

10.1.1 Contraindre la recherche pour mieux l'orienter

La notion générique de contrainte a été introduite au paragraphe 7.5. Nous l'avions alors définie comme une condition que doivent satisfaire les attributs d'une configuration. Rappelons qu'à la différence des descripteurs, qui caractérisent le timbre selon une dimension perceptive, les attributs sont les paramètres symboliques de l'écriture musicale (instrument, note, mode de jeu, dynamique, sourdine...). Les contraintes portées sur les attributs ne restreignent donc l'espace des timbres qu'indirectement, en limitant les possibilités combinatoires dans l'espace des attributs (ou espace des décisions — voir section 7.6).

La contrainte la plus évidente à imaginer est la contrainte de capacité, liée aux limitations de l'effectif orchestral : si l'orchestre ne comprend par exemple que deux violons, l'attribut « instrument » ne pourra prendre la valeur « violon » qu'au plus deux fois, sans quoi la solution n'est pas réalisable. Le caractère incontournable des contraintes de capacité nous a fait adopter au paragraphe 9.2 une modélisation du problème permettant de s'en affranchir. Une orchestration est représentée par un N-uplet donc chaque coordonnée correspond à un instrument précis de l'orchestre, si bien qu'une configuration obtenue par instanciation aléatoire des variables ou par une opération génétique (voir section 9.2.1) est toujours consistante avec les

contraintes de capacité. D'une façon plus générale, on dira que la consistance est garantie pour toute contrainte de cardinalité maximale placée sur l'attribut « instrument », quelle que soit la valeur de ce dernier. Mais, nous allons le voir, il existe bien d'autres types de contraintes, portant sur d'autres attributs.

Notre expérience avec les compositeurs au cours de notre recherche d'une part, et celle plus générale des contraintes en composition assistée par ordinateur d'autre part, nous enseignent qu'à l'exception des problèmes de composition entièrement formalisables par la PPC[1], il est très difficile de spécifier des contraintes en dehors d'un « contexte » musical. Imaginons le cas d'un compositeur désirant orchester une série d'accords et disposant déjà de l'orchestration du premier. Selon le degré de contraste désiré à chaque changement d'accord, on peut imposer pour l'accord courant d'avoir au plus ou au moins un certain nombre de notes communes avec le précédent. Aussi rudimentaire soit-il, l'accord initial constitue déjà un « contexte » : c'est ici un point de départ, ce sera ailleurs un point de passage, ou un point d'arrivée. « Contraindre une orchestration », c'est donc déjà penser le matériau musical dans sa *mise en temps*, dans ses relations temporelles aux autres objets musicaux. Or notre outil d'orchestration a été conçu comme un environnement *hors temps* d'exploration du timbre : il produit une matière, un objet « brut » que le compositeur devra « polir » avant de l'insérer dans la composition. De fait, dans la grande majorité des cas, tous les problèmes d'orchestration commencent sans contraintes.

Bien souvent toutefois, les critiques faites aux suggestions d'orchestration — dans le cas où elles ne satisfont pas le compositeur — portent certes sur leurs propriétés timbrales, mais aussi sur leurs attributs : telle hauteur est présente dans la cible et manque dans l'orchestration, tel mode de jeu est inaudible en présence d'instruments jouant fortissimo, telle sourdine est nécessaire, la densité harmonique (i.e. le nombre de hauteurs différentes jouée simultanément) est trop faible ou trop forte, l'orchestration mobilise une trop grande ou trop petite proportion de l'orchestre, etc. La possibilité d'ajouter des contraintes sur les attributs est alors bienvenue pour « guider » la recherche vers des solutions plus pertinentes. Certes ces contraintes ne viennent pas spontanément au compositeur, mais elles surgissent en réaction aux solutions « libres » retournées par le système. D'un certain point de vue, on peut dire qu'elles viennent combler les carences éventuelles de la description du timbre (voir section 8.1) qui, dans certains cas, ne suffit pas pour découvrir les solutions les plus proches de la cible au sens perceptif.

10.1.2 Langage élémentaire de contraintes

D'un point de vue formel, nous définissons donc une contrainte par la donnée d'un attribut, d'un opérateur, et deux éléments optionnels :

$$<\textit{attribut}><\textit{opérateur}><\textit{quantité}\ (\text{opt.})><\textit{valeur}\ (\text{opt.})>$$

En théorie, n'importe quel attribut est recevable : instrument, famille, hauteur, mode de jeu, dynamique, sourdine, corde... si cela bien sûr a un sens musicalement. Quant aux opérateurs, ils se répartissent en trois catégories selon le nombre d'arguments supplémentaires qu'ils requièrent, comme l'indique le tableau 10.1.

L'opérateur *all-diff* associé à un attribut A signifie que toutes les composantes de l'orchestration doivent avoir des valeurs différentes pour l'attribut A.

[1] Les problèmes de séries d'intervalles, de suites d'accords ou de canons rythmiques, pour n'en citer que quelques uns, appartiennent à cette catégorie — voir à ce sujet la thèse de Charlotte Truchet [Tru04].

TAB. 10.1 – Opérateurs d'arité variable exprimant des contraintes sur les attributs des configurations.

Opérateur	Argument 1	Argument 2
all-diff	-	-
at-most-diff	quantité n	-
at-least-diff	quantité n	-
at-most	quantité n	valeur v
at-least	quantité n	valeur v

`Ex.1` – « L'orchestration ne doit jamais impliquer deux fois la même hauteur » s'écrit :

<*attribut*="**note**"><*opérateur*="**all-diff**">

L'opérateur *at-most-diff* (respectivement *at-least-diff*) associé à un attribut A et à une quantité n signifie que l'orchestration doit comporter au plus (respectivement au moins) n valeurs différentes pour l'attribut A.

`Ex.2` – « L'orchestration doit impliquer au minimum trois cuivres » s'écrit :

<*attribut*="**famille**"><*opérateur*="**at-least-diff**"><*quantité*="**3**"><*valeur*="**brass**">

`Ex.3` – « Tous les instruments doivent jouer avec la même dynamique » s'écrit :

<*attribut*="**dynamique**"><*opérateur*="**at-most-diff**"><*quantité*="**1**">

L'opérateur *at-most* (respectivement *at-least*) associé à un attribut A, une quantité n et une valeur v signifie que l'orchestration doit comporter au plus (respectivement au moins) n fois la valeur v pour l'attribut A.

`Ex.4` – « Au moins deux cordes doivent jouer avec le bois de l'archet » s'écrit :

<*attribut*="**mode-de-jeu**"><*opérateur*="**at-least**"><*quantité*="**2**"><*valeur*="**legno-tratto**">

`Ex.5` – « Au plus deux instruments peuvent jouer fortissimo » s'écrit :

<*attribut*="**dynamique**"><*opérateur*="**at-most**"><*quantité*="**2**"><*valeur*="**ff**">

Ces cinq opérateurs permettent de définir un langage de contraintes simple, expressif, et facilement extensible par l'ajout de nouveaux attributs. On pourrait très bien imaginer, par exemple, un attribut qui précise quel instrumentiste est requis pour émettre le son considéré. Tous les sons de flûte, flûte basse et flûte picolo peuvent en effet être produits par un flûtiste. Si l'orchestre n'en comprend qu'un seul, alors la contrainte :

<*attribut*="**instrumentiste**"><*opérateur*="**at-most**"><*quantité*="**1**"><*valeur*="**flûtiste**">

permet d'effectuer la recherche avec tous les sons de flûte tout en garantissant qu'un seul son sera compris dans l'orchestration. Le même raisonnement est évidemment valable pour les clarinettistes, trompettistes, trombonistes, saxophonistes et percussionnistes, qui peuvent être amenés à changer d'instrument au cours d'une même œuvre.

Nous avons en outre pourvu notre langage de deux opérateurs de cardinalité globale : *size-min* et *size-max*. Comme leur nom l'indique, ils permettent de minorer ou de majorer le nombre

TAB. 10.2 – Contraintes de cardinalité globale

Opérateur	Argument 1	Argument 2
size-min	quantité n	-
size-max	quantité n	-

d'instruments impliqués dans une orchestration. Ces opérateurs sont d'un grand intérêt lorsque le compositeur veut augmenter la « masse orchestrale » ou au contraire réserver une partie de l'orchestre pour un matériau musical se superposant à la texture proposée par l'outil.

Pour les raisons évoquées au début de cette section, les contraintes liées à l'exécution — i.e. au déroulement de la musique dans le temps — ne sont pas prises en compte au sein de notre système. Modéliser le potentiel sonore des instruments hors de tout contexte temporel est une tâche difficile, mais abordable — du moins partiellement — dans le cadre d'une thèse. Maîtriser les possibilités d'enchaînement de timbres réalisables par un instrument solo est d'une complexité bien supérieure, et ne parlons même pas de l'infinité de variations que peut produire un orchestre. Nous verrons cependant en fin de chapitre de quelle manière les contraintes peuvent être utilisées pour transformer progressivement une orchestration, sans toutefois disposer d'une formalisation temporelle du timbre. L'évolution sera alors contrôlée non pas par des descripteurs perceptifs, mais par les attributs symboliques des configurations successives.

10.1.3 Fonctions de coût

La notion de « fonction de coût » a été introduite au paragraphe 7.5. Elle s'appuie sur le concept de « contraintes molles » (*soft constraints*) avec lequel les méthodes de recherche locale sont famillières (voir paragraphe 6.4.1) : chaque contrainte est remplacée par une fonction de coût positive que l'on cherche à minimiser. Cette dernière est nulle si et seulement si la contrainte est satisfaite, et de valeur d'autant plus élevée que la contrainte est violée. L'idée sous-jacente est qu'on a d'autant plus de chance de parvenir à une configuration consistante que la configuration initiale viole peu les contraintes. En d'autres termes, nous supposons la fonction de coût globale, obtenue par sommation des fonctions de coût associées aux contraintes, est localement monotone.

Nos fonctions de coût ne dépendent que de l'opérateur impliqué dans la contrainte : *at-most*, *at-least*, *at-most-diff*, *at-least-diff*, *all-diff*, *size-min*, *size-max*. Nous les construisons selon la règle suivante : elles renvoient le nombre minimum de variables à modifier pour que la contrainte soit satisfaite. Elles sont calculées à partir de la représentation des configurations introduites à la section 9.2 : chaque variable correspond à un instrument, et chaque valeur à un son jouable par l'instrument. Une requête préalable dans une base de donnée permet de retourner, pour chaque son, la valeur de l'attribut impliqué dans la contrainte avant d'en calculer la consistance. Dans la suite de cette section, nous désignons respectivement par n et v les premier et second arguments éventuels de nos opérateurs.

at-most, at-least

Soit n_v le nombre de fois où apparait la valeur v dans la configuration (effectif de la valeur v). On a alors trivialement, pour les opérateurs *at-least* et *at-most* :

$$z_{at\text{-}least} = \max(n_v - n, 0) \qquad \text{et} \qquad z_{at\text{-}most} = \max(n - n_v, 0) \tag{10.1}$$

at-least-diff

Soit maintenant n_A le nombre total de valeurs que peut prendre l'attribut A et $(n_i)_{1\leq i\leq n_A}$ les effectifs des valeurs prises par A. Soit δ la fonction indicateur de $\mathbb{R}_+^*$ ($\delta(x) = 1$ si $x > 0$, $\delta(x) = 0$ sinon). La fonction de coût pour l'opérateur *at-least-diff* compte le nombre de valeurs différentes pour l'attribut A et renvoie la différence avec la quantité limite n si cette dernière et supérieure :

$$z_{at\text{-}least\text{-}diff} = \max\left(n - \sum_{i=1}^{n_A} \delta(n_i), 0\right) \tag{10.2}$$

at-most-diff

De façon non intuitive, la fonction de coût pour *at-most-diff* n'est pas la symétrique de la précédente. Soit par exemple la contrainte :

<*attribut*="**note**"> <*opérateur*="**at-most-diff**"> <*quantité*="**1**">
(Tous les instrument doivent jouer la même note)

Soient alors les deux configurations $\mathcal{K}_1$ = (C4,A4,C4,A4) et $\mathcal{K}_2$ = (C4,C4,C4,A4) (seules les hauteurs ont été mentionnées par souci de lisibilité. Si on compte le nombre de valeurs différentes pour l'attribut A et que l'on renvoie sa différence avec la quantité limite n si cette dernière est inférieure, on obtient alors $z(\mathcal{K}_1) = z(\mathcal{K}_2) = 1$. Autrement dit, il n'y a aucun intérêt pour un algorithme de recherche locale à remplacer A4 par C4 pour la seconde variable, alors que $\mathcal{K}_2$ est plus proche de satisfaire la contrainte que $\mathcal{K}_1$ (il faut modifier une seule variable dans $\mathcal{K}_2$ contre deux dans $\mathcal{K}_1$).

Soit $(\bar{n_i})_{1\leq i\leq n_A}$ la permutation de $(n_i)_{1\leq i\leq n_A}$ pour laquelle les effectifs sont classés dans l'ordre décroissant, ce qui donne $(2, 2)$ pour $\mathcal{K}_1$ et $(3, 1)$ pour $\mathcal{K}_2$. On a alors :

$$z_{at\text{-}most\text{-}diff} = \sum_{i=n+1}^{n_A} \bar{n_i} \ , \tag{10.3}$$

ce qui donne $z(\mathcal{K}_1) = 2$ et $z(\mathcal{K}_2) = 1$.

all-diff

Quant à l'opérateur *all-diff*, l'idée la plus immédiate consiste à retourner l'effectif maximal moins 1. Toutefois, si nous considérons les deux configurations $\mathcal{K}_1$ = (C4,A4,C4,A4) et $\mathcal{K}_2$ = (C4,G4,C4,A4), on obtient pour la contrainte *all-diff* sur les hauteurs les valeurs de coût $z(\mathcal{K}_1) = z(\mathcal{K}_2) = 1$. Encore une fois, un algorithme de recherche locale n'aura donc aucun intérêt à remplacer A4 par G4 pour la seconde variable, alors que $\mathcal{K}_2$ est plus proche de satisfaire la contrainte que $\mathcal{K}_1$. Alors qu'avec la foncion :

$$z_{all\text{-}diff} = \sum_{i=1}^{n_A} (n_i - 1)\delta(n_i - 1) \ , \tag{10.4}$$

on obtient $z(\mathcal{K}_1) = 2$ et $z(\mathcal{K}_2) = 1$.

size-min*, *size-max

Soit n^* le nombre de variables dont la valeur est différente de l'élement neutre e (voir section 9.2). On a alors trivialement :

$$z_{size\text{-}min} = \max(n - n^*, 0) \qquad \text{et} \qquad z_{size\text{-}max} = \max(n^* - n, 0) \tag{10.5}$$

Consistance globale

Comme la plupart des méthodes de recherche locale, notre algorithme favorise dans ces déplacements les configurations de consistance maximale pour l'ensemble du réseau de contraintes. L'évolution dans l'espace de recherche est donc guidée par une « fonction de coût globale », que nous définissons comme la somme des fonctions de coût associées à chaque constrainte. Elle peut s'interpréter une estimation du nombre de variables à modifier pour parvenir à une solution satisfaisant toutes les contraintes.

10.2 Contraintes de design, contraintes de conflit

La méthode *CDCSolver* est une heuristique de recherche locale permettant de résoudre les problèmes de satisfaction de contraintes formalisés à l'aide du langage simple introduit à la section 10.1.2. Comme toute méthode de recherche locale, *CDCSolver* dispose d'une heuristique de voisinage qui à chaque configuration associe un sous-ensemble de l'espace de recherche, parmi lequel sera choisie la nouvelle configuration. Le voisinage est déterminé de manière à améliorer la consistance globale de la solution courante. Dans l'algorithme de recherche adaptative (Codognet & Truchet [CD01] [CDT02] [Tru04]), présenté à la section 6.5 et dont s'inspire en partie *CDCSolver*, le voisinage est construit en remplaçant la variable impliquée dans le plus grand nombre de contraintes par toutes les valeurs de son domaine. Hélas, cette stratégie n'est pas directement applicable à notre problème. Il est en effet une difficulté inhérente aux contraintes globales : on ne peut, par définition, identifier précisément sur quelles variables elles portent. Si une contrainte est violée, il n'est pas possible de décider a priori des variables à modifier pour la satisfaire. Nous verrons toutefois au paragraphe suivant qu'un calcul de complexité raisonnable permet de choisir une variable prioritaire. Cette dernière dépendra avant tout du type de contrainte rencontrée :

- soit la contrainte porte sur une incompatibilié entre les valeurs de différentes variables : on parlera alors de *contrainte de conflit*, et l'on cherchera à réduire la fonction de coût associée en modifiant une variable dont la valeur est différente de l'élément neutre ;
- soit la contrainte définit une caractéristique désirée dans la solution : on parlera alors de *contrainte de design*, et l'on cherchera à réduire la fonction de coût associée en modifiant une variable dont la valeur est égale à l'élément neutre.

Au sein d'une méthode de résolution manipulant des instanciations *partielles* (i.e. des configurations où les variables ne sont pas toutes instanciées), une contrainte de conflit peut être résolue par désinstanciation d'une variable. L'heuristique $\mathcal{CN}$-tabou (Vasquez & Dupont [VHD03]) introduite à la section 6.6 s'appuie sur ce principe pour traiter des contraintes binaires. A chaque étape de l'algorithme, la configuration courante est une instanciation partielle et *localement* consistante (les variables instanciées ne violent aucune contrainte). Les conflits

potentiellement engendrés par l'instanciation d'une nouvelle variable sont alors résolus par désinstanciation des variables incompatibles avec la nouvelle valeur. L'algorithme s'arrête lorsque toutes les variables sont instanciées.

D'un point de vue plus abstrait, on peut donc dire que l'heuristique $\mathcal{CN}$-tabou permet de traiter des problèmes définis par un ensemble de contraintes de conflit et par une contrainte de design exigeant que toutes les variables doivent être instanciées. L'algorithme avance dans la résolution en diminuant la fonction de coût associée à la contrainte de design (i.e. en instanciant des variables), tout en s'adaptant à la volée aux contraintes de conflit qui surgissent des nouvelles instanciations.

L'algorithme *CDCSolver* (*Conflict/Design Contraint Solver*) généralise ce principe à toute containte de design ou de conflit. Nous proposons pour cela une extension des notions d'*instanciation partielle* et de *consistance locale* :

Définition 10.1. *Soit un CSP défini par la donnée d'un ensemble de variables et de domaines associés, et de contraintes N-aires distinguables en contraintes de design et contraintes de conflit. On dira qu'une configuration est partiellement instanciée si et seulement si au moins une contrainte de design n'est pas satisfaite. On dira qu'une configuration est localement consistante si et seulement si elle vérifie toutes les contraintes de conflit.*

A partir d'une configuration initiale inconsistante, l'algorithme *CDCSolver* modifie itérativement les variables impliquées dans les contraintes de conflit (nous verrons au paragraphe suivant comment ces variables sont choisies) jusqu'à l'obtention d'une instantiation partielle localement consistante. Ensuite, une variable dont la valeur est égale à l'élément neutre (ce qui, d'un point de vue musical, correspond à un silence pour l'instrument associé) est modifiée de manière à réduire les violations des contraintes de design. Ce processus est réitéré dès qu'une contrainte de conflit est violée (on peut alors parler de *propagation* des affectations), jusqu'à l'obtention d'une configuration consistante avec l'ensemble des contraintes. Un mécanisme de mémoire à court-terme de type tabou [GL97] permet en outre d'éviter les cycles dans la recherche et de sortir des minima locaux (voir paragraphe 6.4.1).

N.B. Dans un contexte d'orchestration, une configuration dont certaines variables ont pour valeur l'élément neutre (c'est-à-dire, d'un point de vue musical, un silence) n'est en aucun cas une instanciation partielle. Il n'est en effet précisé nulle part que tous les instruments disponibles doivent être mobilisés dans une orchestration. Ce sont au contraire les variations de l'effectif instrumental et des plans sonores qui, au cours du temps, créent la richesse du timbre orchestral. En cela, l'orchestration se distingue des problèmes classiques pour lesquels il n'y a en général pas d'élément neutre. Dans le problème des N-reines par exemple, une configuration dans laquelle la position d'une reine n'est pas déterminée n'est évidemment pas recevable. A l'extrême inverse, une configuration constituée uniquement d'éléments neutres est une solution, certes inintéressante, mais admissible pour un problème d'orchestration sans contraintes de design : le silence total ne peut pas violer de contrainte de conflit. Encore une fois, la notion d'« instanciation partielle » n'a de sens qu'en présence de contraintes de design. Dans la plupart des CSP classiques, il n'y en a qu'une seule : toutes les variables doivent être instanciées.

Le tableau 10.3 recense l'ensemble des opérateurs introduits au paragraphe 10.1.2 et les répartit en fonction du type de contrainte qu'ils induisent. Notre langage de contraintes pour l'orchestration peut ainsi être étendu de deux manières :

TAB. 10.3 – Classification des opérateurs par type de contrainte

Contraintes de design	Contraintes de conflit
at-least	*at-most*
at-least-diff	*at-most-diff*
size-min	*size-max*
	all-diff

- soit, nous l'avons mentionné en infra, par l'ajout de nouveaux attributs ;
- soit par l'ajout de nouveaux opérateurs, dès lors qu'ils s'identifient à l'un des deux types de contraintes — design ou conflit — et qu'ils s'accompagnent d'une fonction de coût conforme (i.e. propotionnelle au niveau de violation de la contrainte, et dans l'idéal renvoyant le nombre minimal de variables à modifier pour la satisfaire).

10.3 *CDCSolver*

Dans cette section, nous nous concentrons sur la dimension théorique de l'heuristique *CDCSolver* et en exposons les mécanismes au travers des notions génériques de *contraintes de design* et *contraintes de conflit*, indépendamment des opérateurs définis en supra.

Nous avons détaillé, au chapitre 6, deux méthodes récentes de recherche locale pour la résolution de problèmes de satisfaction de contraintes : la recherche adaptative et l'heuristique $\mathcal{CN}$-tabou. La première manipule des configurations totalement instanciées et utilise une « projection » des contraintes sur les variables pour choisir celle à modifier en priorité. La seconde garantit en permanence une consistance locale sur des instanciations partielles. Lorsqu'une nouvelle valeur est assignée à une variable libre, l'instancation est propagée en désaffectant les variables en conflit.

L'algorithme *CDCSolver* combine les deux méthodes. A partir d'une configuration inconsistante, on commence par modifier ou désinstancier — en les remplaçant par l'élément neutre — les variables impliquées dans les contraintes de conflit. Lorsque ces dernières sont toutes vérifiées, une variable libre est alors instanciée de manière à réduire les contraintes de design. Les affectations se poursuivent soit jusqu'à la satisfaction de l'ensemble des contraintes, soit jusqu'à l'apparition de nouveaux conflits. Dans ce dernier cas, le processus précédent est répété tant qu'un nombre maximal d'itérations n'a pas été dépassé. A chaque mouvement, une heuristique permet de déterminer la variable à modifier en premier (dans le cas d'une contrainte de conflit) ou à instancier en priorité (dans le cas d'une contrainte de design). Contrairement à la recherche adaptative, il n'est pas possible de comptabiliser le nombre de contraintes violées dans lesquelles intervient chaque variable, en raison du caractère global de nos contraintes. Nous procédons alors de la manière suivante :

- Si une contrainte de conflit est violée dans la configuration courante $\mathcal{K}$, alors nous calculons la consistance pour toutes les configurations obtenues à partir de $\mathcal{K}$ en remplaçant tour à tour chaque variable instanciée par l'élément neutre. La configuration de coût minimal correspond alors à la variable à modifier en priorité (voir algorithme 5).
- Si une contrainte de design est violée dans la configuration courante $\mathcal{K}$, alors nous calculons la consistance pour toutes les configurations obtenues à partir de $\mathcal{K}$ en remplaçant

tour à tour chaque variable libre par une valeur aléatoire de son domaine. La configuration de coût minimal correspond alors à la variable à instancier en priorité (voir algorithme 6). Si aucune variable n'est libre, le choix s'effectue par désinstanciation, selon la méthode appliquée aux variables de conflit.

Dans les deux cas, la valeur de la variable modifiée est remplacée par toutes les autres du domaine, et la configuration de coût global minimal est retenue comme configuration courante.

Algorithme 5 Modification de la pire variable conflictuelle (`change_worst_variable`)

Arguments: Une configuration initiale $\mathcal{K} = (x_1, \ldots, x_N)$, une fonction de coût z.

pour $i \in \{1, \ldots, N\}$ **faire**
  si $x_i \neq e$ **alors**
    $\mathcal{K}_i \leftarrow (x_1, \ldots, x_{i-1}, e, x_{i+1}, \ldots, x_N)$
    $z_i \leftarrow z(\mathcal{K}_i)$
  sinon
    $z_i \leftarrow \infty$
  fin si
fin pour
$i^* \leftarrow \text{argmin } z_i$
pour $x_{i^*}^j \in \bar{D}_{i^*} \setminus \{x_{i^*}\}$ **faire**
  $\mathcal{K}_j \leftarrow (x_1, \ldots, x_{i^*-1}, x_{i^*}^j, x_{i^*+1}, \ldots, x_N)$
  $z_j \leftarrow z(\mathcal{K}_j)$
fin pour
$j^* \leftarrow \text{argmin } z_j$
retourner $\mathcal{K}_{j^*}$

Cycles et minima locaux

Le fonctionnement de *CDCSolver* (voir algorithme 7) est donc basé sur une alternance de méthodes `change_worst_variable` et `instanciate_best_variable`. Bien souvent, ces dernières ont des objectifs contradictoires : remplacer une variable affectée par l'élément neutre permet de soulager les contraintes de conflit, au risque de violer davantage de contraintes de design (et vice-versa). L'algorithme peut donc rapidement se retrouver enfermé dans un cycle dû à la présence d'un minimum local dans la fonction de coût (voir section 6.4.1). Pour éviter ce problème, *CDCSolver* dispose d'une mémoire à court terme de type « tabou ». Cette technique proposée par Glover [GL97] consiste à stocker dans une liste FIFO — dite « liste tabou » — les dernières configurations visitées. Lors de l'exploration du voisinage courant, le mouvement ne peut se faire vers une configuration de la liste tabou.

En pratique on se contente souvent, pour éviter d'avoir à rechercher une configuration passée dans une liste potentiellement longue, de « marquer tabou » la dernière variable modifiée ; on ne pourra alors pas changer cette variable tant qu'elle demeurera dans la liste tabou. *CDCSolver* utilise une version dynamique de ce principe : lorsqu'une variable est modifiée, elle est « marquée tabou » pour un nombre d'itérations égal au nombre de fois où la variable a déjà été modifiée depuis le début de l'exécution. Ce mécanisme, utilisé par Vasquez & Dupont [VHD03] dans l'heuristique $\mathcal{CN}$-tabou, permet de s'adapter en cours d'exécution à la longueur des cycles engendrés par le paysage de la fonction de coût.

Algorithme 6 Instanciation de la meilleure variable libre (`instanciate_best_variable`)

Arguments: Une configuration initiale $\mathcal{K} = (x_1, \ldots, x_N)$, une fonction de coût z.

pour $i \in \{1, \ldots, N\}$ **faire**
  si $x_i = e$ **alors**
    $x_i^r \leftarrow$ une valeur aléatoire de D_i
    $\mathcal{K}_i \leftarrow (x_1, \ldots, x_{i-1}, x_i^r, x_{i+1}, \ldots, x_N)$
    $z_i \leftarrow z(\mathcal{K}_i)$
  sinon
    $z_i \leftarrow \infty$
  fin si
fin pour
si min $z_i = \infty$ **alors**
  $\mathcal{K}^* \leftarrow$ `change_worst_variable`$(\mathcal{K})$
  retourner $\mathcal{K}^*$
sinon
  $i^* \leftarrow$ argmin z_i
  pour $x_{i^*}^j \in D_{i^*}$ **faire**
    $\mathcal{K}_j \leftarrow (x_1, \ldots, x_{i^*-1}, x_{i^*}^j, x_{i^*+1}, \ldots, x_N)$
    $z_j \leftarrow z(\mathcal{K}_j)$
  fin pour
  $j^* \leftarrow$ argmin z_j
  retourner $\mathcal{K}_{j^*}$
fin si

Dans notre algorithme, seules les variables non tabou peuvent donc être modifiées. Cette restriction est forte, car elle interdit d'explorer de nombreuses configurations n'ayant pas encore été visitées. La technique habituellement utilisée pour contrecarrer cet effet est l'*aspiration* : si la modification d'une variable marquée « tabou » permet d'atteindre une valeur de coût inférieure à celle de la meilleure configuration obtenue jusqu'alors, le statut « tabou » est exceptionnellement révoqué. En pratique, les méthodes `change_worst_variable` et `change_best_variable` commencent par essayer de modifier une variable marquée « tabou ». Si l'aspiration n'est pas justifiée, le voisinage non « tabou » est alors exploré.

Pour des raisons de lisibilité, les opérations liées à la gestion des statuts « tabou » ne figurent pas dans les algorithmes 5, 6 et 7.

Complexité

Soient N le nombre de variables (i.e. le nombre d'instruments dans l'orchestre), d le nombre moyen de valeurs dans chaque domaine $\bar{D}_i$ et p le nombre de contraintes dans un problème d'orchestration. A chaque itération de *CDCSolver*, le calcul de la variable à modifier nécessite au plus Np opérations (procédures `change_worst_variable` ou `change_best_variable` et engendre un voisinage de $d-1$ configurations sur lesquelles la consistance doit être calculée à chaque fois, soit $(d-1)p$ opérations. La complexité à chaque itération de *CDCSolver* est donc dans le pire cas en $\mathcal{O}((N+d-1)p)$.

Dans l'algorithme de recherche adaptative, l'identification de la variable à modifier ne dépend que des valeurs des p contraintes calculées sur la configuration en cours, soit un calcul à chaque étape en $\mathcal{O}(dp)$. La complexité de *CDCSolver* est donc légèrement supérieure, mais

Algorithme 7 Réparation des configurations inconsistantes (`cdcsolver`)

Arguments: Une configuration initiale $\mathcal{K}$, une fonction de coût z.

```
1: K_best ← K
2: iter ← 1
3: tant que iter < iter_max faire
4:     tant que K viole au moins une contrainte faire
5:         tant que K viole au moins une contrainte de conflit faire
6:             K ← change_worst_variable(K)
7:             si z(K) ≤ z(K_best) alors
8:                 K_best ← K
9:             fin si
10:            iter + +
11:        fin tant que
12:        si K viole au moins une contrainte de design alors
13:            K ← instanciate_best_variable(K)
14:            si z(K) ≤ z(K_best) alors
15:                K_best ← K
16:            fin si
17:            iter + +
18:        fin si
19:    fin tant que
20: fin tant que
21: retourner K_best
```

reste du même ordre de grandeur que la recherche adaptative, le paramètre N étant dans tous les cas majoré par la taille des plus gros orchestres.

Intégration de la fitness dans la fonction de coût

La fonction de coût z dans l'algorithme 7 est par défaut la somme des fonctions de coût associées à chaque contrainte du réseau. A chaque mouvement, l'algorithme se déplace donc vers la configuration du voisinage courant qui viole le moins les contraintes. Si plusieurs choix sont possibles, la nouvelle configuration est choisie aléatoirement, sans se soucier de ses propriétés timbrales (i.e. de ses valeurs de descripteurs).

Dans un contexte d'optimisation multicritère, on pourrait très bien décider de choisir, au sein du voisinage courant de consistance maximale (i.e. de fonction de coût minimale), une solution non-dominée. Cela nécessiterait toutefois un grand nombre de comparaisons de valeurs à chaque mouvement. En revanche, il existe un cas de figure dans lequel les poids relatifs des critères intervenant dans les fonctions scalarisantes de Tchebycheff (voir sections 5.7 et 9.3) sont connus : lorsqu'après une première exécution de l'algorithme, le compositeur choisit une solution intermédiaire pour relancer la recherche (voir section 13.3) ou pour la transformer à l'aide de contraintes (voir sections 10.6 et 13.3), alors les poids ne sont plus aléatoires, mais déduits des valeurs de critères de cette solution (voir annexe B). La recherche devient alors mono-critère, guidée par une fonction scalarisante unique.

Nous avons dans ce cas tout intérêt à considérer, en plus de la consistance, la valeur de fitness pour guider les mouvements de *CDCSolver*. Lors de l'exploration du voisinage courant, la nouvelle configuration sera celle de meilleure fitness parmi les configurations les plus

consistantes (i.e. de coût minimal). Il suffit pour cela de modifier la fonction de coût z dans les méthodes `change_worst_variable` ou `change_best_variable` ainsi que dans l'algorithme principal : en rajoutant à z la valeur de fitness, on obtient une fonction de coût qui favorise les configurations les plus consistantes, tout en pénalisant, à niveau égal de violation de contraintes, les configurations de plus grande fitness (i.e. les moins adaptées).

Quand cela est possible, *CDCSolver* prend donc à sa charge une partie de l'optimisation. La recherche profite alors de la complémentarité entre les méthodes de recherche locale et les méthodes à population.

10.4 Evaluation

L'algorithme *CDCSolver* a été évalué à l'aide sur des problèmes de petite taille. Nous réutilisons ici les cibles polyphoniques à quatre sons construites à la section 8.2 et utilisées dans l'ensemble du chapitre 8 pour valider notre approche théorique de l'orchestration sur une description réduite du timbre instrumental.

Rappelons que ces cibles ont été construites de manière à satisfaire trois types de contraintes, cela afin de réduire suffisamment l'espace de recherche pour pouvoir en calculer l'ensemble des possibilités en un temps acceptable :

- Contraintes de cardinalité : les cibles sont des mixtures impliquant exactement quatre instruments.
- Contraintes de hauteur : les cibles sont des mixtures comprenant quatre notes différentes dans un intervalle de deux octaves.
- Contraintes de capacité : les cibles sont des mixtures n'utilisant qu'une seule fois le même instrument.

Chaque cible ainsi construite induit un espace de possibilités, que nous avons choisi de restreindre par un orchestre constitué d'un seul instrument de chaque type ; les contraintes de capacité y sont donc toujours vérifiées, grâce à la modélisation adoptée dans le chapitre précédent. En outre, nous excluons de cet ensemble de possibilités toute configuration ne comprenant pas exactement quatre instruments. Ces deux limitations nous permettent de réutiliser les espaces de recherche patiemment calculés au chapitre 8 pour chaque mixture cible.

N.B. Les mixtures dont nous nous servons dans cette évaluation n'ont ici pas d'intérêt en tant que « cibles » (i.e. comme objectifs d'un problème d'optimisation), mais parce qu'elles sont liées à des espaces de possibilités déjà calculés et qui peuvent être définis par deux contraintes et un filtre : contraintes de cardinalité (l'espace se limite aux combinaisons de quatre instruments), contraintes de capacité (on ne peut pas avoir deux fois le même instrument dans une combinaison), et filtre sur les hauteurs (l'espace de recherche est limité aux sons dont la hauteur appartient à celle de la mixture cible — ce n'est qu'au cours de la recherche proprement dite que la condition d'avoir exactement toutes les notes de la cible dans la solution deviendra une contrainte).

A chaque cible correspond donc un ensemble de configurations déjà calculées et satisfaisant une contrainte de cardinalité. Les contraintes de capacités étant implicitement satisfaites par la représentation même des combinaisons, nous n'en ferons plus mention dans cette section.

Pour chaque cible, l'ensemble ainsi défini va alors pouvoir constituer l'espace de recherche d'un problème de satisfaction de contraintes sur lequel évaluer l'algorithme *CDCSolver*. Nous

appelerons désormais cet ensemble « espace de référence de niveau 2 » car il est défini à l'aide de deux contraintes de cardinalité :

<*attribut*=∅><*opérateur*="**size-min**"><*quantité*="**4**"><*valeur*=∅>
<*attribut*=∅><*opérateur*="**size-max**"><*quantité*="**4**"><*valeur*=∅>

A partir de cette espace de référence, nous construisons des sous-ensembles inclus les uns dans les autres par ajout successif de contraintes supplémentaires mentionnées dans le tableau 10.4 (chaque niveau est donc défini par toutes les contraintes des niveaux inférieurs). Cette opération est répétée pour toutes les mixtures polyphoniques à quatre sons : notre base de test dénombre donc 500 instances de problèmes, comprenant chacune 5 niveaux de difficulté croissante (de 3 à 7) par rapport aux espaces de recherche (ensembles de référence de niveau 2). *CDCSolver* a donc été exécuté 2500 fois dans ce processus d'évaluation.

TAB. 10.4 – Contraintes utilisées pour l'évaluation de *CDCSolver*

Niveau	Description
2	Exactement quatre instruments dans l'orchestration
3	Les quatre notes de la cible présentes dans l'orchestration
4	Les instruments sont tous de familles différentes
5	Au moins un violoncelle dans l'orchestration
6	Au plus deux dynamiques différentes
7	Au plus un mode de jeu *ordinario*

TAB. 10.5 – Evaluation de *CDCSolver* : taille des ensembles de solutions

Niveau	Nombre moyen de solutions	Proportion // niveau 2
2	2.3×10^8	100.00%
3	2.0×10^7	8.78%
4	705880	0.31%
5	230700	0.10%
6	120360	0.05%
7	14927	0.01%

Le tableau 10.5 donne une idée de la taille moyenne des ensembles consistants pour chaque niveau de contrainte. Il permet de se faire une idée de la difficulté du problème : pour le niveau 6 par exemple, seules 0.05% des configurations du niveau 2 sont consistantes. La probabilité de tirer au hasard une solution est donc d'environ 1 sur 2000. Une méthode de recherche purement aléatoire ou un algorithme *generate-and-test* (voir paragraphe 6.3.1) découvriront une solution au bout de 2000 itérations en moyenne.

Les performances comparatives de *CDCSolver* par rapport à un algorithme aléatoire (random) sont reportées dans le tableau **??**. La dernière colonne montre que plus la difficulté du problème augmente, plus la performance relative de *CDCSolver* est élevée.

TAB. 10.6 – Performances de *CDCSolver* comparées à un algorithme purement aléatoire (les configurations sont évaluées selon leur niveau de violation de contrainte).

Niveau	Taux moyen de réussite	Nombre moyen d'itérations	Random	Performance // random
3	100.00%	5.0	12.2	2.6
4	99.90%	19.4	403	26.0
5	99.30%	41.1	1115	32.1
6	99.50%	38.3	2104	67.0
7	95.50%	102.8	19469	264.1

Dans cette évaluation, l'algorithme *CDCSolver* ne pouvait exécéder 500 itérations. Dans plus de 99 % des cas pour les problèmes de niveau inférieur ou égal à 6, et dans plus de 95 % des cas pour le niveau 7, une configuration consistante a été trouvée en moins de 500 itérations. Le nombre moyen d'itérations bien inférieur à 500 pour l'ensemble des niveaux de difficulté montre que la légère dégradation des performances pour les problèmes de niveau 7 peut être imputée à des instances particulièrement difficiles pour lesquelles davantage d'itérations auraient été nécessaires.

Nous avons vu à la section 10.3 que le mouvement vers une configuration du voisinage courant pouvait être déterminé de deux manières :

- soit en choisissant une configuration non tabou qui minimise la somme des fonctions de coût associées aux contraintes ;
- soit en choisissant, parmi les configurations non tabou qui minimisent la somme des fonctions de coût associées aux contraintes, celle de fitness minimale, sous réserve que celle-ci soit calculable (i.e. qu'un jeu de poids — ou une fonction scalarisante — soit spécifié pour l'agrégation des contraintes en une valeur unique).

Nous avons également testé les performances de *CDCSolver* dans ce second cas de figure, en donnant pour fonction scalarisante la norme induite (voir section 8.5.1 et annexe B) par chaque solution théorique (i.e. la configuration qui reproduit exactement l'orchestration utilisée comme mixture cible — voir section 8.3).

TAB. 10.7 – Performances de *CDCSolver* (les configurations sont évaluées selon leur niveau de violation de contrainte et sur leur fitness).

Niveau	Taux moyen de réussite	Nombre moyen d'itérations	Random	Performance // random
3	100.00%	4.9	12.2	2.6
4	100.00%	16.6	403	25.2
5	100.00%	32.1	1115	36.4
6	100.00%	31.5	2104	69.1
7	96.60%	57.6	19469	356.6

Les performances avec fitness sont reportées dans le tableau 10.7. Encore une fois, ces dernières augmentent (relativement au hasard) avec la difficulté du problème. Autre phénomène notable, les taux moyens de réussite (découverte d'une solution consistante) et le nombre moyen d'itérations sont meilleurs lorsque les critères audio interviennent dans l'évaluation des configurations. La fitness semble donc apporter une information supplémentaire pour diminuer le niveau de violation des contraintes. C'est a priori surprenant, car la fitness est calculée à l'aide de la norme induite par la solution théorique du problème d'optimisation, laquelle n'a a priori aucune raison de satisfaire les contraintes du tableau 10.4. Toutefois, remarquons que par construction des cibles de test, les solutions théoriques sont toujours consistantes jusqu'au niveau 3. Or les contraintes définissant le troisième niveau figurent (avec celles des niveaux 4 et 7) parmi les plus délicates à satisfaire : il faut dans le pire cas modifier trois variables pour obtenir la consistance. Faire intervenir la fitness dans l'évaluation des configurations permet donc peut-être, en favorisant (en principe) les mouvements vers la solution théorique, de satisfaire plus facilement la contrainte sur les hauteurs.

TAB. 10.8 – Amélioration de fitness avec *CDCSolver* dans le cas où les valeurs de critères interviennent dans l'évaluation des configurations.

Niveau	Fitness moyenne initiale	Fitness moyenne finale	Amélioration
3	0.2706	0.1430	45.96%
4	0.2857	0.1337	50.53%
5	0.2800	0.1161	55.27%
6	0.2856	0.1403	47.35%
7	0.2751	0.1599	37.04%

Le tableau 10.8 montre en outre que lorsque les valeurs de critères audio sont prises en compte dans l'évaluation du voisinage, *CDCSolver* améliore de façon significative la fitness d'une configuration aléatoire inconsistante. L'algorithme peut donc combiner à la fois réparation et affinage des configurations. Cette observation est pour nous d'un double intérêt :

1. Dans un cas d'orchestration sous contraintes, lorsque l'algorithme est relancé à partir d'une solution intermédiaire jugée pertinente par le compositeur (voir sections 9.3 et 13.3), le processus d'optimisation est pris en charge à la fois par l'algorithme génétique et par *CDCSolver*. La complémentarité d'une méthode à population et d'une méthode de recherche locale (voir section 5.3) contribue alors à l'efficacité du processus global.
2. Lorsque *CDCSolver* est utilisé pour transformer graduellement une solution d'orchestration à l'aide de contraintes (voir sections 10.6 et 13.3), le mouvement d'une configuration à la suivante s'effectue de manière à minimiser les différences perceptives entre les timbres.

En résumé, l'intégration d'une phase d'optimisation au sein de *CDCSolver* est possible et souhaitable sitôt qu'une bonne fonction scalarisante a été identifiée pour le problème en cours. Nous retrouvons là le constat énoncé aux chapitres précédents quant à notre approche multicritère : elle doit permettre d'inférer au plus vite les préférences d'écoute du compositeur, afin d'injecter cette information dans les étapes suivantes du processus de recherche.

10.5 Intégration de *CDCSolver* dans l'algorithme d'orchestration

Nous avons vu au paragraphe 6.7 que la gestion des contraintes au sein d'un algorithme génétique pouvait être assurée de plusieurs manières :

- soit *directement*, en s'assurant que la consistance est toujours respectée au sein de la population ;
- soit *indirectement*, en pénalisant les configurations non consistantes.

La première méthode nécessite soit des opérateurs génétiques ad-hoc, capables d'engendrer des individus consistants, soit de rétablir systématiquement la consistance de tout individu violant les contraintes. On a dans le second cas recours à une méthode de « réparation » cohérente avec le processus d'optimisation.

Nos opérateurs génétiques sont « aveugles » aux contraintes, et produisent des configurations inconsistantes en grand nombre. On serait donc tenté d'utiliser *CDCSolver* comme méthode de réparation. Malheureusement, l'essentiel du temps de calcul risque, en suivant cette méthode, d'être consacré à rétablir la consistance, au détriment de l'optimisation.

En conséquence, nous choisissons donc d'utiliser notre méthode de satisfaction de contraintes uniquement au début de l'exécution de l'algorithme d'orchestration : une population initiale est d'abord instanciée aléatoirement de manière à satisfaire les contraintes de cardinalité. Dans un second temps, les individus violant les autres contraintes sont « réparés » à l'aide de *CDCSolver* : on obtient ainsi une population initiale consistante avec l'ensemble du réseau.

Dans la suite, la gestion directe des contraintes cède le pas à une gestion indirecte : au cours de l'évolution, les solutions les plus consistantes sont favorisées dans les étapes de sélection et de mise à jour de la population (voir section 5.5), grâce à la dominance au sens de Deb introduite à la définition 6.1. Cette dernière a l'avantage de ne nécessiter aucun paramètre de pénalisation des critères pour les configurations inconsistantes : les configurations sont comparées soit sur leurs niveaux de violation des contraintes, soit sur leurs critères si elles sont de consistances égales :

$$x \preceq y \quad \Leftrightarrow \quad (z(x) < z(y) \ \vee \ (z(x) = z(y) \wedge x \prec y))$$

Dans notre algorithme d'orchestration (voir section 9.5), les configurations ne sont jamais comparées selon la dominance de Pareto, mais toujours après une agrégation des critères par une fonction scalarisante de Tchebycheff. Les relations de dominance entre les configurations sont alors « projetées » sur une distance mono-valuée — la fitness — qui permet de comparer toute paire de configurations.

En présence de contraintes, nous utilisons après chaque agrégation des critères une fonction **pénaliser_fitness** qui permet de retrouver les propriétés de la dominance au sens de Deb. Soient respectivement $f = \{f(x) \mid x \in X\}$ et $z = \{z(x) \mid x \in X\}$ les valeurs de fitness et de fonction de coût dans la population courante. On définit alors :

$$\forall x \in X, \ \textbf{pénaliser\_fitness}\,(f(x), z(x)) = \begin{cases} f(x) & \text{si } z(x) = 0 \\ z(x) + \max f & \text{sinon} \end{cases}$$

Ce mécanisme de pénalisation intervient à deux reprises au cours d'une itération de l'algorithme :

- Lors de la sélection des individus pour la reproduction : les configurations ont d'autant plus de chances d'être sélectionnées qu'elles violent peu les contraintes.
- Lors de la mise à jour de la population : les configurations de plus faible consistance sont retirées en premier.

L'algorithme d'orchestration sous contraintes (algorithme 8) présente, outre les aspects de pénalisation de fitness pour les configurations inconsistantes, deux particularités par rapport à sa version précédente (algorithme 4) :

- Lors de l'application des opérateurs génétiques, le crossover n'est pas nécessairement suivi d'une mutation (modification aléatoire d'une coordonnée). Nous verrons en effet au chapitre 11 qu'en présence de contraintes, la mutation est plus à même que le crossover de produire des configurations inconsistantes. Elle n'est donc plus utilisée certainement, mais avec une probabilité p_{mute}, égale à la probabilité d'obtenir une configuration consistante par instanciation aléatoire. Elle est calculée sur la population initiale avant réparation.
- La mise à jour de la population (lorsque celle-ci dépasse sa taille maximale autorisée) peut se faire de deux manières différentes : soit on retire itérativement les solutions les moins consistantes jusqu'à descendre à la taille de population désirée, soit le nombre de configurations de consistance minimale excède toujours la limite de taille, et la population est alors réduite par la méthode `decrease_PADE` (algorithme 3).

10.6 Contraintes et évolution temporelle

Limité dans un premier temps par une vision statique du timbre instrumental, notre système ne peut a priori engendrer que des textures orchestrales stables, sans modifications temporelles. Le compositeur souhaitant reproduire un timbre évoluant dans le temps n'a donc pour l'instant d'autre choix que de morceller la cible en autant d'intervalles que nécessaire, chercher une solution pertinente pour chaque segment indépendant, et enfin de « relier les piles du pont » en orchestrant manuellement les transitions.

Au cours de ce processus long et fastidieux, les orchestrations sont recherchées indépendamment les unes des autres, sans aucun contrôle sur le dégré de continuité à chaque transition. L'approche par contraintes peut ici apporter un regard différent sur la question de l'évolution temporelle. Selon la complexité du réseau de contraintes et le niveau de consistance d'une configuration donnée, la « réparation » de cette dernière par une méthode de satisfaction de contraintes engendre une configuration plus ou moins proche de l'état initial. On peut donc très bien considérer les contraintes comme un moyen de transformer un timbre, de le « déplacer » vers une autre région de l'espace. Celle-ci n'est alors plus caractérisée *explicitement* par un ensemble de descripteurs perceptifs, mais *implicitement*, par un jeu de relations logiques que les variables de l'écriture musicale doivent vérifier.

L'algorithme *CDCSolver* a été conçu de manière à parvenir à une configuration consistante en ne modifiant, à chaque étape du calcul, qu'une seule variable dans la configuration courante. En conservant la trace de la résolution, on obtient alors une succession de configurations « évoluant » de manière continue vers la consistance.

La figure 10.1 est un exemple d'une telle transformation. L'orchestration initiale, aux cordes, a été obtenue à partir d'une voix chantée. La recherche était contrainte par l'ensemble de contraintes suivantes :

Algorithme 8 Algorithme d'orchestration sous contraintes

Arguments: Une cible d'orchestration T, les paramètres de population N_{init}^{pop}, N_{max}^{pop}, N_{mate}, N_{max}^{pareto}, un nombre maximal d'itérations $iter_{max}$, une fonction de coût z.

1: $(X, p_{mute}) \leftarrow$ générer une population aléatoire de taille N_{init}^{pop} // (p_{mute} *est la proportion de configurations consistantes dans* X)
2: **pour** $\mathcal{K} \in X$ **faire**
3: $\quad \mathcal{K} \leftarrow$ `cdcsolver`$(\mathcal{K}, z)$ // (*Réparation des configurations inconsistantes*)
4: **fin pour**
5: $C \leftarrow$ **évaluer_population**(X, T)
6: $P \leftarrow$ `extract_pareto_set`(C)
7: $iter \leftarrow 1$
8: **tant que** $iter \leq iter_{max}$ **faire**
9: $\quad C \leftarrow$ **évaluer_population**(X, T)
10: $\quad \lambda \leftarrow$ tirer un jeu de poids
11: $\quad f \leftarrow \|C\|_\lambda$
12: $\quad f \leftarrow$ **pénaliser_fitness**(f, z)
13: $\quad M \leftarrow$ sélectionner N_{mate} individus dans X par tournoi binaire selon f
14: $\quad O \leftarrow$ crossover(M)
15: $\quad O \leftarrow$ 1-mutation(O, p_{mute}) // (*taux de mutation égal à la probabilité de consistance*)
16: $\quad C_O \leftarrow$ **évaluer_population**(O, T)
17: $\quad X \leftarrow X \cup O$
18: $\quad P \leftarrow$ `update_pareto_set`(P, C_O)
19: $\quad$ **si** $\#P > N_{max}^{pareto}$ **alors**
20: $\quad\quad P \leftarrow$ `decrease_PADE`$(P, N_{max}^{pareto}, \text{random})$
21: $\quad$ **fin si**
22: $\quad$ **si** $\#X > N_{max}^{pop}$ **alors**
23: $\quad\quad X \leftarrow X \setminus P$
24: $\quad\quad C \leftarrow$ **évaluer_population**(X, T)
25: $\quad\quad f \leftarrow \|C\|_\lambda$
26: $\quad\quad f \leftarrow$ **pénaliser_fitness**(f, z)
27: $\quad\quad X^* \leftarrow \{\mathcal{K} \in X \mid z(\mathcal{K}) = z_{min}\}$
28: $\quad\quad$ **si** $\#X^* \leq N_{max}^{pop} - \#P$ **alors**
29: $\quad\quad\quad X \leftarrow$ les $N_{max}^{pop} - \#P$ meilleurs éléments selon f // (*on élimine en priorité les éléments violant le plus les contraintes...*)
30: $\quad\quad$ **sinon**
31: $\quad\quad\quad X \leftarrow$ `decrease_PADE`$(X^*, N_{max}^{pop} - \#P, f)$ // (*...ensuite seulement ceux de densité maximale*)
32: $\quad\quad$ **fin si**
33: $\quad\quad X \leftarrow X \cup P$
34: $\quad$ **fin si**
35: $\quad iter++$
36: **fin tant que**
37: **retourner** P

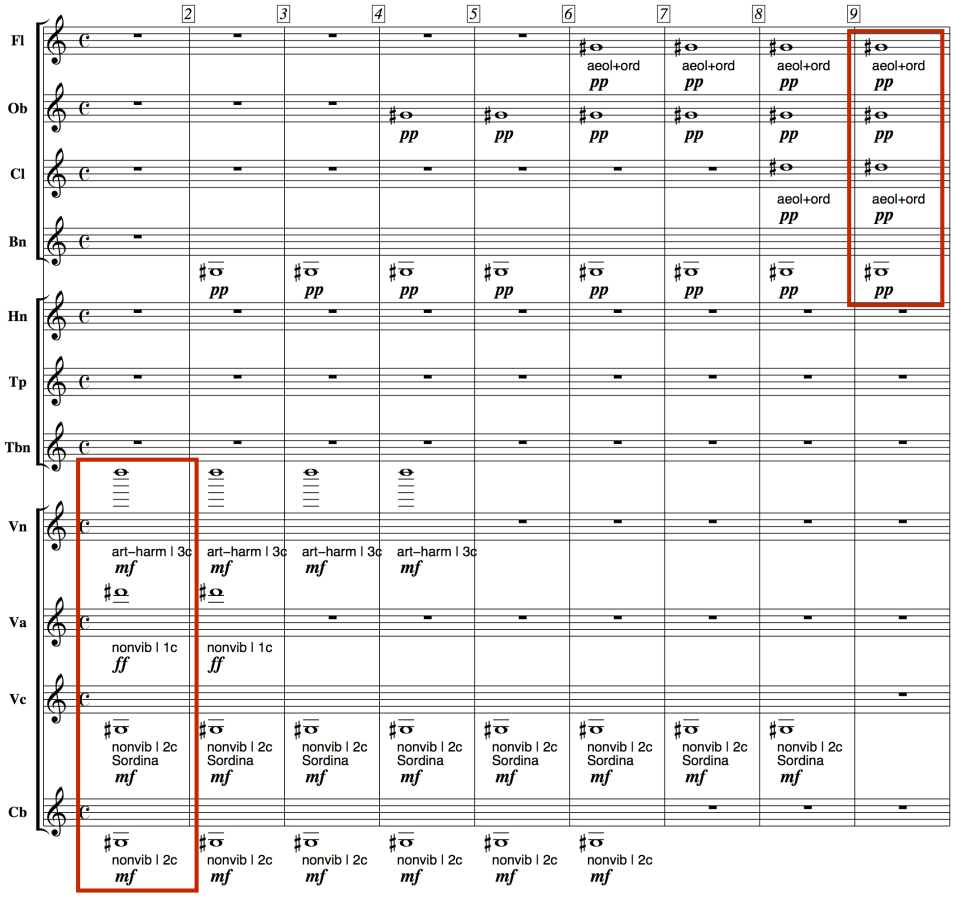

FIG. 10.1 – Passage continu d'un accord de cordes vers un accord de vents à l'aide de *CDC-Solver*. Partition en Ut.

<*attribut*=∅><*opérateur*="**size-min**"><*quantité*="**4**"><*valeur*=∅>
<*attribut*=∅><*opérateur*="**size-max**"><*quantité*="**4**"><*valeur*=∅>
<*attribut*="**famille**"><*opérateur*="**at-least**"><*quantité*="**4**"><*valeur*="**strings**">

L'évolution du timbre se fait par un « transfert » continu des cordes vers les vents, en modifiant la contrainte sur la famille d'instruments (les contraintes de cardinalité sont conservées de manière à abandonner progressivement les cordes) :

<*attribut*="**famille**"><*opérateur*="**at-least**"><*quantité*="**4**"><*valeur*="**woodwinds**">

La partition de la figure 10.1 permet de suivre la trace de cette transformation (chaque mesure correspond à une étape du calcul). L'orchestration finale étant limitée à quatre instruments, l'instanciation d'une nouvelle variable entraine systématiquement la violation de la contrainte de cardinalité maximale : une désinstanciation est alors nécessaire. L'entrée d'un instrument à vent donc est aussitôt suivie, à la mesure suivante, par le retrait d'un instrument à cordes. En ne conservant que les mesures impaires de la figure 10.1, on obtient un mouvement de timbre n'impliquant jamais que quatre instruments.

Du point de vue des contraintes, le choix d'un instrument à rajouter ou à retirer de l'orchestration est, à fonctions de coût égales, indifférent : à la seconde mesure par exemple, la même valeur de consistance est obtenue selon qu'on insère un basson, une flûte, une clarinette ou un hautbois dans l'orchestration. En revanche, la fitness permet de distinguer ces possibilités, en retenant la configuration de fitness maximale selon la fonction scalarisante en cours (cette dernière est calculée à l'aide des valeurs de critères obtenus pour l'orchestration initiale). La continuité du timbre est ainsi encouragée dans le passage d'une configuration à la suivante. L'exemple de la figure 10.1 peut être écouté en ligne sur le site : `http://recherche.ircam.fr/equipes/repmus/carpentier/phdsounds/`.

Chapitre 11

Tests de performances

Difficulté d'évaluer les méthodes multicritères
Nécessité d'une approche multi-expert
Evaluation de l'algorithme sur différentes classes de problèmes
Intérêt des principaux « modules » de l'algorithme d'orchestration

Nous terminons cette partie avec un chapitre consacré à l'évaluation quantitative de notre algorithme d'orchestration. Nous commençons par montrer en quoi l'évaluation des méthodes multicritères est une tâche particulièrement délicate, et discutons brièvement les approches en général adoptées dans la littérature. Nous proposons alors un cadre multi-expert dans lequels les performances de notre méthode sont mesurées selon plusieurs « points de vue ». Nous montrons l'intérêt, selon chaque point de vue, des différents « modules » de l'algorithme d'orchestration : crossover, mutation, réparation des configurations initiales par *CDCSolver*, maintien de la diversité par la méthode PADE.

11.1 Evaluation des méthodes multicritères

Les méthodes multicritères récentes reposent sur la dominance de Pareto. Nous avons vu à la section 5.2 que cet ordre partiel sur l'espace de recherche empêche de comparer toute paire de configurations. Cette indécidabilité entre deux solutions se répercute, de manière décuplée, sur la comparaison des performances de métaheuristiques.

Lorsqu'on a en effet recours à des méthodes incomplètes, celles-ci ne retournent en général jamais le front de Pareo théorique du problème, mais en donnent une approximation. Se pose alors la question de comparer deux approximations P_A et P_B retournées par deux métaheuristiques différentes A et B. D'un point de vue formel, on serait tenté de dire que la méthode A est meilleure que B si et seulement si tout élément de P_B est dominé par au moins un élément de P_A. Or cette condition est si forte qu'en pratique elle n'est, dans un sens comme dans l'autre, presque jamais observée. Il suffit qu'un seul élément de P_B ne soit dominé par aucun élément de P_A (et vice versa) pour que A et B soient incomparables.

La figure 11.1 représente deux approximations du front de Pareto obtenues avec deux méthodes multicritères différentes pour le même problème. A strictement parler, A et B sont incomparables : il existe des points de A non-dominés par B comme il existe des points de B non dominées par A. Il semble pourtant que l'on puisse en dire un peu plus en abandonnant le strict cadre de la dominance de Pareto. Il est en effet clair que A est meilleur que B sur

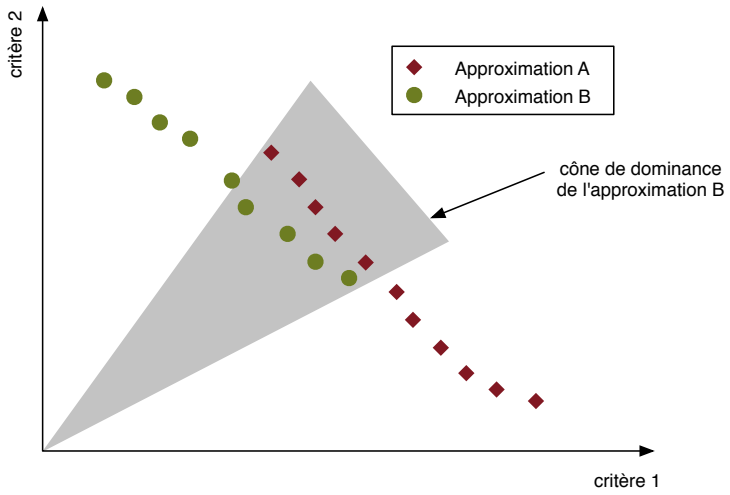

FIG. 11.1 – Deux approximations incomparables du point de vue de la dominance de Pareto

le critère 2, alors que B est meilleur que A sur le critère 1. En outre, il existe une portion de l'espace des critères (le cône de dominance en gris) sur laquelle B domine totalement A, alors que l'inverse n'est pas vrai. Autrement dit, il existe un ensemble de fonctions scalarisantes de Tchebycheff pour lesquelles les éléments de B obtiennent une meilleure fitness que ceux de A.

Si en général deux approximations ne peuvent être comparées selon la dominance de Pareto, on peut donc choisir de les considérer selon un « point de vue » particulier. Se concentrer sur une région de l'espace des critères peut constituer un tel point de vue. Comme nous aurons l'occasion de le voir, il en existe bien d'autres. Cette approche est celle retenue par la plupart des auteurs, qui en général ne font pas intervenir la dominance de Pareto (pour les raisons évoquées ci-dessus) dans leurs processus d'évaluation, mais proposent un certain nombre de « mesures de qualité ».

Une mesure de qualité est une fonction $\mathcal{M}$ (le plus souvent unaire ou binaire) sur l'ensemble des approximations, à valeurs dans $\mathbb{R}$. Elle s'accompagne d'une fonction d'interprétation, en général implicite, permettant de décider, en fonction des valeurs prises par $\mathcal{M}(A)$ et $\mathcal{M}(B)$ (ou $\mathcal{M}(A,B)$ et $\mathcal{M}(B,A)$ dans le cas d'une mesure binaire), si l'approximation A est meilleure ou moins bonne que B. Lorsque le front de Pareto théorique est connu, on peut par exemple utiliser la métrique $\mathcal{T}$ (appelée aussi *generational distance*) de Veldhuizen [vV99], qui renvoie la moyenne des distances euclidiennes entre chaque point de l'approximation et le vrai front.

On rencontre de nombreuses mesures de qualité dans la littérature. Outre la métrique $\mathcal{T}$, les plus connues sont l'*error ratio* de Veldhuizen [vV99], les métriques $\mathcal{S}$ (*hypervolume*) et $\mathcal{C}$ (*coverage*) [ZT98b] [ZT98a] ainsi que l'*epsilon indicator* [ZTL+03] de Zitzler et al., et les indicateurs R_1, R_2 et R_3 de Hansen et Jaszkiewicz [HJ97]. Certaines d'entre elles, que nous utilisons dans notre processus d'évaluation, seront détaillées à la section suivante. Pour les autres, nous renvoyons le lecteur aux travaux de Zitzler et al. [Zit99] [ZTL+03], Knowles et al. [KFTZ05], Hansen et Jaszkiewicz [HJ97], Grosan [Gro05] ou encore Deb [DJ02].

Ziztler et al. [ZTL+03] ont récemment proposé un cadre formel pour l'évaluation des mé-

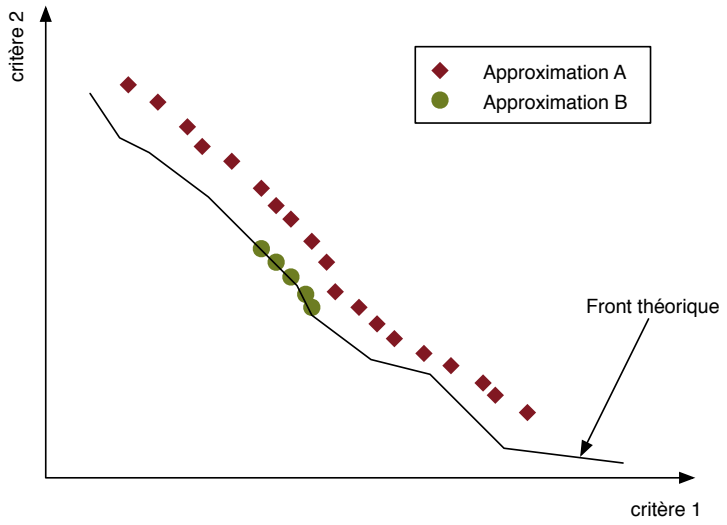

FIG. 11.2 – Ambiguité de la métrique $\mathcal{T}$

thodes multicritères et sont parvenus à un certain nombre de résultats théoriques fondamentaux. Les auteurs ont entre autres montré que dans la plupart des cas, il est impossible d'établir, à partir d'une mesure de qualité et sa fonction d'interprétation, qu'une approximation est meilleure qu'une autre. Considérons par exemple la métrique $\mathcal{T}$ (distance moyenne au front théorique) introduite ci-dessus. Il est intuitif de supposer que si $\mathcal{T}(B) < \mathcal{T}(A)$, alors l'approximation B est meilleure que A. La figure 11.2 montre que ce n'est pas aussi simble : si B est en effet à peine plus proche du front théorique que A, elle ne couvre toutefois qu'une faible partie du front de Pareto, alors que A donne une bonne approximation de son étendue.

Zitzle et al. [ZTL+03] ont montré que parmi toutes les méthodes d'évaluation proposées dans la littérature, seules deux mesures de qualité unaires et quatre mesures binaires permettent de conclure à la supérorité (au sens de Pareto) d'une approximation sur une autre, et ce dans un très petit nombre de cas. Il faut donc être extrêmement prudent lors de leur emploi. L'exemple de la figure 11.2 est frappant. Si les approximations sont comparées selon la distance au front théorique, alors B est incontestablement meilleure que A. Mais si on en juge à la diversité des solutions dans l'espace des critères, alors A l'emporte.

Lorsque deux approximations ne peuvent être comparées au sens de Pareto, il faut donc les évaluer selon plusieurs « points de vue » différents, afin de saisir précisément en quoi elles diffèrent. En d'autres termes, l'évaluation des méthodes multicritères est aussi un problème multicritère.

11.2 Mesures de qualité retenues

Nous proposons dans cette section un ensemble de mesures de qualité pour évaluer les performances de notre algorithme d'orchestration. Comme nous le verrons au paragraphe suivant, l'algorithme est d'abord comparé à une recherche purement aléatoire, puis différentes variantes sont confrontées deux à deux, afin de prouver la pertinence des principaux « piliers » de notre

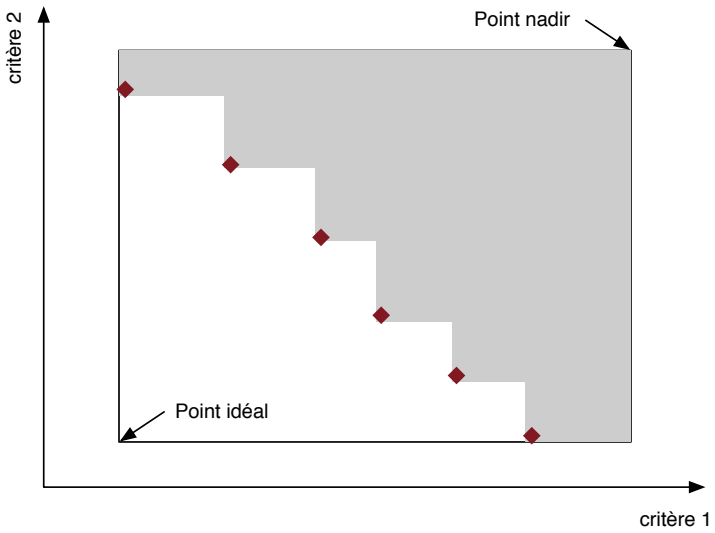

FIG. 11.3 – Hypervolume : portion de l'espace dominée par l'approximation

méthode de recherche : approche génétique, mutation, maintien de la diversité, réparation des solutions avec *CDCSolver*.

La procédure que nous mettons ici en place est « multi-expert ». Les approximations sont comparées selon plusieurs « points de vue » que nous n'agrégeons pas en une valeur unique. Nous montrons ainsi que selon les problèmes, modifier une partie de l'algorithme permet d'obtenir différents types et différentes qualités d'approximations.

11.2.1 Supériorité

Nous commençons par comparer les approximations selon la dominance au sens de Pareto. Ce calcul est basé sur celui de l'hypervolume $\mathcal{H}$ [Zit99], i.e. la partie de l'espace des critères dominée par l'approximation (en gris sur la figure 11.3). Dans le cas général pour un problème de minimisation, $\mathcal{H}$ est le volume délimité par l'union des polytopes de diagonale $\left[x^{max}; x^i\right]$, où x^{max} est le point nadir[1] et x^i un point de l'approximation. Lorsque l'espace de recherche n'est pas connu, ces points sont estimés par génération aléatoire d'un grand nombre de configurations (voir paragraphe 11.3).

Lorsque la dimension de l'espace des critères est supérieure à 2, le calcul de l'hyper volume peut devenir très complexe, et nécessite alors le recours à une méthode adaptée. L'algorithme HSO (*Hypervolume by Slicing Objectives*) [WHBH06] est le plus courant.

En pratique, nous calculons l'hypervolume binaire, qui consiste en la donnée de deux valeurs : $\mathcal{H}(A, B)$, la portion de l'espace dominée par A et non par B, et $\mathcal{H}(B, A)$, la portion de l'espace dominée par B et non par A. Zitzler et al. [ZTL+03] ont montré que si $\mathcal{H}(A, B) > 0$ et $\mathcal{H}(B, A) = 0$, alors A est meilleure que B, i.e. chaque élément de B est faiblement dominé[2] par un élément de A, et $A \neq B$. Dans les cas où les deux approximations peuvent être comparées

[1] Les coordonnées du point nadir sont les pires valeurs obtenues pour chacun des critères.

[2] x domine faiblement y si et seulement si $x \preceq y$ ou $x = y$.

selon la dominance de Pareto, le calcul de l'hypervolume binaire permet d'identifier lequel des deux ensembles est meilleur que l'autre.

11.2.2 Pseudo-dominance

Dans bien des cas toutefois, les approximations sont incomparables au sens de Pareto (voir section 11.1). Il nous faut alors convenir d'une notion plus souple de la dominance. Sur la figure 11.1 par exemple, certains éléments de B dominent des points de A, alors que l'inverse n'est pas vrai. L'approximation B peut donc être jugée préférable à A, car il existe des situations pour lesquelles B est meilleure (elles sont représentées par le cône de dominance en gris).

Nous définissons alors cinq prédicats $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, $\mathcal{P}_4$, $\mathcal{P}_5$, exprimant différents points de vue sur les relations de dominance entre les éléments des approximations :

- ($\mathcal{P}_1$) *Binary epsilon indicator :* $\epsilon(A, B)$ est le facteur unique par lequel il faudrait diviser l'ensemble des valeurs de critères de A pour que A domine B au sens de Pareto. Zitzler et al. donnent un algorithme efficace pour le calcul de ϵ dans [ZTL+03]. $\mathcal{P}_1(A, B)$ est vrai si et seulement si $\epsilon(A, B) < \epsilon(B, A)$.
- ($\mathcal{P}_2$) *Coverage :* $\mathcal{C}(A, B)$ est le pourcentage d'éléments de B dominés par A (voir par exemple [ZT98b]). $\mathcal{P}_2(A, B)$ est vrai si et seulement si $\mathcal{C}(A, B) > \mathcal{C}(B, A)$.
- ($\mathcal{P}_3$) *Binary R_1 :* Cette mesure introduite par Hansen et Jaszkiewicz [HJ97] évalue les approximations à l'aide d'un ensemble de fonctions scalarisantes de Tchebycheff (voir section 5.7). Etant donné un jeu de poids $(\lambda_i)_{1 \leq i \leq n}$, soient respectivement $f^*(A, \lambda_i)$ et $f^*(B, \lambda_i)$ les meilleures valeurs de $\| \cdot \|_{\lambda_i}$ obtenues sur respectivement A et B. $R_1(A, B)$ est alors égal au nombre fois où $f^*(A, \lambda_i) < f^*(B, \lambda_i)$. $\mathcal{P}_3(A, B)$ est vrai si et seulement si $R_1(A, B) > R_1(B, A)$.
- ($\mathcal{P}_4$) *Binary R_2 :* De façon similaire, R_2 est définie par :

$$R_2(A, B) = \frac{1}{N} \sum_{i=1}^{n} f^*(A, \lambda_i) - f^*(B, \lambda_i)$$

  et $\mathcal{P}_4(A, B)$ est vrai si et seulement si $R_2(A, B) < R_2(B, A)$.
- ($\mathcal{P}_5$) *Binary hypervolume :* Cette mesure a été définie au paragraphe précédent. $\mathcal{P}_5(A, B)$ est vrai si et seulement si $\mathcal{H}(A, B) > \mathcal{H}(B, A)$.

Au final, on dira que A *pseudo-domine* B si pour au moins trois des cinq $\mathcal{P}_i$ définis ci-dessus $\mathcal{P}_i(A, B)$ est vrai.

11.2.3 Convergence

Nous avons vu au chapitre 5 qu'une bonne métaheuristique pour l'optimisation multicritère devait assurer à la fois la convergence vers le front de Pareto théorique et la diversité des solutions optimales. Nous proposons donc une mesure adéquate pour chacun de ces aspects.

Nous mesurons la convergence vers la frontière efficiente à l'aide de la métrique $\mathcal{T}$ (*generational distance*) introduite au début de ce chapitre. $\mathcal{T}(A)$ est définie comme la moyenne des distances euclidiennes entre chaque point de l'approximation A et le font théorique. Lorsque ce dernier n'est pas connu, on utilise un ensemble de référence obtenu par génération aléatoire d'un grand nombre de configurations (voir paragraphe 11.3).

Nous dirons que A atteint un meilleur niveau de convergence que B si et seulement si $\mathcal{T}(A) < \mathcal{T}(B)$.

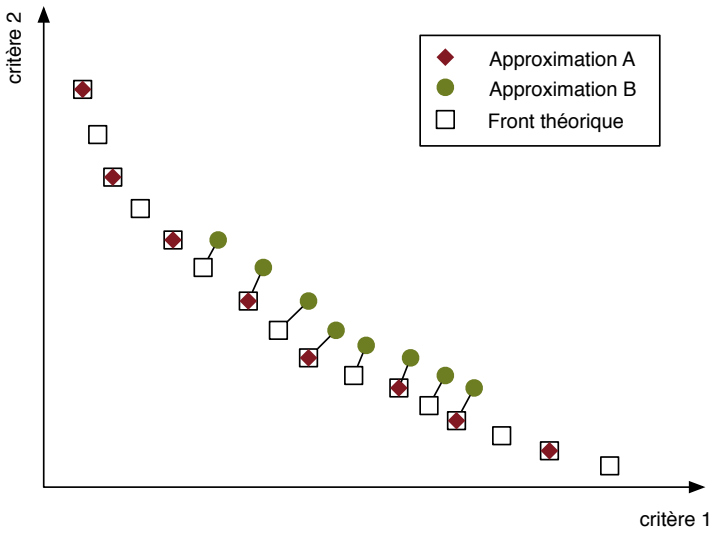

FIG. 11.4 – Deux approximations obtenant les mêmes valeurs de diversité pour la mesure de Grosan

11.2.4 Diversité

Grosan a proposé dans [Gro05] une mesure pertinente pour la diversité des solutions de l'approximation dans l'espace des critères. Pour chaque point de l'approximation, on « marque » le point du front théorique le plus proche au sens de la distance euclidienne. La diversité de l'approximation peut alors être estimée par le pourcentage de points marqués dans le front théorique.

Cette méthode intuitive et facile à mettre en œuvre possède un inconvénient majeur, illustré par la figure 11.4. Bien que beaucoup plus diversifiée, l'approximation A « marque » exactement le même nombre de points que B sur le front théorique. On peut corriger ce défaut en commençant par ramener l'approximation et le front théorique au même nombre d'éléments. Nous réduisons le plus grand des deux ensembles à la taille du plus petit à l'aide de la procédure `decrease_PADE` introduite à la section 9.4, puis calculons la diversité $\mathcal{D}$ proposée par Grosan. Là encore, lorsque le front théorique n'est pas connu, on utilise un ensemble de référence obtenu par génération aléatoire d'un grand nombre de configurations (voir paragraphe 11.3). Nous dirons alors que A est au moins aussi diversifié que B si et seulement si $\mathcal{D}(A) \geq \mathcal{D}(B)$, et vice versa.

Nous évaluons à la section suivante différentes variantes de l'algorithme d'orchestration. Plusieurs couples de variantes sont comparés sur un grand nombre d'instances, pour plusieurs classes de problèmes. Pour chaque couple et chaque classe les résultats sont présentés dans un tableau à quatre entrées (une par expert) dont la lecture est explicitée par le tableau 11.1.

Remarque 11.1. *Dans le cas de problèmes contraints, la question d'une mesure de consistance des solutions de l'approximation pourrait se poser. Les instances de test étant toutefois faiblement contraintes, le hasard des opérations génétiques permet toujours, dans le cadre de cette évaluation, de découvrir des configurations consistantes, même en l'absence d'une mé-*

TAB. 11.1 – Grille de lecture pour la comparaison multi-expert de deux approximations

Expert	Méthode A	Méthode B
Supériorité	Pourcentage des cas dans lesquels A est meilleure que B	Pourcentage des cas dans lesquels B est meilleure que A
Pseudo-dominance	Pourcentage des cas dans lesquels A pseudo-domine B	Pourcentage des cas dans lesquels B pseudo-domine A
Convergence	Pourcentage des cas dans lesquels A converge davantage que B	Pourcentage des cas dans lesquels B converge davantage que A
Diversité	Pourcentage des cas dans lesquels A est au moins aussi diversifié que B	Pourcentage des cas dans lesquels B est au moins aussi diversifié que A

thode de réparation telle que CDCDSolver. Assurés que les approximations ne contiennent que des solutions satisfaisant les contraintes, nous ne retenons donc pas de critère de consistance dans notre évaluation.

11.3 Performances de l'algorithme d'orchestration

Dans cette section, nous évaluons notre algorithme d'orchestration (algorithme 8) à l'aide du cadre multi-expert introduit ci-dessus. Nous utilisons à cette fin les cibles monophoniques et polyphoniques à quatre sons définies au chapitre 8. Ces problèmes sont assortis de contraintes de hauteur et de cardinalité. Pour les problèmes monophoniques, l'espace de recherche est limité aux sons de même hauteur que la cible. Il suffit donc de deux contraintes (*size-min* et *size-max* — voir chapitre 10) pour imposer que les solutions impliquent exactement quatre sons. Quant aux cibles polyphoniques, il faut encore rajouter quatre contraintes de hauteur pour avoir exactement une fois chaque note de la cible dans les solutions.

Sous ces conditions l'espace de recherche, et a fortiori le front de Pareto théorique ainsi que les points idéal[3] et nadir, sont connus. Il est donc possible d'évaluer l'algorithme d'orchestration sur des problèmes contraints à l'aide des mesures de qualité définies au paragraphe précédent. Afin également de mesurer les performances de l'algorithme sur des problèmes libres, nous exécutons notre algorithme sur chaque instance de test déchargée de ses contraintes. Dans ce cas toutefois, l'espace de recherche est trop grand pour pouvoir être calculé en un temps raisonnable. Nous nous contentons donc d'estimer le front de Pareto théorique (on parlera alors d'*ensemble de référence*), ainsi que les points idéal et nadir, par instanciation aléatoire de 10^6 configurations.

En résumé, l'algorithme d'orchestration est donc évalué sur :

- 500 problèmes monophoniques avec contraintes de cardinalité, pour lesquels l'espace de recherche est connu ;
- 500 problèmes polyphoniques avec contraintes de cardinalité et de hauteur, pour lesquels l'espace de recherche est connu ;
- 500 problèmes monophoniques non contraints, pour lesquels l'espace de recherche est inconnu ;

[3]Les coordonnées du point idéal sont les meilleures valeurs obtenues pour chacun des critères.

- 500 problèmes polyphoniques non contraints, pour lesquels l'espace de recherche est inconnu.

Dans le reste de cette section, les différentes versions de l'algorithme sont toujours paramétrées par les mêmes valeurs[4] : $N_{init}^{pop} = 200$, $N_{max}^{pop} = 500$, $N_{mate} = 50$ et $N_{max}^{pareto} = 100$. Le paramètre $iter_{max}$ est calculé de manière à ce que le nombre d'évaluations de fitness soit toujours égal à 20 000.

11.3.1 Algorithme de référence

Dans le cadre de notre évaluation multi-expert, nous comparons deux à deux différentes variantes de l'algorithme d'orchestration. Il nous faut donc un algorithme de référence auquel confronter une première version. Conformément avec la démarche adoptée au chapitre 8 pour la validation de la formulation du problème de l'orchestration et au chapitre 10 pour l'évaluation de *CDCSolver*, nous utilisons l'algorithme 9, purement aléatoire, comme référence. Outre la cohérence par rapport à nos processus de test précédents, ce choix se justifie de deux manières :

- Il n'existe pas à notre connaissance d'algorithme d'orchestration multicritère auquel se confronter.
- Notre algorithme d'orchestration est basé sur une heuristique faisant intervenir du hasard dans la résolution. Le comparer à un algorithme purement aléatoire permet donc d'évaluer sa capacité à « organiser » ce hasard.

Algorithme 9 Algorithme naïf d'orchestration (recherche aléatoire)

Arguments: Une cible d'orchestration T, un nombre d'évaluations de fitness N_f, une taille maximale de l'ensemble Pareto N_{max}^{pareto}, une fonction de coût z.

1: $X \leftarrow$ générer une population aléatoire de taille N_f
2: $X \leftarrow$ ne conserver que les solutions de X minimisant z
3: $C \leftarrow$ **évaluer_population**(X, T)
4: $P \leftarrow$ `extract_pareto_set`(C)
5: **si** $\#P > N_{max}^{pareto}$ **alors**
6: $\quad P \leftarrow$ `decrease_PADE`$(P, N_{max}^{pareto}, \text{random})$
7: **fin si**
8: **retourner** P

Dans la suite de cette section, cet algorithme sera simplement désigné par « `random` ».

11.3.2 Pertinence de l'approche génétique

Nous commençons par évaluer une version très rudimentaire de l'algorithme d'orchestration, dans laquelle :

- les configurations inconsistantes de la population initiale ne sont pas réparées par *CDCSolver* ;
- la mutation génétique est désactivée ;

[4] Notre intention est ici de démontrer l'intérêt qualitatif des différentes composantes de l'algorithme 8, non de déterminer des valeurs optimales de paramètres.

TAB. 11.2 – Performances d'un algorithme génétique élémentaire par rapport à une recherche aléatoire.

	monophonique contraint		*monophonique libre*	
	`v.1.0`	`random`	`v.1.0`	`random`
Supériorité	**18.40 %**	2.04 %	**8.79 %**	1.23 %
Pseudo-dominance	**84.25 %**	15.75 %	**61.15 %**	38.85 %
Convergence	**83.44 %**	16.56 %	**78.73 %**	21.27 %
Diversité	69.12 %	**84.25 %**	25.15 %	**78.32 %**
	polyphonique contraint		*polyphonique libre*	
	`v.1.0`	`random`	`v.1.0`	`random`
Supériorité	**56.00 %**	0.00 %	**27.00 %**	0.00 %
Pseudo-dominance	**98.40 %**	1.60 %	**89.40 %**	10.60 %
Convergence	**95.20 %**	4.80 %	**98.00 %**	2.00 %
Diversité	54.60 %	**82.60 %**	40.60 %	**60.20 %**

- la procédure `derease_PADE` est remplacée par une gestion naïve de la population (lorsque la taille de celle-ci dépasse N_{max}^{pop}, on conserve uniquement les $(N_{init}^{pop}+N_{max}^{pop})/2$ meilleures configurations au sens de Deb).

En d'autres termes, nous comparons au `random` un algorithme génétique élémentaire, que nous désignons par « `v.1.0` », dont les principales caractéristiques sont :

- des contraintes gérées exclusivement par la dominance au sens de Deb (voir définition 6.1) ;
- une sélection par tournoi binaire (voir paragraphe 5.6) ;
- une évolution par crossover uniforme (voir déinition 9.1) ;
- une gestion de la taille de la population par élimination des plus mauvais individus au sens de Deb.

Le tableau 11.2 rapporte les performances comparées des algorithmes `v.1.0` et random. Sur les quatre types de problèmes, les approximations retournées par `v.1.0` dominent fréquemment (au sens de Pareto) celles découvertes par l'algorithme aléatoire, et leurs performances en termes de pseudo-dominance et de convergence sont écrasantes, en particulier pour les problèmes polyphoniques. L'intérêt de l'approche génétique est patent : l'opérateur de crossover uniforme permet bien d'améliorer itérativement la qualité des configurations.

Lorsqu'on compare toutefois les deux méthodes selon un critère de diversité, la supériorité de l'algorithme génétique est moins évidente. Cependant, la capacité du hasard pur à produire des solutions très différentes les unes des autres ne doit pas vraiment nous surprendre. Mais, comme le montre la figure 11.5, cette diversité n'est rien sans l'optimalité : si les solutions de l'algorithme aléatoire sont en effet plus diversifiées que celles de l'algorithme génétique, en revanche ces dernières les dominent toutes au sens de Pareto. Il faut donc se méfier de la mesure de diversité, et toujours l'interpréter en regard des autres critères.

En outre, rappelons qu'il est connu que sans méthode explicite de maintien de la diversité (comme c'est le cas pour `v.1.0`), un algorithme génétique à tendance à évoluer vers une zone particulière de l'espace des critères, au détriment des autres. C'est le phénomène de dérive génétique (ou *genetic drift*, voir paragraphe 5.5.4). La figure 11.5 en est un exemple. Elle

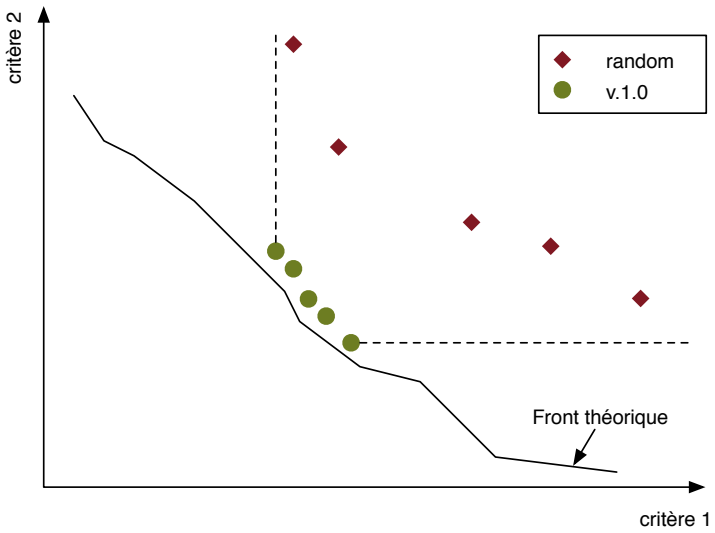

FIG. 11.5 – Diversité contre dominance

montre qu'un petit nombre de solutions optimales est souvent préférable à un large ensemble de configurations médiocres.

Nous aurons l'occasion de rediscuter des effets de l'aléatoire pur dans la suite de cette section. Nous verrons en outre que sous couvert de diversité apparente, une trop grande part d'aléatoire introduit en réalité du « bruit » dans le processus d'optimisation.

11.3.3 Pertinence de la mutation

Nous avons vu à la section 5.5.3 que dans un algorithme génétique le crossover et la mutation étaient des acteurs complémentaires de l'intensification et de la diversification de la recherche. Nous évaluons ici l'interêt de la mutation monopoint (voir définition 9.2) en termes de convergence et de diversité. Nous introduisons pour cela l'algorithme `v.1.1`, similaire à `v.1.0`, mais dans lequel le crossover est systématiquement suivi d'une mutation.

Le tableau 11.3 montre que la mutation monopoint systématique permet d'augmenter tous les critères de performance pour les problèmes libres. En revanche, si elle améliore la diversité des approximations en présence de contraintes, elle détériore nettement les autres critères, en particulier pour les instances polyphoniques pour lesquelles la consistance est plus difficile à maintenir. Nous pouvons ici à nouveau déceler les effets néfastes d'un « excès » d'aléatoire. En présence de contraintes, la mutation systématique à en effet tendance à engendrer des configurations inconsistantes. Sous couvert de diversité, l'évolution est compromise par un bruit qui perturbe la convergence vers l'optimalité.

Nous proposons donc de remplacer la mutation systématique de la version `v.1.1` par une mutation aléatoire (`v.1.2`) appliquée avec une probabilité adaptée à la difficulté du problème contraint (c'est la stratégie retenue dans la version finale de l'algorithme). En pratique, la probabilité de mutation est égale à la probabilité de consistance dans la population initiale

TAB. 11.3 – Pertinence de la mutation génétique

	monophonique contraint		*monophonique libre*	
	`v.1.1`	`v.1.0`	`v.1.1`	`v.1.0`
Supériorité	1.43 %	1.84 %	**3.48 %**	1.02 %
Pseudo-dominance	**59.30 %**	40.70 %	**62.58 %**	37.42 %
Convergence	47.24 %	**52.76 %** %	50.31 %	49.69 %
Diversité	**84.46 %**	78.32 %	**65.24 %**	40.90 %
	polyphonique contraint		*polyphonique libre*	
	`v.1.1`	`v.1.0`	`v.1.1`	`v.1.0`
Supériorité	0.40 %	**1.60 %**	**4.40 %**	0.00 %
Pseudo-dominance	42.80 %	**57.20 %**	**70.00 %**	30.00 %
Convergence	40.20 %	**59.80 %**	**62.00 %**	38.00 %
Diversité	**77.40 %**	74.40 %	**71.20 %**	30.40 %

TAB. 11.4 – Pertinence de la mutation aléatoire pour les problèmes contraints

	monophonique contraint		*polyphonique contraint*	
	`v.1.2`	`v.1.0`	`v.1.2`	`v.1.0`
Supériorité	**2.25 %**	0.82 %	1.60 %	1.40 %
Pseudo-dominance	**57.26 %**	42.74 %	**53.60 %**	46.40 %
Convergence	51.74 %	48.26 %	**54.00 %**	46.00 %
Diversité	79.35 %	79.75 %	76.00 %	76.20 %

X_{init} :

$$p_{mute} = \mathrm{P}(z(x) = 0) \ , \ x \in X_{init}$$

Pour les problèmes libres, les versions `v.1.2` et `v.1.1` sont équivalentes. Nous ne comparons donc `v.1.2` et `v.1.0` que sur les problèmes contraints.

Le tableau 11.4 montre qu'en présence de contraintes, la mutation aléatoire n'agit pas comme un opérateur de diversification, mais permet dans certains cas de parvenir à des approximations sensiblement meilleures, en termes de pseudo-dominance, que celles retournées par la version `v.1.0`. La mutation agit donc ici comme un opérateur d'intensification de la recherche : lorsqu'elle n'en détériore pas la consistance, elle permet dans certains cas d'affiner la qualité des configurations.

11.3.4 Maintien de la diversité par population aléatoire

Nous évaluons ici la possibilité d'encourager la diversité des approximations en introduisant périodiquement (une génération sur cinq) N_{mate} configurations aléatoires dans la population courante.

La lecture du tableau 11.5 est sans équivoque : La diversification par « excès » d'aléatoire a un effet dévastateur sur l'ensemble des mesures de performance. Pour les problèmes contraints, l'amélioration sensible de la diversité n'est pas significative en regard de la dégradation des autres indicateurs. Nous retrouvons là nos conclusions de la section 11.3.2 sur les conséquences du hasard pur.

TAB. 11.5 – Effets de l'introduction périodique de solutions aléatoires

	monophonique contraint		*monophonique libre*	
	`v.2.0`	`v.1.2`	`v.2.0`	`v.1.2`
Supériorité	0.41 %	**23.72 %**	0.82 %	**9.82 %**
Pseudo-dominance	7.98 %	**92.02 %**	22.90 %	**77.10 %**
Convergence	23.72 %	**76.28 %** %	28.83 %	**71.17 %**
Diversité	**79.96 %**	76.69 %	43.56 %	**63.60 %**
	polyphonique contraint		*polyphonique libre*	
	`v.2.0`	`v.1.2`	`v.2.0`	`v.1.2`
Supériorité	0.00 %	**56.40 %**	0.00 %	**13.80 %**
Pseudo-dominance	2.00 %	**98.00 %**	16.40 %	**83.60 %**
Convergence	13.80 %	**86.20 %**	14.60	**85.40 %**
Diversité	**75.60 %**	63.80 %	24.00 %	**76.80 %**

TAB. 11.6 – Pertinence de la réparation des solutions initiales pour les problèmes contraints

	monophonique contraint		*polyphonique contraint*	
	`v.3.0`	`v.1.2`	`v.3.0`	`v.1.2`
Supériorité	**13.70 %**	0.41 %	**1.80 %**	0.20 %
Pseudo-dominance	**58.28 %**	41.72 %	**56.60 %**	43.40 %
Convergence	51.33 %	48.67 %	48.60 %	51.40 %
Diversité	**78.73 %**	74.64 %	**81.80 %**	74.60 %

11.3.5 Pertinence de la réparation des solutions initiales

Nous revenons donc à la version `v.1.2` de l'algorithme, et y rajoutons la réparation des configurations inconsistantes dans la population initiale à l'aide de *CDCSolver*. De façon évidente, cette dernière version (`v.3.0`) est comparée à `v.1.2` sur les problèmes contraints uniquement.

Le tableau 11.6 montre que sans altérer la convergence, la réparation des solutions dans la population initiale permet d'augmenter de manière significative la diversité et la pseudo-dominance des approximations, voire la dominance au sens de Pareto (supériorité). Il est donc permis de supposer que dans la version `v.1.2`, une partie du temps de calcul est absorbée par la recherche de la consistance, là où la version `v.3.0` peut se contenter de maintenir la consistance établie par *CDCSolver* en début d'exécution.

11.3.6 Maintien de la diversité par la méthode PADE

Nous terminons par évaluer l'intérêt de la méthode PADE (*Population size Adapted Densisy Estimation* — voir sections 5.8 et 9.4)) pour le maintien de la diversité au sein de la population. La version obtenue (`v.4.0`) correspond à l'algorithme d'orchestration présenté au chapitre 10.

D'après le tableau 11.7, le maintien explicite d'une densité homogène dans la population permet d'obtenir des approximations beaucoup plus diversifiées dans le cas de problèmes

TAB. 11.7 – Intérêt de la préservation de la densité selon la méthode PADE

	monophonique contraint		*monophonique libre*	
	`v.4.0`	`v.3.0`	`v.4.0`	`v.3.0`
Supériorité	**4.50 %**	1.02 %	**5.11 %**	0.61 %
Pseudo-dominance	50.72 %	49.28 %	**65.44 %**	34.56 %
Convergence	31.70 %	**68.30 %**	50.72 %	49.28 %
Diversité	**83.25 %**	75.05 %	**76.29 %**	28.63 %
	polyphonique contraint		*polyphonique libre*	
	`v.4.0`	`v.3.0`	`v.4.0`	`v.3.0`
Supériorité	0.20 %	**1.20 %**	0.80 %	0.00 %
Pseudo-dominance	39.20 %	**60.80 %**	**73.40 %**	26.60 %
Convergence	38.80 %	**61.20 %**	**54.80 %**	45.20 %
Diversité	**76.60 %**	74.60 %	**87.60 %**	12.40 %

libres. On constate également une amélioration sensible de la pseudo-dominance, voire de la convergence ou de la dominance de Pareto (supériorité) selon les types de problèmes.

L'intérêt de la méthode est en revanche moins évident pour les problèmes contraints, pour lesquels une légère augmentation du critère de diversité se fait au prix d'une dégradation significative de la convergence, et même des deux autres critères pour les instances polyphoniques. Ce phénomène est peut-être dû au maintien d'une population de taille fixe par la méthode PADE, là où la version `v.3.0` ramène la cardinalité de la population à $(N_{init}^{pop}+N_{max}^{pop})/2$ dès que celle-ci dépasse N_{max}^{pop}. Dans la version `v.3.0`, la taille de la population évolue ainsi en dents de scie, avec une purge périodique où les configurations les moins consistantes sont éliminées. A l'opposé, la version `v.4.0` peut maintenir un grand nombre de configurations inconsistantes dans la population, perturbant potentiellement l'évolution.

Magré tout, la diversité des solutions de l'approximation est sans doute un critère à privilégier dans un cadre de création musicale. C'est donc pour l'instant la version `v.4.0` qui est implémentée dans le prototype d'aide à l'orchestration qui fait l'objet de la dernière partie de ce document.

Conclusion

Nous avons montré, à travers un processus d'évaluation multi-expert, que notre algorithme d'orchestration converge rapidement vers la frontière efficiente tout en maintenant la diversité au sein de la population et parmi les solutions optimales. Au cours de cette évaluation, les paramètres de notre algorithme n'ont pas été modifiés. On sait pourtant que l'optimisation par algorithmes génétiques demande une attention particulière dans le choix de paramètres tels que la taille de la population et la taille du bassin de reproduction. Il conviendrait donc de reprendre cette évaluation avec différentes valeurs pour ces paramètres.

Gardons toutefois à l'esprit que le but ultime de notre méthode est avant tout de découvrir une solution qui satisfasse les exigences esthétiques de l'utilisateur. En garantissant l'équilibre entre convergence et diversité, on peut donc espérer identifier rapidement une configuration pertinente, à partir de laquelle la recherche peut être intensifiée par inférence des préférences

d'écoute du compositeur (voir paragraphe 7.2.6). Le caractère interactif et supervisé de l'optimisation dans une situation de création musicale doit donc tempérer la nécessité de découvrir les valeurs optimales pour les paramètres de l'algorithme. Si le choix de ces valeurs peut dans certains cas être crucial, il n'engendre souvent qu'une légère amélioration de performances si la méthode d'optimisation respecte les deux principes fondamentaux de la recherche multicritère : convergence et diversité. Si ce gain peut être déterminant dans le cas de problèmes de référence pour lesquels il existe de nombreux benchmarks, ou dans un contexte industriel où une amélioration infime peut avoir des conséquences énormes en termes de coût, il ne consitue en rien une priorité pour l'aide à l'orchestration. Ce pourquoi nous avons évalué notre méthode de manière plus qualitative que quantitiative, en montrant l'intérêt de ses differentes composantes dans un cadre de création.

Résumé de la deuxième partie

L'aide à l'orchestration est abordée comme la recherche, au sein de grandes banques de sons instrumentaux, de combinaisons d'échantillons s'approchant le plus possible d'un timbre cible. Afin d'une part de tenir compte du caractère multidimensionnel de la perception du timbre, d'autre part de traiter directement le problème combinatoire des mélanges, nous ramenons l'orchestration à une tache d'optimisation combinatoire multicritère sous contraintes.

La caractérisation des possibilités timbrales des instruments et de leurs alliages est abordée à travers un paradigme de description du signal. L'analyse automatique de grandes banques d'échantillons sonores permet de condenser une connaissance synthétique du son instrumental en un ensemble de descripteurs perceptifs, de prévoir les propriétés acoustiques des mélanges de timbre et d'évaluer leur similarité avec un timbre cible selon plusieurs dimensions. A travers ce cadre formel, les combinaisons instrumentales ne sont plus identifiées par les variables de l'écriture musicale, mais par un vecteur de distances perceptives, dans un espace où un problème d'optimisation multicritère peut être posé. Nous montrons sur un ensemble de problèmes de petite taille qu'une description spectro-harmonique du timbre permet de traiter un ensemble conséquent de cas d'orchestration et prouvons que les solutions optimales du problème d'optimisation correspondent à des configurations pertinentes d'un point de vue perceptif. Nous validons ainsi l'apport des approches descriptive, multicritère et combinatoire pour aborder la complexité de l'orchestration.

Un algorithme génétique multicritère est introduit, dans lequel les propositions d'orchestration sont représentées par des N-uplets d'entiers, ce qui permet d'ignorer les contraintes liées aux limitations de l'instrumentarium utilisé. Les configurations sont évaluées à l'aide de fonctions scalarisantes de Tchebycheff à poids aléatoires, garantissant l'optimisation simultanée de l'ensemble critères perceptifs. Une méthode d'estimation de la densité locale au sein de la population assure en outre le maintien de sa diversité et empêche une convergence prématurée vers une zone particulière de l'espace des critères.

Un cadre théorique pour l'expression et la gestion de contraintes globales sur les variables symboliques de l'orchestration est également proposé. La découverte de solutions consistantes est prise en charge par l'heuristique de recherche locale *CDCSolver*, qui s'appuie sur la distinction entre contraintes de design et containtes de conflit, et dont l'efficacité est prouvée sur des problèmes de petite taille. Dans certains cas, *CDCSolver* peut assurer, en complément de l'algorithme génétique, une partie de l'optimisation. Une intégration dans l'algorithme d'orchestration pour la gestion de problèmes contraints est proposée, de manière à répartir équitablement le temps de calcul : *CDCSolver* intervient comme procédure de réparation tandis que l'évolution vers les configurations consistantes est encouragée par une dominance de Pareto étendue. Nous montrons enfin comment *CDCSolver* peut être utilisé pour des transformations continues de timbre.

Les performances de l'algorithme d'orchestration sont évaluées dans un premier temps sur

un ensemble de problèmes contraints de petite taille pour lesquels les fronts de Pareto théoriques sont connus, puis sur des instances plus conséquentes pour lesquelles le front théorique est estimé par une approximation de référence. Un système d'évaluation multi-expert est proposé pour comparer différentes versions de l'algorithme selon différents critères : dominance faible, pseudo-dominance, convergence, diversité. Nous prouvons tout d'abord la supériorité d'une version naîve de l'algorithme par rapport à une recherche purement aléatoire, et justifions l'apport de la mutation génétique, de la réparation des configurations inconsistantes, et du maintien de la diversité.

Troisième partie

Outil d'aide à l'orchestration

Chapitre 12

Prototype actuel

Un premier outil d'orchestration développé sous MATLAB©
L'apport des technologies existantes à l'IRCAM
En quoi les limites actuelles conditionnent l'esthétique

Cette troisième partie est consacrée au prototype d'outil d'orchestration, développé par nos soins en parallèle de nos recherches. Le choix d'un environnement de développement s'est rapidement porté sur MATLAB©, pour au moins quatre raisons majeures :

1. MATLAB© est un langage multi-plateforme. Le prototype est donc facilement portable et utilisable par des compositeurs ne travaillant pas nécessairement sur des machines Apple™.
2. MATLAB© est un langage interprété. La modification de certains éléments du projet n'impose donc pas de recompiler l'ensemble.
3. Nos recherches ont été menées parallèlement à la thèse de Damien Tardieu, doctorant dans l'équipe Analyse-synthèse des sons à l'IRCAM et collaborant avec nous sur l'aide à l'orchestration. Les travaux de ce dernier ont porté sur la caractérisation du timbre instrumental à l'aide de descripteurs du signal audio (voir section 2.6) et de l'élaboration de modèles d'instruments (voir section 7.1.2 et [TR07]). Or, MATLAB© est un outil de référence en traitement du signal. Afin de favoriser la communication et l'interopérabilité de nos recherches, il était donc préférable de travailler dans le même environnement.
4. Si MATLAB© est majoritairement utilisé pour le calcul numérique, il s'agit également d'un outil de développement pratique et puissant pour le prototypage, notamment pour la conception d'interfaces graphiques.

12.1 Architecture

Notre outil d'aide à l'orchestration répond au schéma général de la figure 12.1. La connaissance instrumentale y est représentée par la base de données de descripteurs et d'attributs obtenus respectivement par extraction automatique et indexation manuelle du signal audio, à partir des échantillons de SOL [BBHL99], la banque de sons instrumentaux enregistrée à l'IRCAM (voir paragraphe 7.1.2).

La notion de cible à orchestrer a été introduite au paragraphe 7.1.1. Elle est donnée en entrée par le compositeur, par le biais d'un son enregistré dans le cas d'une orchestration

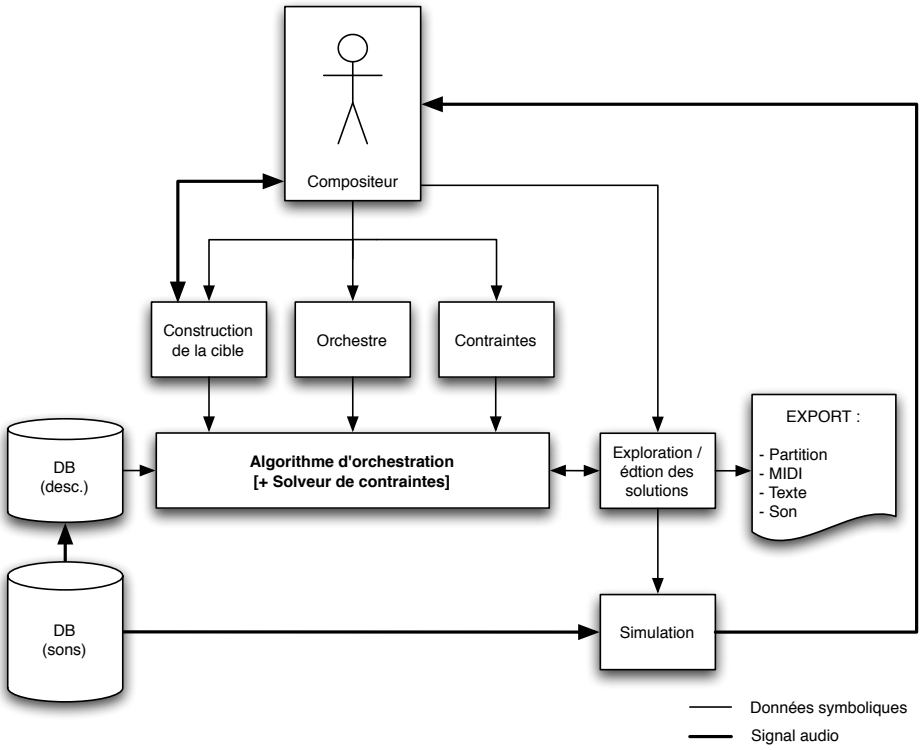

FIG. 12.1 – Schéma général de l'outil d'orchestration

imitative ou d'un son de synthèse pour une orchestration générative. Le compositeur doit en outre préciser la composition de l'orchestre qu'il utilise, et est libre de spécifier un ensemble de contraintes supplémentaires sur les attributs des solutions qu'il souhaite obtenir (voir section 7.5 et chapitre 10).

L'algorithme d'orchestration, que nous avons présenté au chapitre 9, va alors chercher selon les critères de timbres proposés en 8.1.2 un ensemble de configurations efficientes, en optimisant les distances entre leurs descripteurs et ceux de la cible. Au cours de cette recherche, un solveur de contraintes (voir chapitre 10) s'assure que les configurations proposées sont consistantes avec le réseau de contraintes définies par le compositeur.

Une fois la recherche terminée, une interface de navigation permet au compositeur d'explorer l'ensemble des configurations efficientes trouvées par le système, selon plusieurs points de vue. Espaces de décisions, espace des descripteurs et espace de critères (voir section 7.6) sont simultanément accessibles à l'utilisateur, et ce de façon croisée (les déplacements dans un espace se propageant dans les deux autres). Couplé à cette interface, un échantillonneur permet, grâce aux échantillons instrumentaux de la base de données, de simuler les propositions d'orchestration.

L'interface de navigation incorpore les mécanismes d'inférence des préférences de l'utilisateur introduits à la section 7.2.4. Le compositeur a la possibilité de relancer la recherche à partir d'une solution intermédaire ; dans ce cas, le système privilégie la combinaison de critères qui a implicitement présidé au choix de cette solution. Tout se passe donc comme si la recherche s'intensifiait dans son voisinage. L'utilisateur peut également éditer manuellement les solutions, ainsi que les transformer par ajout de contraintes suppémentaires.

La figure 12.1 montre que le signal audio intervient de façon privilégiée dans la communication entre le compositeur et l'outil d'orchestration, que ce soit au niveau de la construction de la cible ou de l'exploration des solutions. Ce « primat du sonore » permet d'éviter le recours à un vocabulaire peu précis pour la caractérisation du timbre, et de se cantoner aux symboles usuels de l'écriture musicale (hauteurs, intensités, mode de jeu,...). De plus amples détails sur l'interaction avec l'outil seront donnés au chapitre 13.

12.2 Composantes héritées des outils existants

Nous avons vu au paragraphe 4.1 que l'aide à l'orchestration est un domaine de recherche au carefour de nombreuses disciplines. Le contrôle du timbre dans un environnement de composition assistée par ordinateur impose d'établir une connexion entre la représentation symbolique des structures musicales et les connaissances récentes en psychoacoustique et en description du signal audio. En outre, de larges banques d'échantillons sonores sont nécessaires pour la modélisation des possibilités instrumentales, et éventuellement la simulation d'orchestrations.

Le développement d'un environnement d'aide à l'orchestration passe donc par l'intégration de toutes ces compétences au sein d'un même outil. C'est une tache évidemment impossible à accomplir seul dans le cadre d'une thèse de recherche. Il n'est donc pas surprenant que dans sa version actuelle, notre prototype utilise un certain nombre d'outils existants et nécessite l'exécution — en parallèle de MATLAB© — d'autres programmes. Il s'agit là d'une contrainte lourde : en informatique musicale, les utilisateurs n'adoptent en général une technologie que si l'ensemble des opérations qui lui sont rattachées sont réalisables dans un unique environnement, où si cette technologie communique de manière souple (par exemple via une architecture maître/esclave, en général sous forme de « plug-in ») avec un environnement de référence.

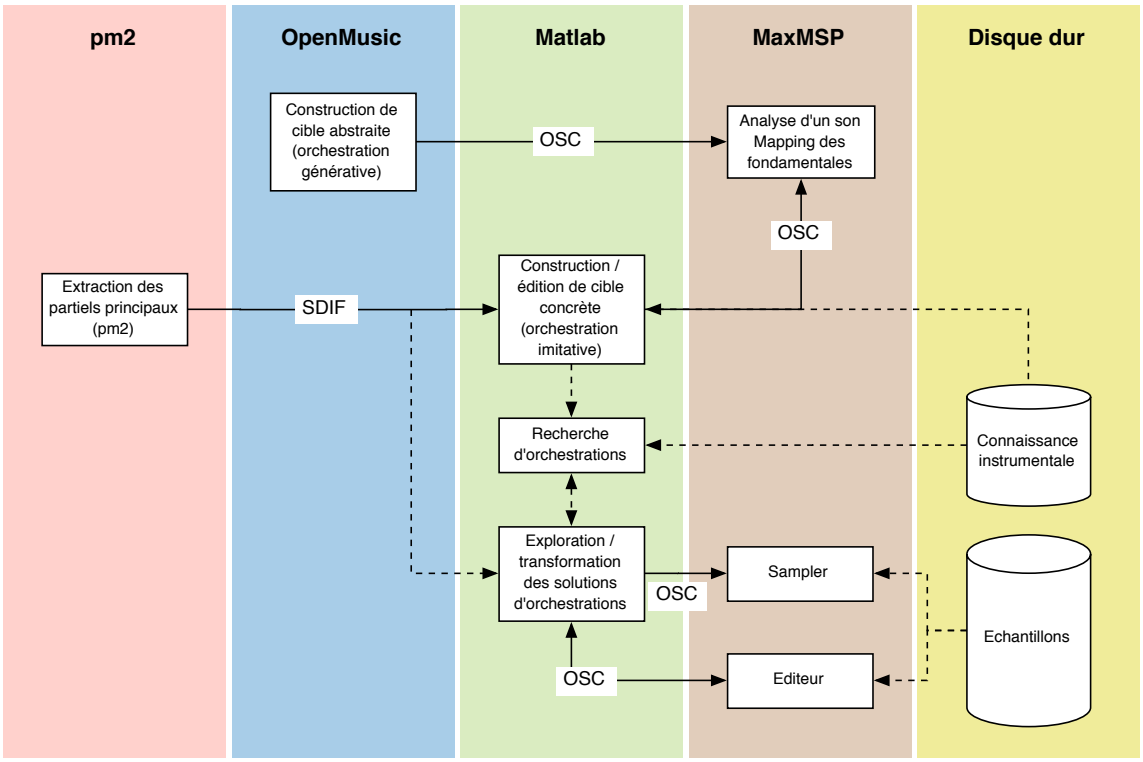

FIG. 12.2 – Prototype d'aide à l'orchestration : espace disque et outils existants nécessaires. Les communications internes sont en pointillés ; celles par fichier ou par OSC sont en traits pleins.

Nous sommes aujourd'hui loin de cette encapsulation idéale des technologies. Nous ne pouvons pas même dire si, dans sa version finale, l'outil d'orchestration sera un programme à part entière (*standalone*), un « plug-in », ou une bibliothèque (mais dans quel langage ?) dans l'environnement de CAO OPENMUSIC [Ago98]. En vérité, nous pensons qu'il faut d'abord en passer par une phase conséquente d'expérimentation auprès des compositeurs avant de décider d'une architecture finale et d'une stratégie de développement ; car ces dernières devront être suffisamment souples et visionnaires pour intégrer les résultats de recherches futures.

En attendant, nous tirons parti des technologies existantes et implémentons, de manière rapide et sûre, les modules « satellites » de notre outil au sein d'environnements tels qu'OPENMUSIC ou MAX/MSP [Puc91]. Quant à MATLAB©, nous y réservons le cœur de notre recherche : optimisation combinatoire multicritère (chapitre 9), gestion des contraintes (chapitre 10), et interfaces de navigation dans les propositions d'orchestration (chapitre 13).

La figure 12.2 résume l'ensemble des interactions entre MATLAB© d'une part, les technologies et bases de données disponibles aujourd'hui à l'IRCAM d'autre part.

Comme on peut le constater, la majeure partie des modules externes ont été développés dans MAX/MSP [Puc91], un environnement conçu à l'IRCAM par Miller Puckette à la fin des années 80. Comme OPENMUSIC, MAX/MSP établit une correspondance entre un programme et sa représentation graphique sous forme de boîtes connectés entre elles au sein d'un « patch ».

Mais là où OpenMusic permet, à travers des structures de données complexes, un haut niveau d'expressivité pour la programmation, Max/MSP propose un langage plus pauvre, mais d'une très grande efficacité pour le contrôle et le traitement du signal en temps réel. OpenMusic est donc préféré pour la composition, tandis que Max/MSP est utilisé avant tout dans des contextes performatifs.

En ce qui nous concerne, Max/MSP s'est naturellement imposé pour la conception des interfaces dans lesquelles circulent des flux audio : analyse du son cible, simulation des propositions d'orchestration (*sampler*), édition des solutions. Dans les deux derniers cas, la difficulté est de déclencher la lecture simultanée de plusieurs échantillons de la base de sons. Cet ordre peut être passé aussi bien depuis Matlab© que Max/MSP, et la lecture doit commencer aussitôt après, sans quoi l'interaction avec le compositeur à travers une écoute des propositions d'orchestration n'est pas possible. La rapidité de Max/MSP en termes d'accès disque et de chargement des *buffers* audio et sa grande facilité d'utilisation (due au paradigme de programmation visuelle) permettent de satisfaire ces exigences tout en économisant de fastidieuses heures de développement. Par ailleurs l'aspect modulaire de Max/MSP permet d'imaginer un certain nombre de traitements possibles (individuels ou collectifs) sur les échantillons impliqués dans une orchestration : modification continue de paramétres de jeu (intensité, hauteur...), modulations (vibratos, tremolos...), déplacements dans l'espace (à l'aide du spatialisateur *Spat* [Jot97] par exemple). Ces « mouvements » permettent, à travers un processus interactif d'essais/erreurs, de créer des timbres dynamiques ; ils correspondent à des paramètres traditionnels de l'écriture musicale. D'autres traitements sont également envisageables, si le son des échantillons est pensé comme une « matière ». A l'instar du compositeur Gérard Buquet (voir section 14.5), on peut en effet utiliser l'aide à l'orchestration comme un environnement de production sonore.

La souplesse et la rapidité dans la gestion des flux audio ne sont pas les seules raisons du choix de Max/MSP. Cet environnement comporte également un certain nombre d'« objets » prédéfinis, tels que le clavier de piano ou la représentation d'un accord sous forme de partition, qui permettent d'agir de façon rapide et intuitive sur le paramétrage d'une orchestration. Ces éléments sont particulièrement utiles pour le « mapping de fondamentales » opération qui consiste à « expliquer » le spectre de la cible par un ensemble de spectres harmoniques (pour plus de détails, voir sections 13.1 et A).

Les communications entre Matlab© et Max/MSP se font par échange de fichiers temporaires ou par messages OSC (*OpenSound Control*) [WFM03]. Développé au CNMAT[1], OSC est un protocole de communication entre ordinateurs, synthétiseurs et contrôleurs multimédia, basé sur la norme UDP[2] et optimisé pour les technologies organisées en réseaux.

Nous avons insisté au paragraphe 7.1.1 sur la nécessité de trouver une alternative à l'orchestration imitative, dans laquelle le timbre cible est donné par le compositeur sous forme d'un son enregistré. Nous avons proposé pour cela le concept d'orchestration générative : à partir d'un matériau musical symbolique (un accord par exemple), le compositeur va générer un son de synthèse dont les caractéristiques sont déterminées par des paramètres de haut niveau. Modifiable à souhait jusqu'à l'obtention d'un timbre satisfaisant, ce son est alors considéré comme la cible d'un problème d'orchestration imitative. D'un point de vue formel, il s'agit d'une abstraction : lisse et froid, ce son de synthèse est un « portrait-robot » qui cristallise les propriétés acoustiques du timbre recherché. Face à la difficulté de caractériser verbalement le

[1] *Center for New Music and Audio Technology*, Université de Californie, Berkeley.
[2] *User Datagram Protocol.*

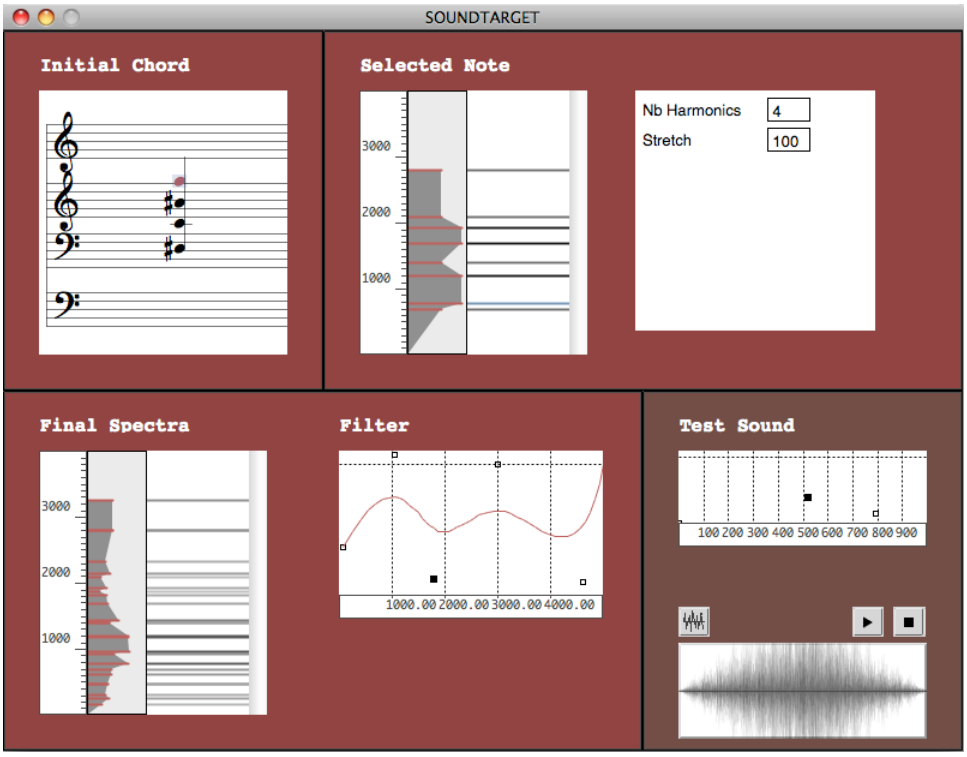

FIG. 12.3 – Construction d'une cible dans OPENMUSIC à l'aide de paramètres exclusivement symboliques (orchestration générative).

timbre (voir chapitre 2 et section 7.1.1), c'est encore le moyen de s'entendre sur une « idée de timbre ».

Cette piste de travail a été explorée en parallèle de nos travaux, en collaboration avec Jean Bresson, chercheur à l'IRCAM dans l'équipe Représentations musicales, et en charge notamment du développement d'OPENMUSIC. OPENMUSIC (OM) est un environnement de CAO basé sur la programmation visuelle et doté d'un haut niveau d'expressivité. Il permet la mise en relation d'objets musicaux (symboliques ou concrets) au travers d'un ensemble de « boîtes » interconnectées au sein d'un « patch », et formalisant un processus compositionnel (voir la figure 14.2 pour un exemple de patch OM).

Un objet `sound-target` (voir figure 12.3) a donc été implémenté dans OM : il prend en entrée un accord (objet `chord`) et propose un ensemble d'outils pour sa « spectralisation » : chaque note est assortie d'un spectre dont les caractéristiques (enveloppe, inharmonicité, fréquences et amplitudes des partiels...) sont aisément éditables. Un spectre complexe peut ainsi être construit, puis affiné par un filtre global avant d'être « rendu » par un moteur de synthèse additive dans CSOUND [Bou00]. Le compositeur peut ainsi « construire » le timbre recherché à travers un processus itératif. Une fois satisfait, le son cible est « envoyé » au système d'aide à l'orchestration par un message OSC.

L'extraction des descripteurs du son cible se fait d'une part grâce au programme *ircamdescriptor* de Geoffroy Peeters [Pee04] (développé en MATLAB©), d'autre pat via la commande *pm2* [Rod97] qui sert en partie de noyau au logiciel AUDIOSCULPT [BR05]. Dans l'état actuel

des choses, l'utilisateur a donc également besoin d'une licence pour *pm2*. Les résultats de cette analyse sont importés dans MATLAB© sous forme d'un fichier SDIF[3] [SW00].

En dernier lieu, notons que si la connaissance instrumentale (voir section 7.1.2) est stockée dans une structure de données dont la taille avoisine les 10 mégaoctets, en revanche l'ensemble des échantillons instrumentaux nécessaires aux simulations d'orchestration occupe aujourd'hui un espace mémoire de plusieurs gigaoctets, et ce pour une représentativité seulement partielle des possibilités instrumentales (voir paragraphe 8.1.1).

Il est évident que la simulation par lecture de simples fichiers son n'est pas extensible à l'infini. Deux solutions s'offrent alors à nous :

1. Ne retenir qu'un échantillon par zone de hauteur et de dynamique où le timbre varie peu, et recourir « à la volée » à des techniques de transposition et d'ajustement de volume. C'est la stratégie adoptée par la plupart des *samplers*.
2. Développer un moteur de synthèse instrumentale puissant et réaliste, mais cela sort totalement de notre domaine de compétences. Il serait alors plus raisonnable de convenir d'un format standard de données (de type MIDI[4]) grâce auquel les propositions d'orchestrations pourraient être « interprétés » par un environnement de synthèse existant, quoique la généricité d'un tel format n'a rien d'évident. Cette question reste aujourd'hui en suspens...

12.3 Considérations sur les limites actuelles

Notre outil d'aide à l'orchestration est encore loin de répondre à toutes les situations d'« écriture du timbre » : dans sa version actuelle, il ne permet de traiter qu'une partie des scénarii imaginés par les compositeurs (voir section 4.2). Il convient toutefois de distinguer, parmi ces limites, celles qui tiennent à l'état de nos connaissances et à l'avancement de nos recherches, de celles imposées par notre approche de l'orchestration assistée par ordinateur.

12.3.1 Description du timbre

Nous ne pouvons nier que la description sonore utilisée dans notre système ignore de nombreux aspects du timbre. En outre, la pertinence d'une approche multicritère utilisant des descripteurs essentiellement harmoniques ou spectraux — donc potentiellement redondants — mérite d'être posée. Nous avons toutefois montré au chapitre 8 que lorsque la cible à orchestrer peut être décrite de manière uniquement spectrale[5], notre approche se justifie malgré la potentielle corrélation entre les différentes dimensions. Quant aux aspects temporels ou spectro-temporels du timbre, nous renvoyons le lecteur aux travaux de Damien Tardieu [TR07] portant sur la caractérisation du timbre instrumental à l'aide de modèles statistiques, en rappelant que notre système devrait en bénéficier d'ici peu. Pour l'heure, nous ne pouvons que nous restreindre à une description du son accessible à notre expertise.

[3] *Sound Description Interchange Format*. Il s'agit d'un format standard pour la description des sons, défini par l'IRCAM et le CNMAT.

[4] *Musical Instrument Digital Interface (voir [SF97])*.

[5] Les timbres utilisés comme cibles de test aux chapitres 8 et 11 sont des textures polyphoniques sans variations temporelles, et dont le spectre est relativement stable dans le temps.

12.3.2 Timbre et hauteur

Tous les sons de la base de données constituant la connaissance instrumentale sont harmoniques et monophoniques. L'extraction des principaux partiels résolus de la cible permet de déduire la contribution de chaque son de la base à un spectre « simplifié », et cette contribution se fait actuellement par le prisme de la hauteur (voir section A en annexe). Dans notre système d'aide à l'orchestration, la hauteur est donc implicitement considérée comme un attribut privilégié, alors qu'une part significative de la musique contemporaine prétend au contraire s'en affranchir.

Peut-il en être autrement ? « Le problème, lorsqu'il s'agit d'évaluer le rôle du timbre en soi, est que, pour la plupart des musiques occidentales dont l'importance esthétique est avérée, l'utilisation du timbre est largement subordonnée aux structures de hauteurs et de rythmes. » (McAdams & Saariaho, *Qualités et fonctions du timbre musical*, in [Bar85]). Erickson [Eri75] confirme ce point du vue, et va jusqu'à douter qu'on puisse se passer de la hauteur : « La hauteur complique énormément la situation, et il n'est pas sûr que l'on puisse envisager des situations où elle n'intervienne pas de façon significative. »

En se tournant vers le timbre (voir chapitre 2), la pensée musicale a certes tempéré l'importance de hauteur, mais ne l'a pas éradiquée pour autant. L'écriture basée sur une organisation des hauteurs a perduré avec la musique spectrale, dans laquelle le timbre est majoritairement considéré comme une agrégation fusionnée de hauteurs. D'autre part, l'utilisation massive de la transformée de Fourrier, qui permet de décomposer le son en une somme de hauteurs élémentaires, a très certainement contribué au maintien du primat de la hauteur. Notre outil d'orchestration, en partant de l'analyse d'un son cible et en tentant de l'approcher à l'aide d'une mixture de sons monophoniques ne cautionne-t-il pas cette supériorité de la hauteur, et ne favorise-t-il pas une approche spectrale de l'orchestration ?

Au début de notre thèse, les techniques liées aux descripteurs spectro-harmoniques du signal étaient déjà suffisamment matures pour qu'un modèle spectral du timbre instrumental, à la fois descriptif et prédictif, puisse être rapidement développé. Nos premiers travaux [CTA+06] ont ainsi concerné l'imitation d'un instrument de l'orchestre par un petit groupe de deux ou trois instruments. L'analyse spectrale et la décomposition harmonique de la cible à travers les MRP (voir paragraphe 8.1.2 et annexe A) procédaient de cette démarche initiale. Dans un second temps seulement fut-il question de problèmes plus généraux, que nous avons délibérément choisi d'aborder avec la même description. Les composantes transitoires ou bruitées du timbre étant beaucoup plus délicates à modéliser, il était normal que la recherche sur l'aide à l'orchestration se concentre dans un premier temps sur des problèmes spectraux, sans jamais cependant les traiter comme une finalité, mais comme un moyen à la fois d'éprouver notre approche combinatoire de l'orchestration et de valider nos stratégies d'optimisation basées sur une conception descriptive du timbre.

Cette vision de l'orchestration, dans laquelle le timbre comme objet d'analyse induit une organisation des hauteurs, s'apparente fort à la démarche du mouvement spectral. Bien que n'ayant jamais affiché la volonté de privilégier une esthétique musicale particulière, il est inévitable que dans les premiers temps de son émergence une technologie favorise certaines pratiques. Là où certains pourraient donc dénoncer une dérive ou une focalisation esthétique, nous préférons y voir un point d'entrée vers la complexité de l'orchestration. Notre recherche n'est en rien confinée dans le spectralisme, mais l'approche analytique de cette démarche l'a rendue plus perméable à une pensée scientifique qui s'aventurait en terre inconnue.

12.3.3 Liberté et contrôle

Aussi bien dans le cas d'une orchestration imitative que d'une orchestration générative, l'analyse d'un son est une étape incontournable de notre approche. A partir de cette analyse, notre système va chercher des combinaisons instrumentales dont le timbre résultant présente une certaine similarité avec le son initial. Les orchestrations proposées par l'outil demeurent donc dans un rapport indépassable de subordination au son premier. La « synthèse instrumentale » du son visé, toujours conditionnée par l'analyse du « timbre premier », ne pourra jamais prétendre à la même liberté expressive que la démarche sémantique de Varèse ou de Lachenmann (voir section 2.1), où la mise en temps et en espace du vocabulaire sonore n'est pas gouvernée par un système préexistant à l'œuvre. En revanche, elle sera plus aisément « contrôlable » par un outil scientifique orienté vers la création tel que celui que nous proposons.

12.3.4 Fusion instrumentale

Si les fonctions d'agrégation définies en annexe A permettent de prédire avec une précision acceptable (cf. chapitre 8) les valeurs de descripteurs d'une configuration, rien ne permet en revanche d'estimer la qualité de la fusion instrumentale des solutions proposées. En d'autres termes, rien ne permet de prédire si les orchestrations découvertes par notre système seront perçues comme un accord où comme un timbre unifié (sur ce point voir Erickson [Eri75]). Les études menées sur les phénomènes de fusion acoustique ont montré qu'ils étaient aussi délicats à décrire qu'à prévoir. Hormis quelques cas triviaux — on sait par exemple qu'un violon vibré et un accord de cuivres sont perçus comme deux plans sonores distincts — la fusion instrumentale dépend de nombreux paramètres, comme la synchronicité de l'entrée et de l'évolution des voix, les rapports de hauteurs et d'intensité, les instruments utilisés et leur disposition spatiale... A l'heure actuelle, nous n'avons pas d'autre choix que de laisser au compositeur le soin de contrôler et d'estimer par lui-même le niveau de fusion instrumentale. Après tout, on trouve déjà chez Wagner, Debussy, Schoenberg, Webern et Stravinsky des agrégats de hauteurs que l'on perçoit davantage comme des timbres que des accords. Cent ans après, il est donc permis de supposer chez les compositeurs une bonne connaissance de ces mécanismes, grâce à laquelle il pourront adapter les orchestrations proposées par le système au degré de fusion instrumentale souhaitée.

12.3.5 Aspects temporels

Notre système ne pouvant aujourd'hui décrire et orchestrer que des sons stationnaires, l'évolution temporelle du timbre n'est pour l'instant possible que via un découpage de la cible en plusieurs segments statiques. Le compositeur est donc amené à définir autant de cibles et à rechercher autant de solutions que de segments. Non seulement cette approche ne permet d'appréhender que des tranformations relativement lentes du timbre, mais elle n'offre en outre aucun contrôle sur les transitions entre les segments. Les orchestrations étant recherchées de manière indépendante, la continuité timbrale et la jouabilité des transitions sont laissées à l'appréciation du compositeur. Nous verrons toutefois au chapitre 14 un exemple de timbre dynamique dont seul le début a été orchestré avec notre système, et dont la fin est déduite par modification continue de paramètres de jeu. Nous présenterons également deux exemples dans lesquels le solveur de contraintes présenté au chapitre 10 est utilisé pour transformer

graduellement une orchestration par l'ajout de contraintes supplémentaires, offrant ainsi la possibilité d'un contrôle temporel du timbre à l'aide de variables symboliques.

12.3.6 Etendue de la connaissance instrumentale

Au cours de nos multiples collaborations et rencontres avec les compositeurs, ces derniers n'ont pas manqué d'émettre un certain nombre de critiques sur notre outil. La principale concerne les limitations de l'instrumentarium actuellement disponible. Notre caractérisation exclusivement spectro-harmonique du timbre restreint en effet la connaissance instrumentale à des échantillons d'instruments à sons harmoniques et entretenus. Les sons de percussions, de piano, de harpe, de guitare ou encore de pizzicati de cordes n'y figurent donc pas. Bien qu'ils possèdent tous, à l'exception d'une certaine classe de percussions, des composantes harmoniques et une hauteur unique clairement identifiable, leur intégration dans notre système nécessite une compréhension de la façon dont ils se mélangent aux son entretenus, ce qui dépasse aujourd'hui le spectre de nos connaissances.

Au caractère entretenu des échantillons instrumentaux s'ajoute également une condition d'harmonicité. Notre approche de l'orchestration consiste en effet à « expliquer » un spectre complexe par un ensemble de spectres harmoniques. Au grand dam de nombreux compositeurs, les instruments à spectre complexe (tels que certaines percussions ou les vents en « multiphonique ») ne peuvent donc intervenir dans nos propositions d'orchestration. Leur incorporation à l'instrumentarium nécessiterait de « repenser » notre critère harmonique : comment en effet qualifier la contribution de ce type de sons aux principaux partiels résolus de la cible en dehors du prisme monophonique ? C'est une question qui reste pour l'instant ouverte.

Nous ne devons cependant pas oublier que la connaissance instrumentale de notre système est encore bien loin de couvrir l'ensemble des sons harmoniques et entretenus. Tuba, accordéon, saxophones, contrebasson, clarinette basse, trombone basse, flûte picolo, flûte basse produisent des timbres accessibles à nos descripteurs et leur utilisation est fréquente dans les grands orchestres comme dans les ensembles. La généralisation du notre outil d'orchestration pourrait donc déjà commencer par l'ajout de ces instruments, ainsi qu'à la possibilité pour l'instrumentarium complet de jouer en quarts de tons.

La possibilité pour les compositeurs de compléter la connaissance instrumentale à l'aide de leurs propres banques de sons est également apparue, au cours des divers échanges et collaborations avec ces derniers, comme un enjeu crucial. Nombreux sont en effet les compositeurs qui travaillent à partir d'esquisses enregistrées par leurs futurs interprètes et qui préfèrent utiliser ces sons plutôt que ceux du système. Le cas d'un compositeur/interprète ayant une connaissance plus approfondie que le système sur la pratique de son instrument peut également se présenter. Enfin, l'apport de sons électroniques doit également permettre d'utiliser l'outil pour des pièces mixtes.

Toutes ces raisons justifient la possibilité de pouvoir « personnaliser » la connaissance instrumentale de l'outil d'aide à l'orchestration, via une interface dévolue à la création de nouveaux instruments, couplée à un outil automatique d'extraction de descripteurs. Cette fonctionnalité requiert toutefois que les sons importés par l'utilisateur soient cohérents en termes de dynamiques avec les sons « natifs » du système, sous peine d'obtenir des orchestrations aberrantes du point de vue des équilibres acoustiques entre les différentes sources. Là encore, il n'existe pas de réponse toute faite. On peut éventuellement envisager une procédure bien calibrée pour ajuster le gain des échantillons en fonction d'une dynamique théorique don-

née par l'utilisateur. Une telle solution n'a rien d'évident à mettre en œuvre, car l'intensité perceptive est une donnée éminemment subjective. En outre, elle ne résout pas le problème des sons électroniques, pour lesquels la notion de dynamique — au sens d'une intention de compositeur laissée à l'appréciation des interprètes — n'existe pas vraiment.

Chapitre 13

Scénario d'utilisation

Un exemple d'orchestration imitative
Relance de l'algorithme à partir d'une solution intermédiaire
Contrôle de l'évolution temporelle à l'aide de contraintes

Ce chapitre est un exemple d'orchestration imitative dont le processus complet, du son cible initial à la partition définitive, est détaillé étape par étape. La notion d'interface avec l'utilisateur est ici centrale. Les concepts utilisés ont tous été introduits dans les chapitres précédents, à l'exception toutefois de la notion de « clustering par schémas » (voir section 13.2), qui permet de regrouper les solutions efficientes en catégories distinguées selon des paramètres symboliques. Le lecteur est invité à avoir en tête les schémas des figures 12.1 et 12.2.

13.1 Données d'entrée

Les éléments permettant de contrôler le système en entrée sont reportés figure 13.1. La fenêtre de gauche permet de définir l'effectif orchestral. Au centre, un patch MAX/MSP contrôle l'analyse du son à orchestrer ainsi que le « mapping de fondamentales » (voir infra). A droite, les partiels de la cible sont « projetés » sur un ensemble de spectres harmoniques.

Le patch central se décompose lui-même en deux blocs principaux. Dans la partie supérieure, une visualisation en forme d'onde permet de sélectionner une partie « stable » du timbre à reproduire. Un certain nombre de paramètres contrôlent également l'extraction des principaux partiels résolus (MRP — voir paragraphe 8.1.2 et annexe A) par le programme *pm2*. La partie inférieure est utilisée pour « relier » les MRP à un ensemble réduit de hauteurs fondamentales sur ces partiels. Cet ensemble de hauteurs agit comme un filtre sur la base de données constituant la connaissance instrumentale : les sons dont la hauteur n'appartient pas à l'ensemble ne peuvent pas figurer dans les propositions d'orchestration. L'hypothèse sous-jacente est qu'ils ne possèdent pas de partiels communs avec les MRP de la cible.

Dans le cas présent, la cible est un son de trombone joué avec une anche de basson (ce problème nous a été soumis par le compositeur Yan Maresz). Nous avons ici choisi d'extraire 25 partiels principaux avec une résolution fréquentielle ($f0_{min}$) de 50 Hz. Selon le type de cible, les valeurs de ces paramètres peuvent être capitales pour la pertinence de l'analyse. Vient alors ensuite une autre étape cruciale pour la réussite de l'orchestration : la détermination des hauteurs parmi lesquelles l'algorithme de recherche pourra choisir les sons candidats. Plusieurs possibilités s'offrent à l'utilisateur :

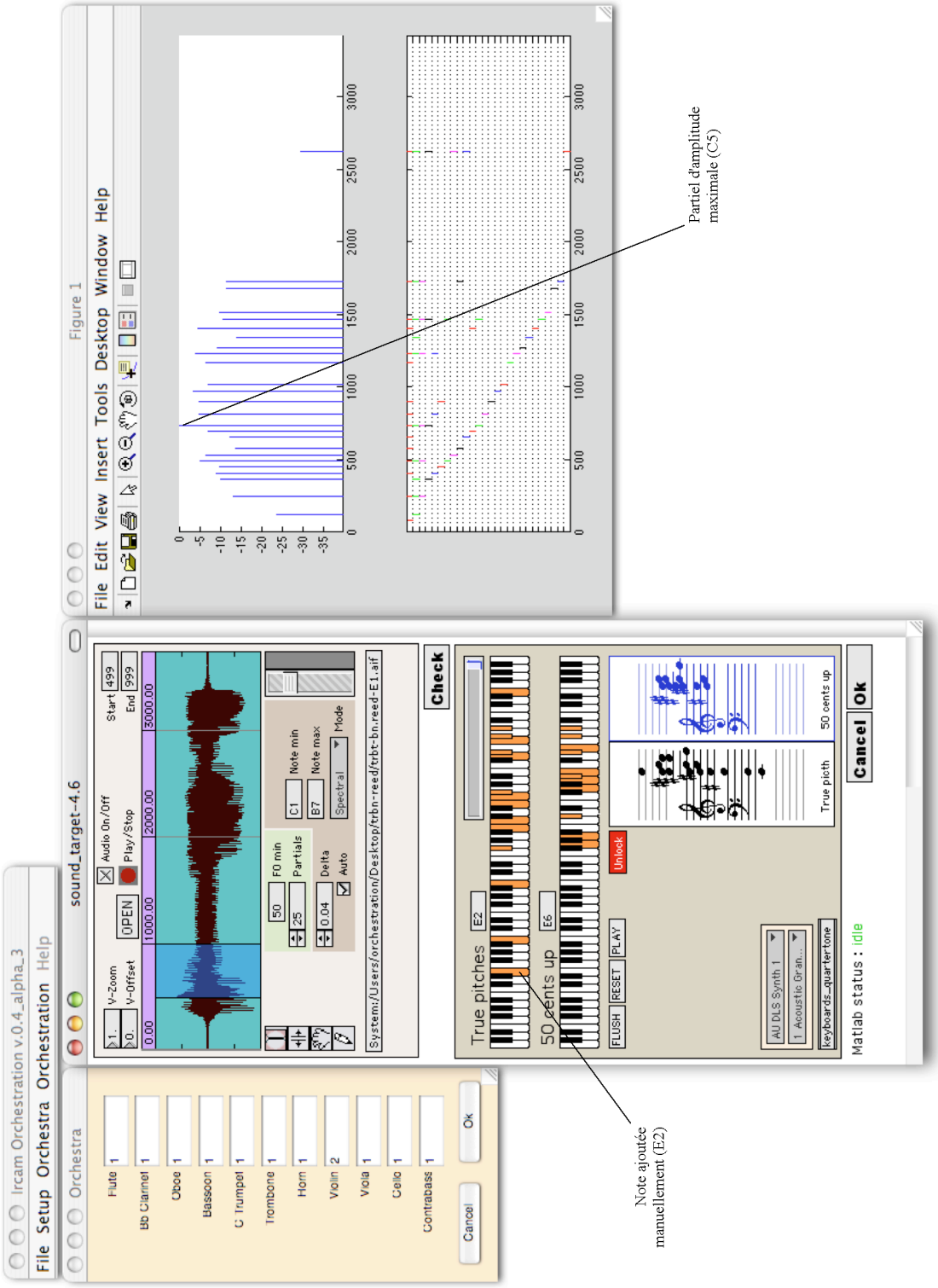

FIG. 13.1 – Choix d'un orchestre et interface d'analyse de la cible

1. Définir manuellement les hauteurs possibles. Un système de claviers (le second est un quart de ton plus haut que le premier) est prévu à cet effet dans la partie inférieure du patch MAX/MSP.
2. Déduire automatiquement des fréquences de partiels les hauteurs fondamentales susceptibles de les expliquer. Cette méthode à été présentée dans [CTA+06] et donne de bons résultats lorsque le son cible est harmonique ou une mixture de sons harmoniques. Elle prend toutefois en argument des paramètres dépendant du problème et en général délicats à fixer.
3. Convertir les fréquences des partiels en hauteurs de note par approximation au quart de ton le plus proche : c'est la technique habituelle des compositeurs « spectraux », et la méthode par défaut dans notre système.

Dans les deux derniers cas, l'utilisateur a évidemment la possibilité d'éditer manuellement les hauteurs suggérées par le système. C'est d'ailleurs ce que nous avons fait en rajoutant la note E2 (voir figure 13.1), très présente à l'écoute, mais dont la fréquence n'appartient pas aux principaux partiels résolus (MRP).

Suite à cette étape, seuls les sons dont la hauteur appartient à l'ensemble précédemment construit pourront figurer dans les propositions d'orchestration. Avant de lancer la recherche, la connaissance instrumentale est donc filtrée selon l'attribut « hauteur », et la contribution aux MRP de chacun des candidats est calculée selon la méthode exposée en annexe A. Les contributions sont calculées par groupes de même hauteur : ce que nous avons appelé en infra le « mapping des fondamentales » consiste à projeter chaque hauteur précédemment sélectionnée sur les MRP de la cible. En pratique, nous associons à chaque hauteur un peigne harmonique théorique (sans inharmonicité[1]) et nous identifions les harmoniques dont la fréquence coïncide (à une erreur relative près) avec un partiel de la cible. La fenêtre de droite de la figure 13.1 illustre cette mise en correspondance. Tous les sons de hauteur Mi2 (première série rouge) contribuent aux MRP numéros 2, 4, 6, 8, 9, 11, 12, 13, 16, 17, 20, 21, 24, 25 de la cible. Tous les sons de hauteur Si2 (première série verte) contribuent aux MRP numéros 1, 2, 3, 6, 11, 17, 19, 21, 24, 25 de la cible, etc. Un son de hauteur Sol#6 haut ne contribue qu'au 24^{e} MRP, un son de hauteur Mi7 au 25^{e}. Il y a contribution si l'erreur relative entre la fréquence théorique de l'harmonique et la fréquence d'un MRP est inférieure à un certain seuil, modifiable par l'utilisateur.

A la fin de ce processus, qui pour certaines cibles peut être assez long, les descripteurs de la cible ont été extraits, la banque de sons filtrée selon les hauteurs et les contributions aux MRP de la cible calculées pour chaque son candidat. La recherche d'orchestrations peut alors commencer. Cette étape, qui constitue l'essentiel de notre travail, n'est pas décrite ici. Nous renvoyons pour cela le lecteur au chapitre 9.

13.2 Visualisation des solutions

Lorsque l'algorithme de recherche se termine, les configurations efficientes sont retournées à l'utilisateur et peuvent être visualisées selon plusieurs points du vue :

[1]Négliger l'inharmonicité, c'est-à-dire le fait que les fréquences des partiels ne soient pas exactement des multiples de la fréquence fondamentale, n'est pas aberrant pour des instruments à son entretenu (i.e. fonctionnant en régime forcé). On sait en effet que les phénomènes de déviation harmonique sont beaucoup plus marqués dans les instruments à régime libre, comme les percussions ou les cordes frappées ou pincées.

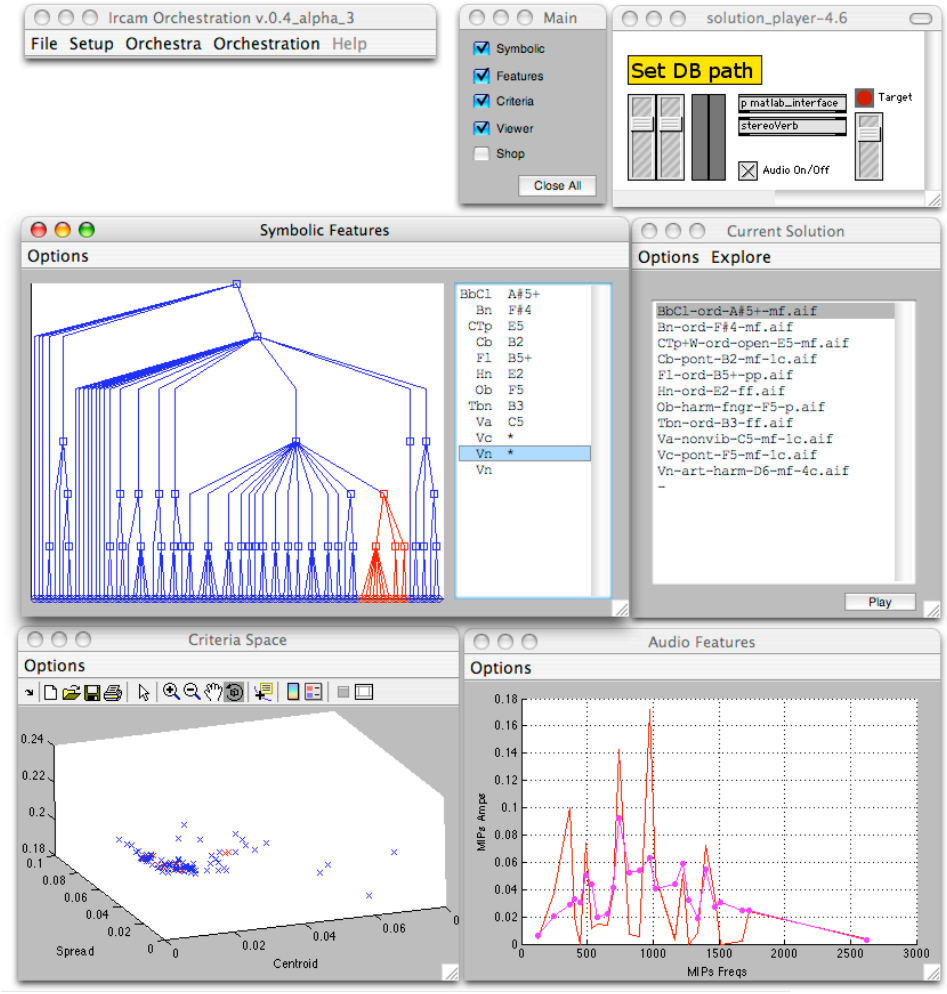

FIG. 13.2 – Navigation dans l'ensemble des solutions

1. Description textuelle de la solution couramment explorée : située en haut à droite de la figure 13.2, une fenêtre donne un accès aux composantes de la solution courante. L'orchestration peut être simulée globalement à travers un *sampler* développé dans MAX/MSP (voir section 12.2), et chaque composante peut être entendue séparément.

2. Navigation dans l'espace des critères : la fenêtre en bas à gauche sur la figure 13.2 permet d'explorer l'approximation du front de Pareto dans l'espace des critères. Le détail de la solution sélectionnée est envoyé dans la fenêtre de solution courante et l'orchestration correspondante est simulée dans le *sampler*. L'utilisateur peut choisir de « balayer » les solutions individuellement ou par groupe ; dans ce dernier cas, un algorithme de clustering hiérarchique permet de diviser l'approximation en un nombre de sous-ensembles défini par l'utilisateur. Un représentant est identifié pour chaque cluster et envoyé dans la fenêtre de solution courante lorsqu'un des élements du cluster est sélectionné.

3. Navigation par valeurs de descripteurs : Dans cette fenêtre (en bas à droite sur la figure 13.2), l'utilisateur a accès aux descripteurs des solutions. L'approximation est projetée dans un espace bi-dimensionnel, permettant la navigation selon deux descripteurs scalaires (moments spectraux) ou un descripteur vectoriel (MRP), comme c'est d'ailleurs le cas sur la figure 13.2.

4. Navigation par valeurs d'attributs : ce dernier mode de représentation (en haut à gauche sur la figure 13.2) offre la possibilité de parcourir l'ensemble des solutions selon un attribut particulier, c'est-à-dire une caractéristique symbolique en lien direct avec l'écriture musicale : hauteur, dynamique, mode de jeu, etc. L'approximation est représentée sous forme d'un arbre dont les solutions de chaque sous-arbre comportent des caractéristiques symboliques communes. Un sous-arbre peut donc être représenté par une « abstraction » ou, pour rester dans le jargon des algorithmes génétiques, un « schéma ». Le compositeur a ainsi une compréhension intuitive de la façon dont sont distribuées les différentes « écritures » proposées pour orchester une même cible. La construction de cet arbre fait appel à la notion de « clustering par schémas », que nous détaillons en infra dans cette section.

Les différents éléments de notre interface de navigation sont dépendants les uns des autres, de telle manière que la sélection d'une solution ou d'un groupe de solutions dans une fenêtre est automatiquement « propagée » aux autres vues. Sur la figure 13.2, le sous-arbre en rouge de l'espace des attributs a été sélectionné ; les solutions correspondantes apparaissent également en rouge dans l'espace des critères et dans l'espace des descripteurs[2]. Cette mise en relation des différents points de vue illustre les correspondances entre espace des décisions (monde du compositeur), espace des descripteurs (monde psychoacoustique) et espace des critères (monde de l'optimisation)[3]. D'un point de vue calculatoire, le passage du premier au dernier se fait par application successive des fonctions d'agrégation et fonctions de comparaison. L'interface de navigation permet de dépasser cette unilatéralité et d'observer la transmission, d'un espace à l'autre, des similitudes et des dissemblances.

[2]Seuls les MRP du sous-ensemble sont représentés dans ce mode de vue ; ceux de la cible y apparaissent en violet.

[3]Les caractéristiques de ces espaces ont fait l'objet de la section 7.6. Nous y avons notamment vu que seul le troisième pouvait supporter la notion d'optimalité, au prix d'une formalisation dont la pertinence a été prouvée au chapitre 8.

Clustering hiérarchique par schémas

La construction de l'arbre symbolique représenté sur la fenêtre supérieure gauche de la figure 13.2 fait appel à la notion de « clustering par schémas ». L'idée maîtresse est de construire un arbre hiérarchique dont chaque nœud incorpore une information sur les propriétés de son sous-arbre. Le clustering par schémas s'apparente à un clustering hiérarchique classique dans lequel un nouvel élément est créé à chaque regroupement, représentant les propriétés communes du groupe.

Nous avons vu au chapitre 9 que l'espace des décisions pouvait être défini comme un produit de domaines finis, chaque domaine étant associé à un instrument de l'orchestre. Chaque solution du probème d'orchestration est donc représentée par un N-uplet de $D = \bar{D}_1 \times ... \times \bar{D}_N$ (voir section 9.2). Nous commençons par étendre l'espace de recherche à $D^* = \bar{D}_1^* \times ... \times \bar{D}_N^*$, avec $\bar{D}_n^* = \bar{D}_n \cup \{*\}$, où $*$ représente n'importe quel élément de $\bar{D}_n$. Un N-upluet de D^* comporte donc des éléments déterminés (appartenant aux domaines $\bar{D}_n$) et des éléments non déterminés, marqués par le symbole $*$. Il représente ainsi un *ensemble* de solutions qui peut être obtenu en instanciant les coordonnées marquées par le symbole $*$. Nous donnons à cette abstraction le nom de « schéma », par analogie avec les schémas génétiques. Plus le nombre de symboles $*$ est élevé, plus le schéma correspond à un niveau d'abstraction élevé. Nous munissons alors D^* d'une distance de hamming d^*, définie comme le nombre de coordonnées différentes entre deux N-uplets, aux symboles $*$ près.

Les éléments initiaux du clustering hiérarchique sont soit les configurations efficientes, soit des abstractions de ces configurations selon un attribut particulier. Sur la figure 13.2 les configurations ont par exemple été remplacées par des abstractions selon la hauteur de note : au lieu de désigner une configuration par la liste précise des attributs de chacune de ses composantes, on s'en tient à la seule information de hauteur pour chaque instrument, les autres paramètres de l'écriture n'étant pas considérés. Les coordonnées des configurations sont dans ce cas des hauteurs (il existe donc plusieurs réalisations possibles d'une même configuration, définie au mode de jeu, à la dynamique, à la sourdine et à la corde près), et le symbole $*$ désigne n'importe quelle hauteur jouable par l'instrument en question.

Le processus du clustering hiérarchique commence par calculer les distances d^* pour chaque paire de configurations. Les configurations les plus proches sont regroupées au sein d'un même cluster et leurs coordonnées communes permettent de construire un schéma réprésentatif du groupe (les coordonnées dont les valeurs changent au sein du groupe sont remplacées par le symbole $*$). Les nouveaux schémas sont ensuite ajoutés aux configurations qui n'ont pas encore été regroupées et le processus se poursuit itérativement jusqu'à l'obtention d'un schéma unique. Plus on avance dans la construction de l'arbre, plus le niveau d'abstraction des schémas augmente. Lorsque deux schémas identiques naissent de sous-groupes différents, ceux-ci sont fusionnés en un groupe unique.

La fenêtre en haut à gauche sur la figure 13.2 représente le résultat d'un clustering par schémas sur les hauteurs de note. Les solutions sont représentées en bas de l'arbre par des cercles. Les schémas apparaissent sous formes de petits carrés, et leur hauteur dans l'arbre correspond à leur niveau d'abstraction : le schéma sélectionné (sous-arbre rouge) est situé deux niveaux au dessus des solutions et comporte deux fois le symbole $*$. Toutes les configurations de ce cluster ont donc dix caractéristiques communes : la clarinette joue toujours un La5# haut, le basson un Fa#4, la trompette un Mi5, etc. Le second violon ne joue jamais. Ce qui différencie les solutions au sein du groupe sont les hauteurs jouées par le violoncelle et le

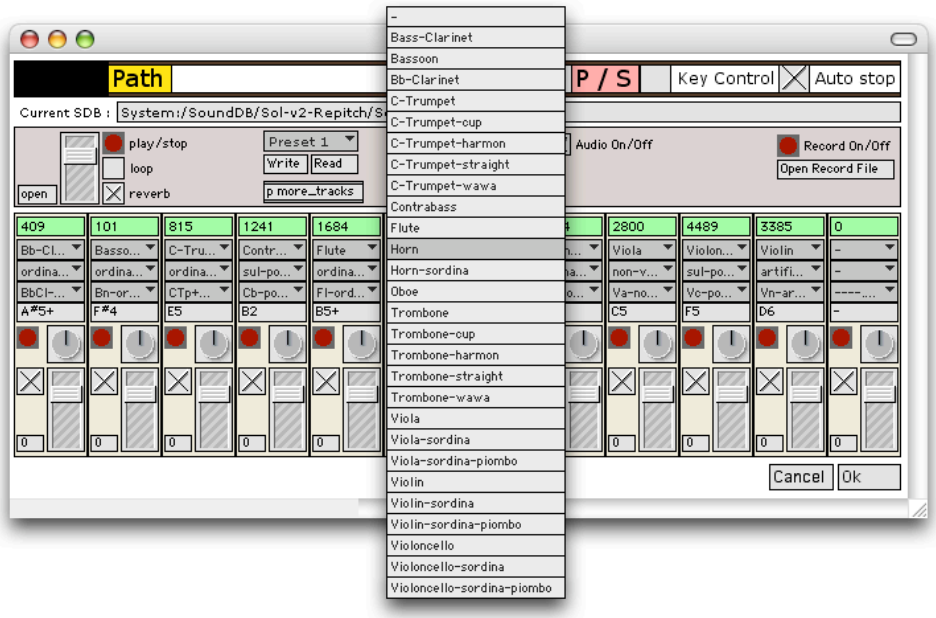

FIG. 13.3 – Edition manuelle d'une solution

premier violon. Le schéma associé au nœud courant est représenté dans la partie droite de la fenêtre. Chaque cluster est représenté par la solution qui minimise la somme des distances aux autres membres du groupe dans l'espace des critères normalisés. C'est ce représentant qui est envoyé dans la fenêtre de solution courante et joué dans le *sampler*.

Le clustering hiérarchique par schémas permet au compositeur d'identifier des grandes « catégories d'écriture » pour la cible à orchestrer. Deux éléments proches dans l'arbre symbolique correspondent à deux orchestrations similaires, deux éléments lontains présentent des cararctéristiques symboliques différentes. Par défaut, l'attribut utilisé pour la catégorisation est la hauteur, mais on peut très bien construire l'arbre à l'aide d'une distance sur le mode de jeu, la dynamique, ou encore, si cela a un sens, la sourdine.

La notion de schéma symbolique est en lien direct avec celle de schéma génétique (voir section 9.2) : de façon évidente, deux éléments d'un même cluster engendrent par crossover (échange aléatoire des coordonnées) deux rejetons qui partagent toujours le même schéma symbolique. Les clusters symboliques sont donc stables par application du crossover. Cette remarque nous permet peut-être de concevoir une application moins aveugle du crossover au cours de la recherche génétique. Si nous voulons intensifier la recherche dans une zone particuluère de l'espace de recherche, on peut alors restreindre le crossover aux éléments d'un même cluster de faible niveau d'abstraction. A l'inverse, appliquer le crossover à des éléments très éloignés dans l'arbre symbolique encourage l'exploration de l'espace de recherche. A méditer pour une recherche future...

13.3 Edition, affinage, transformation

Lors de l'exploration des configurations efficientes, une option permet d'archiver une solution courante jugée intéressante par le compositeur dans un « magasin » au sein duquel plusieurs opérations d'édition et de transformation sont possibles.

La commande `edit` ouvre la solution courante dans une interface d'édition développée en MAX/MSP et représentée sur la figure 13.3. Cet environnement se présente comme une table de mixage dont chaque piste est attribuée à un instrument. L'utilisateur a ainsi la possibilité de modifier manuellement les échantillons de l'orchestration suggérée par le système, ainsi que d'ajuster les niveaux et les panoramiques (position dans l'espace stéréophonique) de chaque piste, afin de parfaire le réalisme de la simulation. En ce qui nous concerne nous avons simplement modifié la dynamique de la clarinette par rapport à la solution de la figure 13.2 : en diminuant simplement la dynamique de *mf* à *pp*, nous améliorons le niveau de fusion dans la mixture.

Lorsque le compositeur est satisfait avec une proposition d'orchestration, une commande `redraw` permet de relancer l'algorithme de recherche afin d'améliorer cette dernière. Les préférences d'écoute (i.e. les poids relatifs associés aux critères) sont déduites des valeurs de critères de la solution intermédiaire (voir paragraphes 7.2.4 et 9.3) par la méthode décrite en annexe B. Une population initiale est alors formée, constituée pour moitié de clones de la solution intermédiaire, pour moitié de configurations instanciées aléatoirement. L'algorithme 4 est relancé avec pour fonction de fitness la norme induite (voir section 8.5.1) par la solution intermédiaire. La recherche est donc favorisée vers la zone de l'espace des critères dominée par cette solution. Les poids relatifs des critères étant désormais fixés, le problème devient mono-critère.

Lorsque l'algorithme se termine, la population finale peut être totalement ordonnée à l'aide de la norme induite par la solution intermédiaire. Sous l'hypothèse que les préférences d'écoute du compositeur peuvent être déduites de cette solution, toutes les configurations de norme inférieure sont en théorie plus proches perceptivement de la cible. Dans le cas présent, nous choisissons de remplacer la solution intermédiaire par la meilleure solution obtenue après relance de l'algorithme, comme le montre la figure 13.4. La mention (D) indique que la solution domine la solution intermédiaire au sens de Pareto. Une comparaison avec la figure 13.2 fait apparaître un noyau invariant entre les deux orchestrations : la clarinette (dont la dynamique avait été manuellement modifiée), la flûte, le cor, le hautbois et le trombone n'ont pas été changés.

La solution ainsi obtenue va alors pouvoir être insérée dans un « mouvement orchestral » développé et contrôlé à l'aide de contraintes supplémentaires. L'algorithme *CDCSolver* présenté au chapitre 10 possède la particularité de ne modifier qu'une seule coordonnée à la fois au cours de la résolution. Il est donc possible, en partant d'une solution initiale et d'un réseau de contraintes, de transformer continument cette solution pour parvenir à un autre timbre. L'« enregistrement » de l'évolution se fait en conservant la trace de l'algorithme de résolution. Celle-ci se présente alors comme une suite de configurations dans laquelle un seul instrument est modifié à chaque étape. Rappelons que l'algorithme de résolution de contraintes utilisé dans ce cas — par opposition à la procédure de réparation des configuration initiales (voir chapitre 10) — incorpore des notions d'optimisation. Lors du mouvement vers une configuration voisine, le niveau de violation des contraintes ainsi que la fitness selon la fonction scalarisante en cours (voir section 5.7) sont simultanément considérés à travers la notion de dominance au sens de Deb (voir définition 6.1). L'évolution se fait donc de manière certes à améliorer

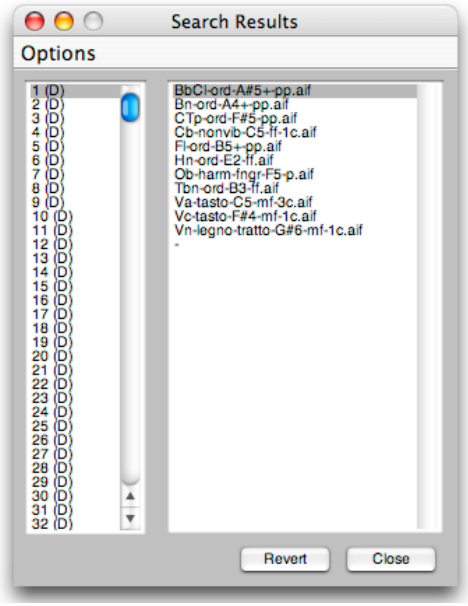

FIG. 13.4 – Intensification de la recherche à partir d'une solution intermédiaire

le niveau de consistance mais également à pénaliser le moins possible (voire à diminuer) la fonction de fitness, garantissant une certaine continuité de timbre entre deux configurations voisines.

La figure 13.5 illustre ce processus. Partant de la solution de la figure 13.4, nous avons imposé les contraintes suivantes :

1. Dans l'orchestration finale, tous les instruments jouent à l'unisson.
2. L'orchestration finale doit mobiliser au minimum trois instruments.
3. L'orchestration finale doit mobiliser au maximum trois instruments.
4. Dans l'orchestration finale, tous les instruments doivent jouer avec dynamique *pp*.

Ces quatre contraintes apparaissent dans la partie supérieure de la figure 13.5. Dans la partie inférieure, une résolution « tracée » en onze étapes aboutit à une solution consistante.

Une fonction d'export de partition via le format libre de notation musicale `abc` permet alors d'obtenir la partition du mouvement complet (voir figure 13.6). La progression a été ici transcrite dans l'ordre inverse de la résolution : elle commence avec trois instruments à l'unisson et finit avec l'imitation du son original. Chaque mesure représente une étape intermédiaire du calcul, et le passage d'une mesure à la suivante se fait par modification d'un seul élément. Remarquons que la note conservée pour l'orchestration initiale est un Do5, partiel d'amplitude maximale dans la cible (voir figure 13.1). Au cours de la résolution, le mouvement vers une configuration voisine est en effet gouverné par la notion de dominance au sens de Deb (voir définition 6.1) : l'algorithme cherche d'abord à satisfaire les contraintes, mais au prix d'une dégradation minimale de la fonction de fitness.

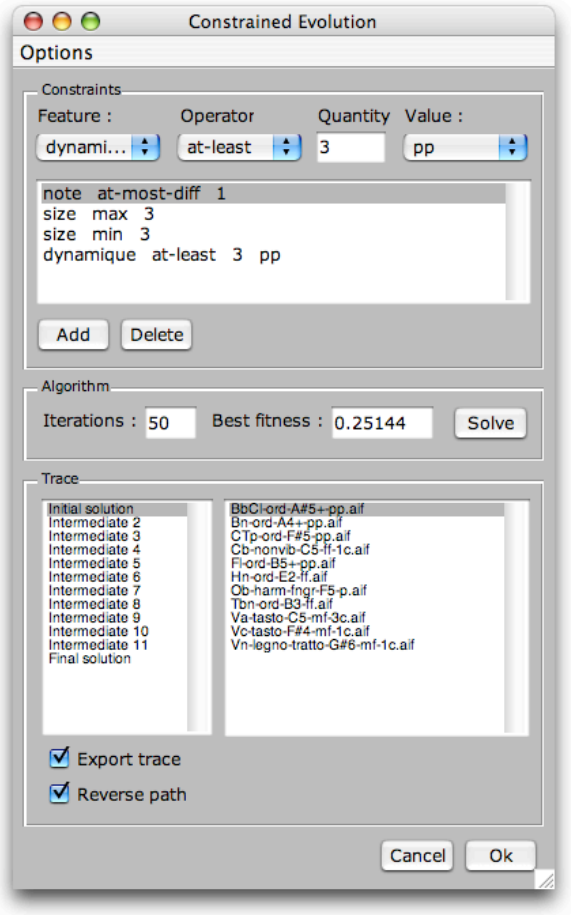

FIG. 13.5 – Transformation progressive du timbre à l'aide de contraintes

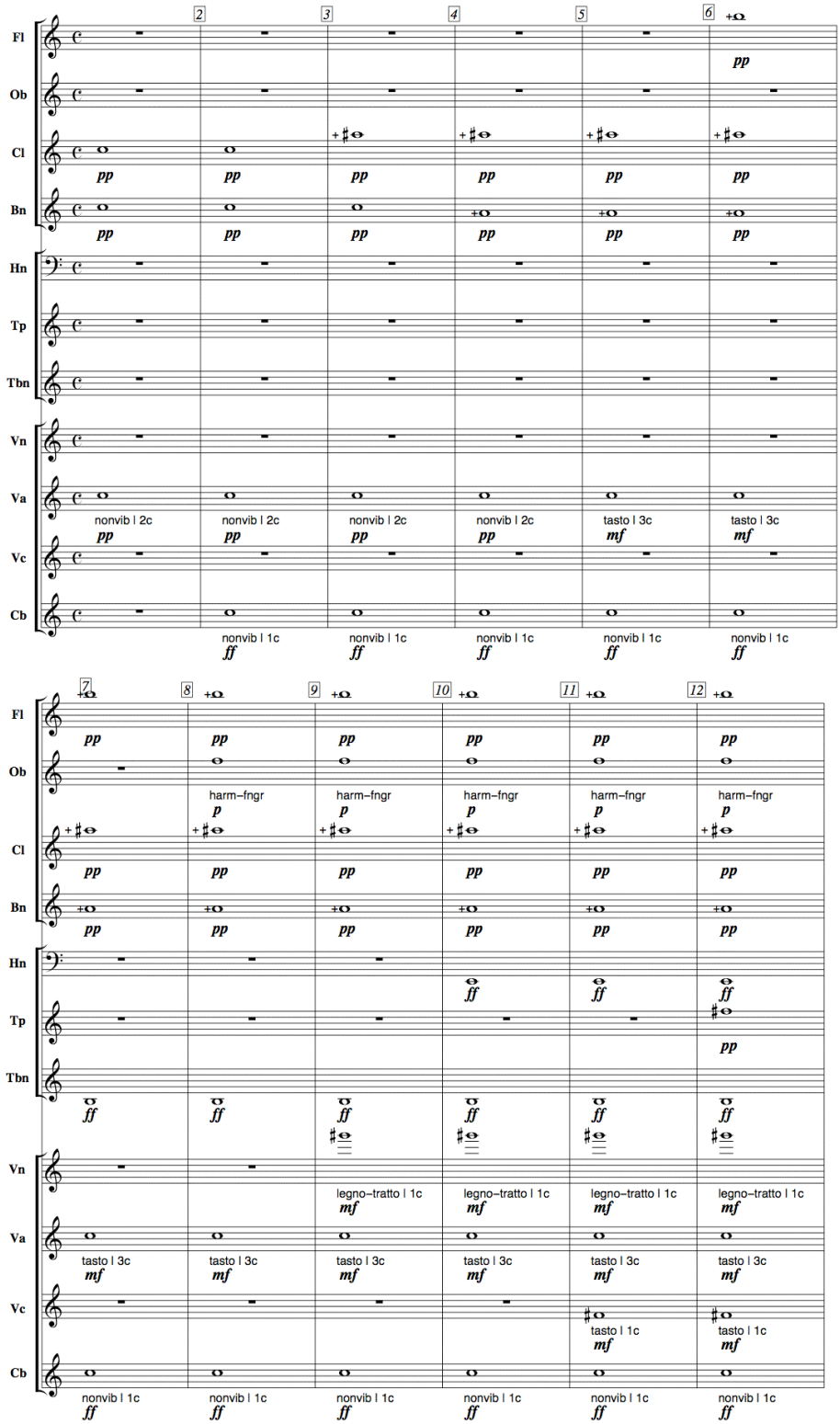

FIG. 13.6 – Partition finale d'une l'orchestration imitative et évolutive. A chaque mesure, seul un élément est modifié. Partition en Ut.

Chapitre 14

Exemples et productions

Exemples d'orchestrations (imitatives et génératives)
Contraintes et contrôle temporel du timbre
Ecriture pour bande et production sonore

Nous reportons dans ce chapitre des exemples probants d'orchestration obtenus à l'aide de notre système. Dans le premier, la cible est un son concret pré-enregistré ; dans le second l'environnement de CAO OpenMusic a été utilisé pour générer un son cible à partir d'un accord écrit. Le troisième est une requête du compositeur Oscar Bianchi. Il s'agit d'un timbre dynamique dont seule la première partie est orchestrée avec notre système, en spécifiant des contraintes qui permettent de « déduire » la seconde partie. Le quatrième exemple résulte d'une collaboration avec le compositeur Jonathan Harvey. La encore, des contraintes sont utilisées pour le contrôle temporel du timbre, mais l'ensemble du mouvement est gouverné par une fonction objectif globale traduisant une intention précise du compositeur. Enfin, le dernier exemple est tiré d'un travail récent avec le compositeur Gérard Buquet, dans lequel l'outil d'orchestration est utilisé comme un environnement de production sonore. Tous ces exemples peuvent être écoutés en ligne sur le site : `http://recherche.ircam.fr/equipes/repmus/carpentier/phdsounds/`.

14.1 Deux orchestrations de klaxon

La cible à orchestrer est ici un son de klaxon de voiture, enregistré dans la rue. Il s'agit d'un timbre polyphonique dans lequel deux hauteurs sont clairement identifiables, G#4 et C5 (C4 correspondant au Do central). L'espace de recherche a donc été limité aux sons de hauteurs G#4 et C5, plus leurs harmoniques respectives : G#5, C6, D#6, G6, etc.

La figure 14.1 reporte la transcription de deux orchestrations de ce même son de klaxon, la première à l'aide d'un petit ensemble de vents, la seconde avec un petit ensemble de cordes. La partition est en *hauteurs réelles*, y compris l'harmonique artificielle de contrebasse dont la notation ne correspond pas à l'usage mais est plus lisible dans une partition en Ut.

De façon surprenante, l'imitation semble moins convaincante avec les vents qu'avec les cordes. Cela est sans doute dû à une plus grande homogénéité des familles et des dynamiques dans le second cas, favorisant davantage la fusion des timbres. De plus, même si l'orchestration des vents est plus proche du son cible du point de vue de la brillance, le grain des cuivres dans le fortissimo engendre une rugosité qu'on ne retrouve pas dans le klaxon original.

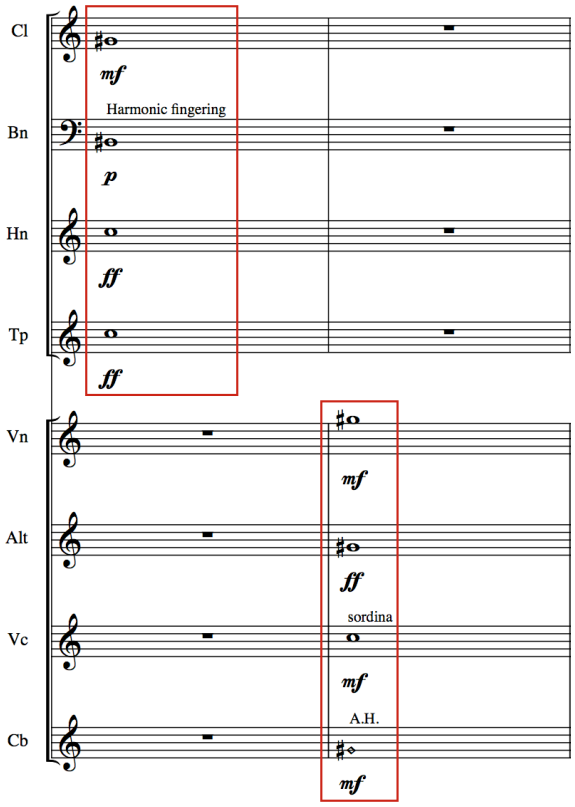

FIG. 14.1 – Deux orchestrations d'un même son de klaxon (en haut avec un ensemble de cuivres, en bas avec un ensemble de cordes). Partition un Ut.

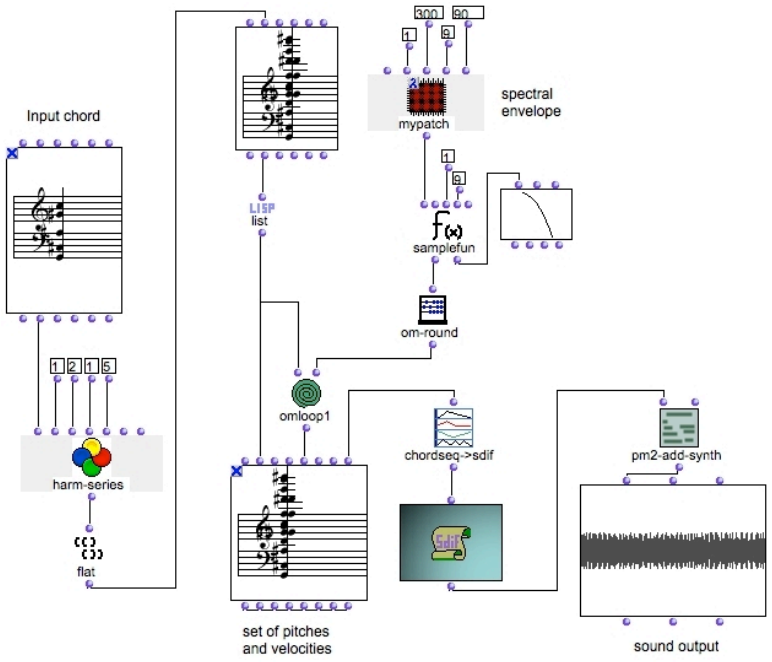

FIG. 14.2 – Un patch dans OpenMusic générant une cible à partir d'un accord

14.2 Une cible construite dans OPENMUSIC (Tristan Murail)

Cet exemple a été proposé par le compositeur Tristan Murail. A défaut d'un son enregistré (orchestration imitative), le point de départ est ici un accord écrit (G2, C#3, A#3, G#4, B4+[1]), à partir duquel il s'agit de générer un timbre. Le compositeur a ici eu recours au programme OPENMUSIC [Ago98] pour transformer l'accord initial en un son de synthèse. Ce processus illustre le concept d'orchestration générative introduit dans les paragraphes 7.1.1 et 12.1.

OPENMUSIC (OM) est un environnement de programmation visuelle pour la Composition assistée par ordinateur (CAO), développé à l'IRCAM par Gérard Assayag, Carlos Agon et Jean Bresson. Dans OM, il y a une parfaite équivalence entre un programme et son expression visuelle sous frome de « patch ». Un patch est une fenêtre comportant des boîtes connectées entre elles, représentant des objets musicaux, des fonctions,...ou des patches. Des connexions relient les entrées et sorties des fonctions, ainsi que les divers champs des objets. L'évaluation se fait « du haut vers le bas » : l'évaluation d'une fonction ou d'un objet déclenche le calcul de tous les éléments situés en amont.

La figure 14.2 est un patch OM prenant en entrée l'accord (G2, C#3, A#3, G#4, B4+). Celui-ci est tout d'abord complété par l'ajout d'harmoniques naturelles grâce à la fonction `harm-series`, puis les dynamiques sont ajustées grâce à un profil d'enveloppe spectrale. L'accord résultant est alors transformé en un ensemble de couples fréquence/amplitude engendrant un son cible via une méthode de synthèse additive.

L'orchestration du son synthétisé est reportée figure 14.3. Toutes les hauteurs de l'accord

[1]La signe « + » renvoie à la notation en quarts de tons : « B4+ » signifie donc « un quart de ton au dessus de Si4 », ou encore « Si4 haut ».

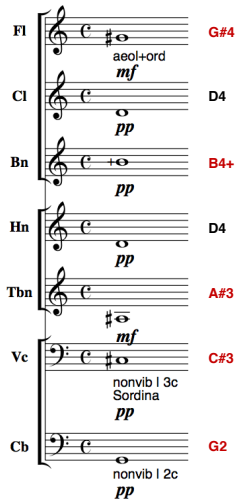

FIG. 14.3 – Orchestration d'un son de synthèse généré à partir d'un accord dans OpenMusic (les notes en rouge font partie de l'accord initial). Partition en Ut.

initial sont présentes, complétées par deux sons de hauteur Ré4, troisième harmonique de Sol2. L'orchestration résultante possède les mêmes caractéristiques harmoniques que le son de synthèse, auxquelles s'ajoutent la richesse des « petites variations » du timbre instrumental.

14.3 Une cible dynamique (Oscar Bianchi)

L'enthousiasme dont le compositeur Oscar Bianchi a fait preuve à l'égard de nos travaux l'a récemment amené à travailler avec nous en vue d'une future pièce pour orchestre. L'une des cibles alors données par le compositeur est particulièrement intéressante : elle illustre de quelle manière l'orchestration d'un timbre qui varie dans le temps peut, dans certains cas, être obtenue en travaillant dans un premier temps sur une partie statique du son, puis en « propageant » l'orchestration obtenue aux autres parties. De façon évidente, rien n'est ici possible sans le savoir et l'intuition du compositeur. Mais n'oublions pas que notre système n'est pas un « orchestrateur artificiel », mais un outil d'*aide* à l'orchestration : il ne donne pas de solutions toutes faites, mais suggère des pistes de travail.

Le son proposé dans cet exemple est une succession de deux phonèmes chantés : le premier est une voyelle « OU » dans le grave, fortement harmonique et relativement longue ; le second est une voyelle « AH » courte et majoritairement bruitée (elle comporte peu de composantes sinusoïdales). Ce son est représenté sous forme de sonagramme[2] sur la figure 14.4.

La première voyelle étant de plus grandes durée et harmonicité que la seconde, nous commençons par chercher une orchestration sur ce segment. Anticipant sur la « propagation » de ce timbre au phonème suivant, nous imposons, à l'aide du formalisme introduit au chapitre 10 les deux contraintes suivantes :

1. *L'orchestration doit contenir au moins un cuivre pédale de hauteur G#1 :* cette contrainte permet d'obtenir à la fois la hauteur fondamentale du son cible ansi que

[2]Le temps est en abscisse, les fréquences en ordonnées. La valeur de gris est proportionnelle à l'intensité du spectre.

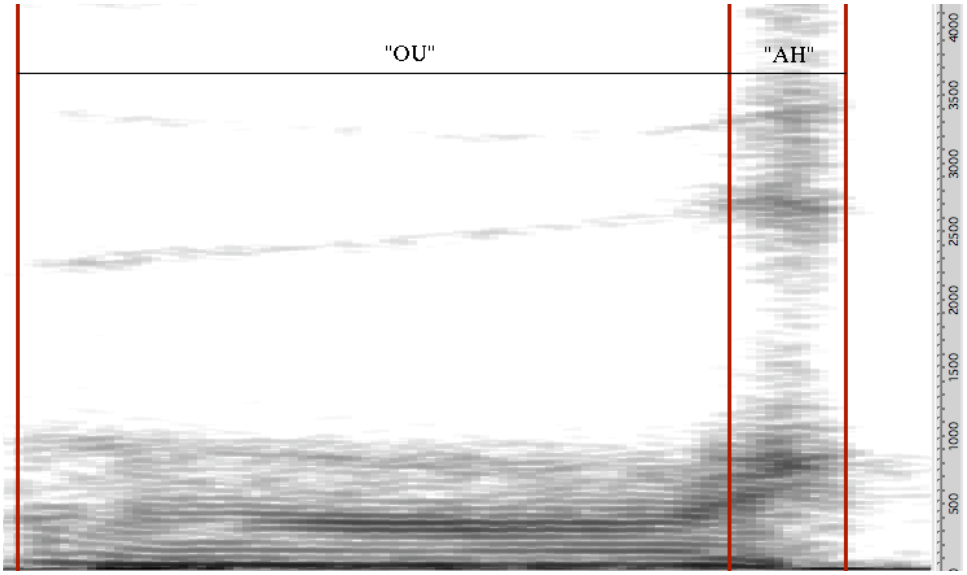

FIG. 14.4 – Une cible constituée de deux phonèmes chantés.

le caractère rugueux (pédale[3]) de la voix chantée à cette hauteur (très grave).

2. *L'orchestration doit contenir au moins trois cuivres avec une sourdine wah-wah en position fermée :* lorsqu'on passe rapidement de la position fermée à la position d'une wah-wah sur un cuivre, on obtient un son qui s'apparente à un « OU-AH » (d'où le nom de la sourdine). En faisant jouer la syllable « OU » avec une sourdine en position fermée, on prépare ainsi la transition vers la syllabe « AH », qui pourra être obtenue, en autres, à l'aide d'une ouverture rapide.

Sur la partie gauche de la figure 14.5 a été reproduite une proposition retournée par le système pour la première syllabe. Le signe « + » sur les cuivres indique la position fermée de la sourdine. La figure de droite illustre le mouvement de « OU » vers « AH », obtenu à l'aide de crescendos et d'une ouverture brusque des sourdines (signe « o »). Notons en outre que certaines dynamiques ont été ajustées manuellement.

14.4 *Speakings* (Jonathan Harvey)

Cet exemple résulte d'une collaboration avec le compositeur britanique Jonathan Harvey pour sa pièce pour orchestre et électronique *Speakings*, commande commune de l'IRCAM, de la BBC et de Radio-France. Comme le suggère le titre de la pièce, le compositeur s'est attaché à l'idée d'un orchestre imitant les timbres de la voix humaine. Nous avons rencontré le compositeur dans un studio de l'IRCAM après qu'il a eu vent du projet sur l'aide à l'orchestration et manifesté son désir d'utiliser cette technologie pour l'écriture de *Speakings*.

Au début de la troisième partie de cette pièce, une partie de l'orchestre exécute un ostinato (i.e. un motif longuement répété) qui sert de « trame harmonique » sur laquelle les autres instruments, notamment le trombone, évoluent en solistes. Cette « texture orchestrale » a

[3]Les notes pédales sur les cuivres correspondent au registre le plus grave et se caractérisent par la présence de battements (forte modulation d'amplitude).

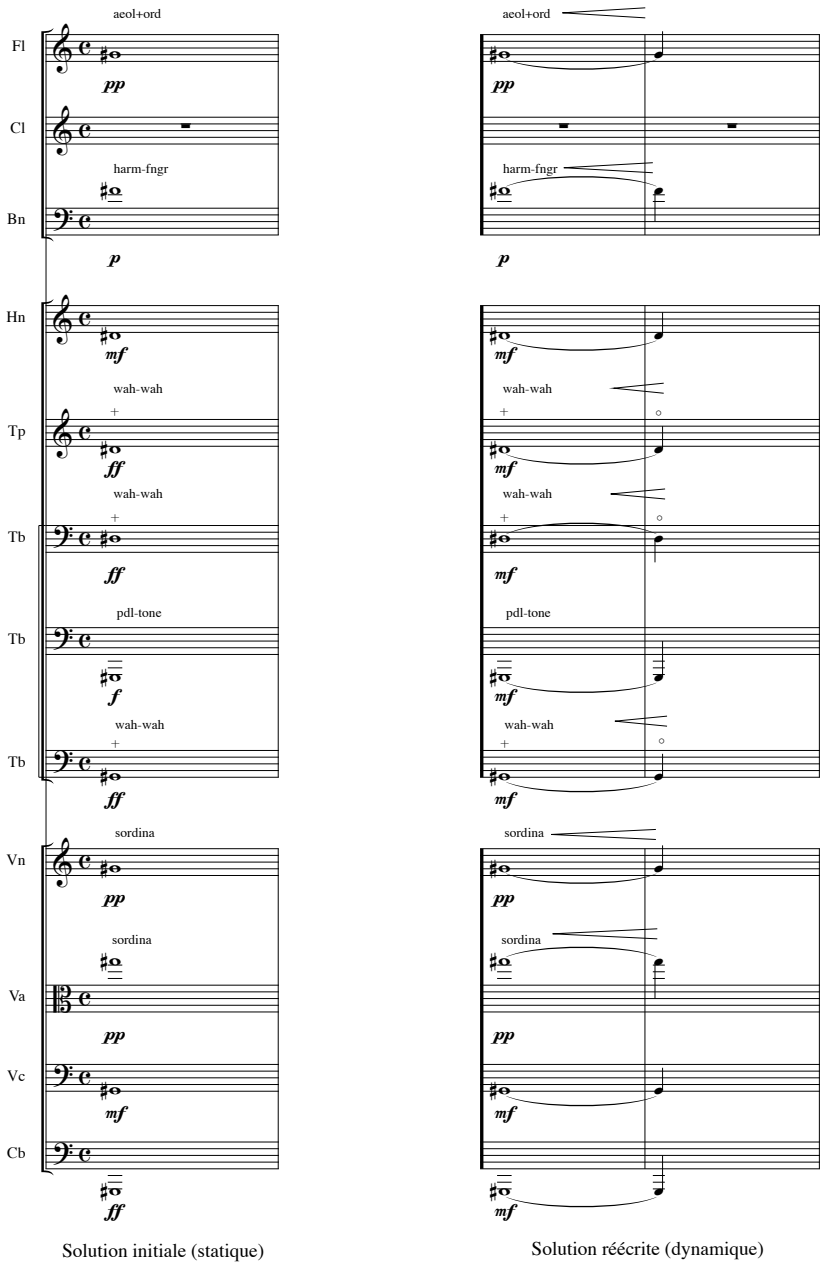

FIG. 14.5 – Orchestration d'un timbre dynamique par « propagation » d'un timbre statique. Partition en Ut.

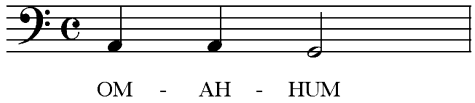

FIG. 14.6 – Mantra chanté par le compositeur Jonathan Harvey

été entièrement écrite à l'aide de l'outil d'orchestration à partir d'un mantra chanté par le compositeur (voir figure 14.6).

Jonathan Harvey souhaitait que le motif de la figure 14.6 soit répété 22 fois, non pas à l'identique, mais en respectant les « directions d'évolution » suivantes :

1. La dynamique de l'ostinato doit aller crescendo au cours des 22 répétitions.
2. Le son de l'orchestre doit devenir de plus en plus brillant, et se rapprocher de plus en plus du timbre des voyelles chantées.
3. Les orchestrations doivent utiliser des hauteurs de plus en plus élevées (harmoniques des notes fondamentales chantées), et composer des accords de densité harmonique (nombre de hauteurs différentes) croissante.

L'outil d'orchestration n'étant pas directement utilisable pour traiter ce problème, nous avons donc procédé de la manière suivante :

1. Le « sous-orchestre » mobilisé pour jouer l'ostinato étant composé de 13 musiciens, au plus 13 notes sont jouables simultanément. Pour chaque voyelle chantée, nous avons donc généré 13 ensembles de 10 solutions chacun, avec pour le k-ième ensemble, la contrainte d'avoir au moins k hauteurs différentes dans les propositions d'orchestration.
2. Pour chaque voyelle chantée, nous avons donc obtenu 130 solutions de timbres variées, pour lesquelles nous avons calculé plusieurs descripteurs : Intensité perceptive, centroïde spectral, MRP (voir section 8.1.2 et annexe A), nombre de sons utilisés, densité harmonique, hauteur maximale.
3. Pour chaque voyelle, nous avons enfin utilisé un algorithme de recherche locale pour trouver un « chemin » reliant 22 solutions parmi 130, de telle manière que les valeurs des descripteurs ci-dessus soient des fonctions croissantes du temps.

Un extrait de la solution totale calculée par cette procédure est reproduit sur la figure 14.7, et la partition finale de ce passage est donnée figure 14.8 (les parties écrites à l'aide du système d'orchestration sont surlignées en rose). Dans l'ensemble, l'orchestration suggérée par le système à été conservée par le compositeur, à quelques modifications près. Ces dernières concernent essentiellement la conduite des voix, c'est-à-dire la cohérence interne de chaque partie, dans son développement horizontal, « mélodique ». Notre outil d'orchestration n'a en effet pour l'instant qu'une vision verticale, « harmonique », de la texture. En outre, l'absence de contraintes d'exécution dans notre modélisation engendre immanquablement des enchaînements irréalisables. Par exemple, le trompettiste qui veut scrupuleusement suivre la partition de la figure 14.7 doit changer de sourdine au milieu de la 13^{e} mesure, l'enlever sur le premier temps de la 14^{e}, puis en changer encore deux fois sur les deux temps suivants. C'est évidemment parfaitement impossible, et le compositeur a choisi de garder la même sourdine pour le passage entier.

Une autre différence remarquable est la disparition aux cordes des « legno-tratto », mode de jeu contemporain qui consiste à jouer avec le bois de l'archet au lieu de la mèche. Le son

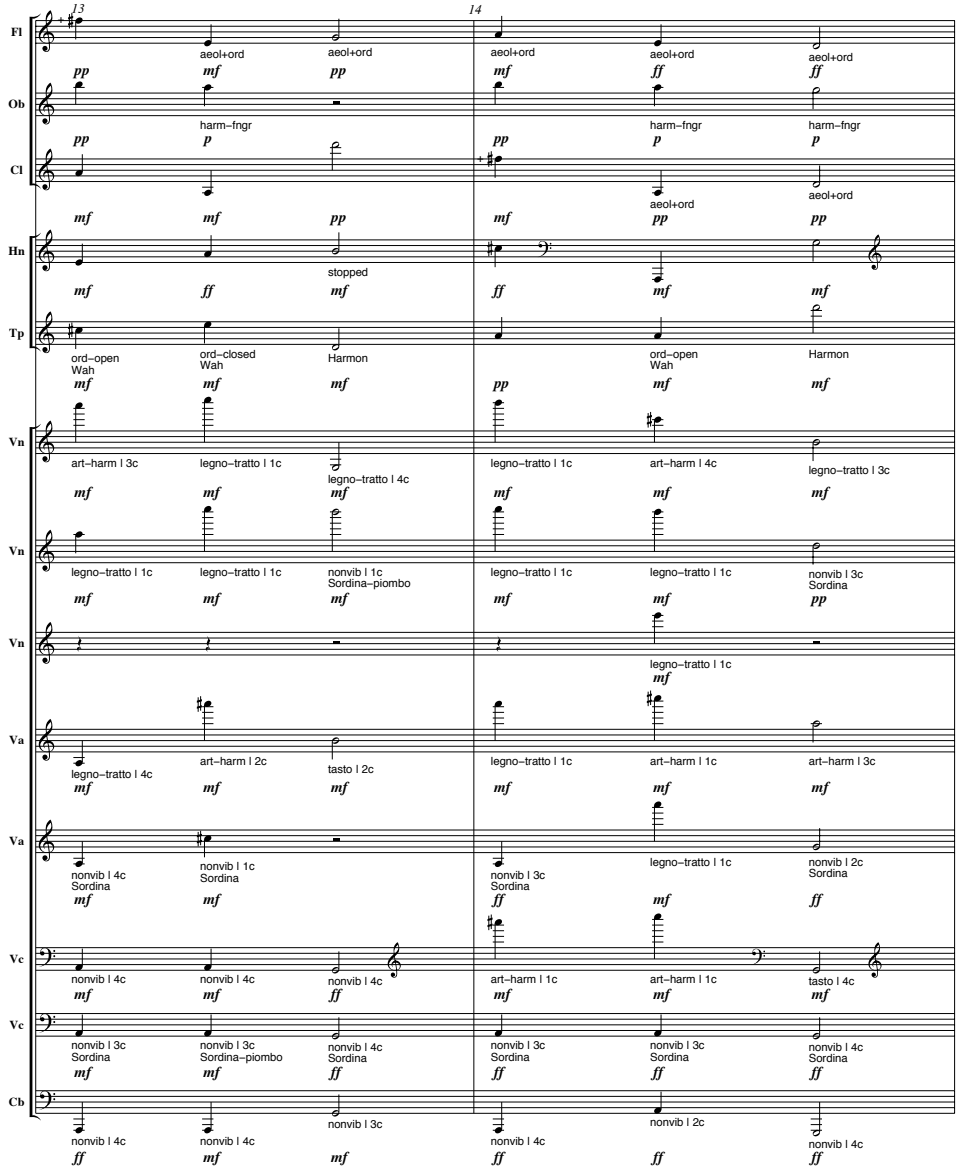

FIG. 14.7 – Mesures 11 et 12 d'une orchestration de mantra (ostinato) pour la pièce *Speakings* de Jonathan Harvey. Partition en Ut.

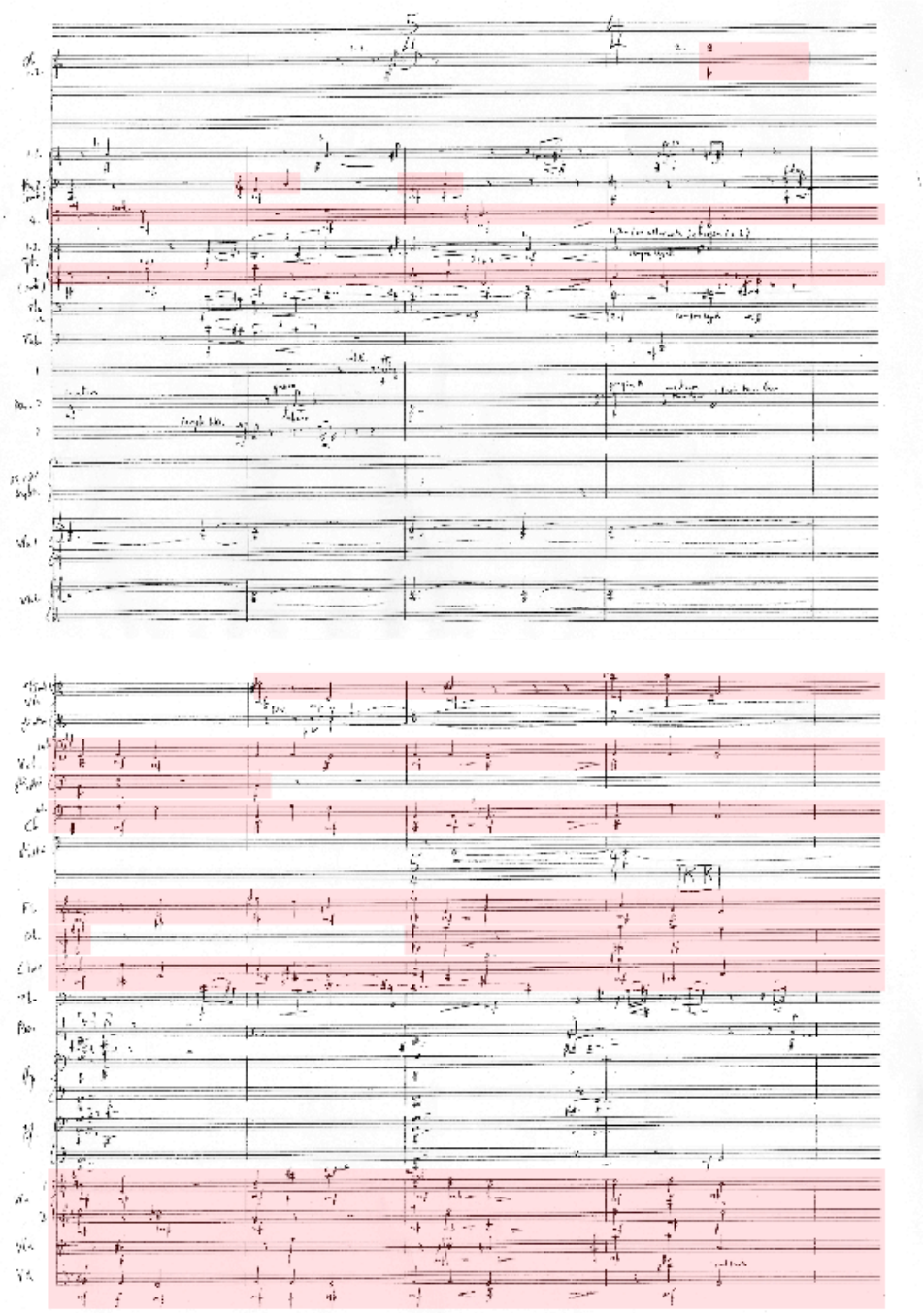

FIG. 14.8 – Mesures 11 à 14 de l'ostinato dans la partition de *Speakings* (Jonathan Harvey). Les parties écrites à l'aide du système d'orchestration sont surlignées en rose.

qui en résulte est de très faible intensité, avec une forte composante bruitée. Si les « tratto » sont fréquents dans le jeu en solo ou en petit ensemble, ils sont plus difficiles à utiliser dans un contexte orchestral où ils risquent d'être masqués par d'autres instruments.

En revanche, les changements de dynamique (qui surviennent presque à chaque note) ont été majoritairement conservés par le compositeur avec, en début de passage, une note à l'attention du chef d'orchestre : *« Great care to respect the dynamics. »* L'outil d'orchestration a donc ici été d'un grand secours pour parvenir à un ajustement précis des intensités.

Cet exemple est pour nous le signe d'une double réussite. Tout d'abord, il prouve que l'outil d'orchestration, bien qu'ayant dans un premier temps été conçu pour produire des timbres statiques, peut, à l'aide d'un système de contraintes adéquat, être utilisé pour contrôler l'évolution temporelle du timbre. En second lieu, il a suscité l'intérêt et l'enthousiasme d'un compositeur émérite, peu familier des outils technologiques, et dont les talents d'orchestrateur sont avérés et reconnus depuis longtemps. Cette collaboration aura permis de parcourir toutes les étapes de la création musicale, de l'idée du compositeur à l'exécution en concert : *Speakings* a été créé le 19 août 2008 au Royal Albert Hall de Londres, interprété par le BBC Scottish Symphony Orchestra, dirigé par Ilan Volkov.

14.5 Un exemple de production sonore (Gérard Buquet)

Au cours d'entretiens avec le compositeur Gérard Buquet lors de sa dernière visite à l'IRCAM, a germé la possibilité inattendue d'utiliser le système d'orchestration non pas comme une aide à l'écriture, mais comme un environnement de production sonore. L'idée de Gérard Buquet consiste à « récupérer » les flux audio sortant de notre système et les utiliser comme une « matière » concrète pour l'écriture des parties électroniques. Travaillant alors sur une pièce pour saxophone soprano et électronique, le compositeur a expérimenté le système avec pour cible un son de multiphonique de saxophone baryton. L'interface d'édition des solutions en Max/MSP permet de facilement rajouter des échantillons et de les « mixer » (c'est-à-dire d'ajuster leurs niveaux sonores et leurs positions dans l'espace stéréophonique) jusqu'à parvenir au timbre souhaité. En ce sens, l'outil d'aide à l'orchestration permet donc de fournir une « matière » sonore issue du monde instrumental. Mais ce n'est pas tout. Si les échantillons instrumentaux ont jusqu'ici été conçus comme les composantes d'un timbre unifié, rien n'empêche de les utiliser pour leurs caractéristiques sonores propres. Considérés isolément, ils peuvent se prêter à tous types de transformations et de mélanges, dont le timbre obtenu à l'aide de l'outil n'est qu'un exemple parmi d'autres.

La figure 14.9 montre de quelle manière une proposition d'orchestration imitative (ici le multiphonique de saxophone baryton utilisé par Gérard Buquet) peut être utilisé pour générer un matériau électronique. Notre outil d'orchestration dispose en effet d'une interface d'édition manuelle des solutions, qui se présente comme une table de mixage traditionnelle dont chaque piste correspond à un instrument. La fenêtre inférieure permet de rajouter facilement des sons extérieurs à l'orchestre (y compris électroniques). La modularité de Max/MSP intervient ici de manière essentielle. Plutôt que de diriger les flux audio sortant de chaque piste vers la sortie son de l'ordinateur, on peut très facilement les envoyer vers des modules de traitement sonore. Le timbre initial devient alors le générateur d'un matériau concret riche et varié, à travers un processus qu'on pourrait appler « synthèse par transformation de composantes ». Dans l'exemple de la figure 14.9, les composantes de l'orchestration sont invididuellement

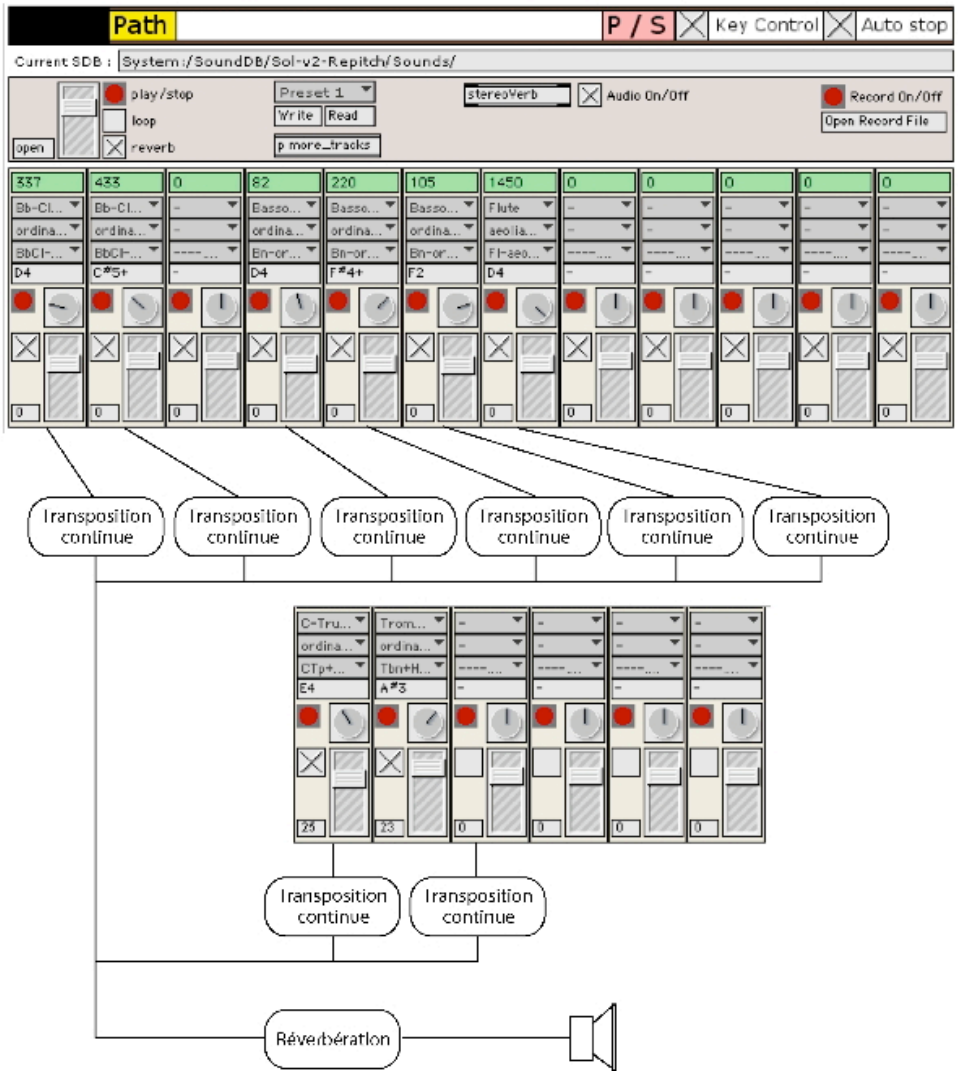

FIG. 14.9 – Utilisation de l'outil comme environnement de production sonore

transposées au cours du temps, avec un rapport différent pour chaque son. On obtient ainsi une texture qui s'« étire » peu à peu, à la frontières des mondes acoustique et électronique.

Résumé de la troisième partie

Un prototype expérimental d'aide à l'orchestration permet aujourd'hui aux compositeurs de réaliser des orchestrations imitatives, c'est-à-dire d'analyser un son pré-enregistré et de le reproduire avec l'orchestre de leur choix. Autour d'un noyau central en MATLAB©, ce prototype comprend un certain nombre de modules en MAX/MSP pour la simulation des propositions d'orchestration. Lorsque le compositeur ne dispose pas du timbre à reproduire, une interface de synthèse permet d'en construire un avatar de synthèse dans OPENMUSIC à partir d'un accord écrit (orchestration générative).

L'exploration des solutions retournées par le système est favorisée par une interface multi-points de vue dans laquelle différentes « lectures » du timbre sont mises en relation. Espace symbolique de l'écriture musicale, espace des descripteurs perceptifs et espace des critères d'opimisation communiquent en répercutant les actions effectuées dans un espace sur les deux autres. Au sein de chaque représentation, un algorithme de clustering permet de considérer des « catégories » de solutions plutôt que des solutions individuelles. Dans l'espace des décisions, cette opération est rendue possible grâce à un « clustering hiérarchique par schémas » qui construit une représentation arborescente des solutions selon un attribut symbolique de l'écriture. Chaque solution d'orchestration peut être éditée manuellement par l'utilisateur, ainsi que servir de point de départ pour une nouvelle recherche. Dans ce dernier cas, les préférences implicites du compositeur sont inférées à partir des valeurs de critères de la solution choisie, puis injectées dans l'algorithme d'orchestration. La recherche s'intensifie alors autour de la solution intermédiaire.

La connaissance instrumentale réduite dont dispose aujourd'hui l'outil d'orchestration, ainsi que sa « vision » essentiellement spectro-harmonique du timbre le destinent pour l'instant avant tout à la création de textures harmoniques complexes dans lesquelles la hauteur est un élément essentiel de la structure. Une série d'exemples concrets tirés de collaborations avec les compositeurs montrent toutefois que ce cadre restrictif peut souvent être dépassé grâce au savoir et à l'expérience de ces derniers. En outre, l'heuristique *CDCSolver* offre la possibilité d'un contrôle temporel du timbre à l'aide de contraintes globales.

Conclusion

Synthèse

En abordant le problème de l'aide à l'orchestration, nous nous sommes posé la question de la complexité de cette pratique. Comment penser cette complexité ? Comment concevoir des formalismes qui la traduisent et l'expriment au mieux, sans la réduire à un modèle simplificateur ? En quoi un outil informatique peut-il aider les compositeurs à faire face à cette complexité ? Telles sont les questions qui ont animé ce travail de thèse.

Nous avons montré que la complexité de l'orchestration était multiple, qu'elle se situait à la rencontre de plusieurs axes, plusieurs « faisceaux » de complexité : complexité de la perception du timbre, complexité combinatoire des mélanges instrumentaux, complexité de nos rapports aux structures temporelles en musique. Pour chacun d'entre eux, nous avons proposé des formalismes et des méthodes adaptés.

Le caractère multidimensionnel du timbre a été traité à travers une approche descriptive. L'extraction automatique de descripteurs perceptifs dans de grandes banques d'échantillons sonores a permis d'aboutir à un modèle robuste du timbre instrumental, capable de prédire les propriétés acoustiques des mélanges. Nous avons ensuite proposé un algorithme de recherche d'orchestrations fondé sur des métaheuristiques d'optimisation multicritère. Une recherche interactive et supervisée permet la découverte simultanée d'orchestrations pertinentes et des dimensions privilégiées de l'écoute pour un problème donné. Nous avons enfin montré que la contextualisation des orchestrations dans un processus compositionnel plus général était possible à l'aide de contraintes sur les variables symboliques de l'écriture, et que cette approche permettait en outre d'aborder le problème du temps.

Notre travail va donc bien au delà des potentialités des quelques outils actuels d'aide à l'orchestration. Il permet à l'informatique musicale d'aborder pleinement une discipline de l'écriture qui jusqu'alors s'était toujours soustraite à la formalisation, et laisse entrevoir la possibilité d'une compréhension plus fine des rapports entre deux mondes hétérogènes, celui de la partition comme système symbolique et celui du timbre comme réalité sonore.

Nous disposons donc aujourd'hui d'un prototype d'aide à l'orchestration capable de proposer rapidement au compositeur un ensemble diversifié d'orchestrations pour un timbre statique donné en entrée, et d'en contrôler l'évolution temporelle à l'aide d'un système de contraintes. Que cet outil fut à plusieurs reprises utilisé avec enthousiasme et succès par des compositeurs renommés dans leurs créations récentes constitue, au delà de notre apport scientifique, la preuve tangible d'une avancée significative.

En résumé, la contribution de notre travail à la littérature se résume en sept points :

- une formalisation générique et extensible du problème de l'aide à l'orchestration comme une tâche d'optimisation multicritère sous contraintes ;
- une validation expérimentale de la pertinence de descripteurs spectro-harmoniques pour la caractérisation perceptive du timbre dans le cadre de sons complexes ;

- un algorithme évolutionnaire d'optimisation combinatoire multicritère adapté au problème de l'orchestration, respectant l'équilibre entre intensification et diversification de la recherche et pouvant tenir compte des préférences de l'utilisateur.
- une généralisation des notions d'*instanciation partielle* et de *consistance locale* dans les problèmes de satisfaction de contraintes, à travers les concepts de *contraintes de design* et *contraintes de conflit* ;
- une métaheuristique de recherche locale pour la satisfaction de contraintes globales, s'appuyant sur la distinction entre contraintes de *design* et contraintes de *conflit*.
- une validation expérimentale, sur des instances de petite taille, de l'efficacité des deux méthodes précédentes ;
- un prototype expérimental d'aide à l'orchestration facilitant l'exploration du timbre orchestral via une représentation multi-points de vue des solutions et favorisant la découverte des préférences implicites du compositeur.

Extensibilité à court terme de l'outil d'orchestration

Au terme de ce travail, nous devons distinguer deux horizons de recherches futures. Avant d'entamer de nouveaux chantiers, il est un certain nombre de points sur lesquels nous pouvons avancer dans un relatif court terme, afin de proposer rapidement aux compositeurs un outil plus complet.

Extension de l'instrumentarium

Pour de multiples raisons que nous ne rappellerons pas ici, nous nous sommes restreints dans notre travail à une description spectro-harmonique du timbre, limitant ainsi la connaissance instrumentale aux sons harmoniques, entretenus et sans variations temporelles. Or cette large classe de sons n'est pas intégralement représentée par l'instrumentrium actuel au sein de notre système. Manquent en particulier les sons de tuba, accordéon, saxophones, contrebasson, clarinette basse, trombone basse, flûte picolo, flûte basse. Ces instruments produisent des timbres accessibles à nos descripteurs et leur utilisation est fréquente dans les grands orchestres comme dans les ensembles. La généralisation de notre outil d'orchestration pourrait donc commencer par l'ajout de ces instruments, ainsi qu'à la possibilité pour l'instrumentarium complet de jouer en quarts de tons.

Intégration des modèles d'instruments

Notre vision exclusivement spectrale du timbre est aujourd'hui sur le point d'être étendue par les résultats des travaux menés en parallèle des nôtres à l'IRCAM sur les modèles d'instruments. De nouvelles composantes du timbre, telles que le bruit ou la modulation d'amplitude, pourront être prises en compte. Il conviendra alors d'évaluer l'intérêt de ces modèles probabilistes, en comparant leurs performances à celles de notre approche descriptive actuelle. Si les résultats sont concluants, la connaissance instrumentale de notre outil pourra alors être étendue aux sons bruités et modulés.

Perspectives de recherche à plus long terme

A plus long terme, la pluridisciplinarité de l'aide à l'orchestration laisse entrevoir de multiples directions de recherche. Nous ébauchons ici quelques pistes de travail.

Modélisation des unissons

Dans notre système, les échantillons de la connaissance instrumentale proviennent tous d'instruments enregistrés en solo. Notre outil a donc tendance à considérer l'orchestre comme un ensemble de solistes, et ne peut tenir compte des effets d'orchestration que l'on tire des unissons, c'est-à-dire plusieurs mêmes instruments jouant exactement la même partie (violons par huit, violoncelles par six, clarinettes par trois, etc.). Si une modélisation des unissons est donc nécessaire, elle ne se réduit pas pour autant à l'enregistrement et à l'analyse de ces sons. Se pose notamment un problème majeur d'allocation des ressources instrumentales. Dans un orchestre comprenant huit violons, ces derniers peuvent jouer tous à l'unisson, par groupe de quatre, par groupe de six avec deux violons solo, etc. Notre représentation actuelle des orchestrations, basée sur le maintien implicite de la consistance de capacité, ne peut exprimer ces multiples possibilités. Une procédure de réparation des solutions violant les contraintes de capacité devra donc être envisagée.

Contraintes semi-globales explicites

Lles contraintes exprimables par le langage défini au chapitre 10 ne peuvent pas concerner plusieurs attributs simultanément. On peut dépasser cette limitation en remarquant que ces contraintes sont de deux types. Les *Contraintes globales* concernent l'ensemble des variables d'une configuration. Typiquement, les contraintes *all-diff*, *at-least-diff* et *at-most-diff* sur l'attribut « note » sont des contraintes globales. A l'opposé *Contraintes semi-globales implicites* concernent un sous-ensemble précis de variables, implicitement désigné par la valeur de l'attribut intervenant dans la contrainte. Par exemple, la contrainte « au moins trois instruments doivent jouer *sul ponticello* » ne s'applique évidemment qu'aux instruments à cordes.

Il est alors possible d'imaginer, sans remettre en cause le formalisme de ce langage élémentaire, des *contraintes semi-globales explicites*, en rajoutant un cinquième champ *valeur* dans la syntaxe. En présisant dans ce cinquième champ les variables sur lesquelles la contrainte doit être calculée, on peut ainsi facilement exprimer des contraintes plus élaborées, où peuvent intervenir plusieurs attributs, telles que : « tous les trombones jouent à la même dynamique », « le groupe de hauteurs (C3,G3,D4) est au moins présent une fois aux cordes », « le groupe violon/flûte/clarinette joue à l'unisson », etc.

Le problème des bicordes

Comme leur nom l'indique, les bicordes désignent une technique de jeu des instruments à cordes qui consiste à jouer deux cordes simultanément. Ce procédé est particulièrement intéressant dans le cas d'orchestrations pour petits ensembles, où il s'agit souvent de produire l'effet maximal avec l'effectif minimal. Au sein de notre représentation, les bicordes pouront être modélisés par deux instruments identiques liées par une contrainte binaire. La difficulté sera alors de faire coexister contraintes binaires et contraintes globales au sein d'une heuristique de réparation. Si les contraintes binaires sont définies explicitement, un filtrage par réduction de domaines avant l'exploration du voisinage courant sera peut-être envisageable.

Inférence automatique des critères pertinents

Si les progrès à venir sur la caractérisation du timbre instrumental permettront d'en considérer de nombreuses dimensions supplémentaires, ils augmenteront aussi considérablement le nombre de critères dans le processus d'optimisation. Or, plus le nombre de critères est en effet élevé, plus une configuration aléatoire a une probabilité forte d'être non-dominée. L'utilisateur risque alors d'être confronté à des ensembles de solutions gigantesques dans lesquels l'identification d'une configuration pertinente peut s'avérer longue et fastidieuse.

Dans un contexte d'aide à la décision mutli-critères comme l'orchestration assistée par ordinateur, la singularité d'une solution tient moins à un jeu de valeurs précises pour un grand nombre de critères qu'à l'identification de descripteurs adéquats pour le problème considéré. Il est inutile d'accumuler les critères de décision si ceux-ci ne permettent pas de distinguer perceptivement les solutions. Une description approfondie et multidimensionnelle du timbre instrumental est donc certes un pré-requis indispensable pour l'aide à l'orchestration, à condition toutefois de pouvoir proposer à l'utilisateur un nombre réduit de critères pertinents pour un problème particulier. A terme, il conviendrait donc d'imaginer un moyen de déduire de l'analyse de la cible les meilleurs critères pour une optimisation dans un espace de dimension acceptable.

Inférence automatique de contraintes

Une autre piste de recherche est la déduction automatique de contraintes à partir de l'analyse de la cible sonore. Si le timbre à reproduire comporte par exemple une modulation d'amplitude, on peut alors imposer que l'orchestration contienne au moins un son tremolo. D'un point de vue théorique, il s'agirait donc de déduire, à partir des valeurs de descripteurs de la cible, un certain nombre de caractéristiques d'un niveau d'abstraction plus élevé, sous formes de prédicats. Dans un second temps, les valeurs de ces prédicats devraient être interprétées, à l'aide d'un système de règles prédéfinies, en contraintes symboliques.

Fusion attaque/résonnance

Du côté de la psychoacoustique et de la perception du timbre, les phénomènes de fusion entre attaque et résonnances sont également à investiguer. En orchestration, les sons non entretenus sont fréquemment employés pour leurs attaques rapides. Dès le début de XX^{e} siècle, les compositeurs ont créé des timbres nouveaux en les associant aux vents ou aux cordes jouées à l'archet. Le caractère inouï de ces sonorités résulte de leur aspect à la fois percussif et soutenu : ne s'apparentant à aucun instrument connu, elles sont très déroutantes pour l'écoute qui les perçoit comme un timbre « unifié ». L'intégration des instruments à sons non entretenus au sein de notre système passe donc par une compréhension des phénomènes de couplage entre une attaque rapide produite par une catégorie d'instruments et une résonnance assurée par une autre.

Ecriture de l'électronique

Nous avons vu en dernière partie de cette thèse que l'outil d'orchestration pouvait non seulement aider à l'écriture instrumentale, mais également servir d'environnement de création sonore pour des parties électroniques dans le cadre de musiques mixtes. En admettant que du point de vue de la modélisation du timbre les sons électroniques peuvent être représentés par les

mêmes descripteurs que les sons instrumentaux, l'orchestration avec parties électroniques ne se résume pas à l'ajout d'instruments supplémentaires. Dans notre modèle actuel d'orchestration, chaque son de la connaissance instrumentale est soumis à des contraintes d'intégrité (soit il est utilisé dans l'orchestration, soit il ne l'est pas) et de capacité (le nombre total de sons provenant d'un même instrument est limité par l'effectif orchestral). Ces contraintes n'ont plus cours pour l'écriture des parties électroniques : les sons sont mélangeables à volonté, et dans toutes les proportions de volume possibles. Autrement dit, la représentation même des orchestrations est dans ce cas à repenser. Quant à l'algorithme d'optimisation, il devra pouvoir traiter simultanément variables binaires et variables continues.

Le problème du temps

Au commencement de nos recherches sur l'aide à l'orchestration, nous avons fait le choix de laisser de côté, dans un premier temps, les aspects temporels du timbre. Nous avons défini une cible à orchestrer comme un ensemble de descripteurs spectro-harmoniques statiques, calculés sur une partie supposée « stable » du son. En conséquence, notre outil est capable d'assister les compositeurs dans la création de textures complexes, d'une grande richesse spectrale, mais ne peut pas générer de « mouvements ».

Malgré cela, la question d'un timbre évoluant dans le temps, n'a jamais cessé, au cours de nos recherches, d'être posée. Peut-on considérer l'orchestration dynamique comme une simple extension du modèle statique, où l'évolution au cours du temps ne serait rien d'autre qu'une succession de « segments » ? Ou cela nécessite-t-il au contraire une modélisation temporelle des possibilités instrumentales ? Il n'est pas dit que nous disposions à l'heure actuelle des outils et du recul nécessaire pour décider de la pertinence et de la faisabilité de chacune des approches, mais nous pouvons déjà apporter un éclairage sur cette question en distinguant deux approches du temps.

D'un point de vue formel, si l'évolution du timbre dans le temps est considérée comme un ensemble de relations d'antériorité, de postérité ou de simultanéité, on peut alors décider que tel timbre est suivi de tel autre, se superpose à tel troisième, se « fond » dans tel quatrième, etc. Il s'agit évidemment d'une vision segmentée de l'orchestation, dans laquelle les couleurs se succèdent comme des « panneaux » détachés les uns des autres. A priori, cette approche du temps ne remet pas en cause notre modèle combinatoire, car l'horizontalité de l'écriture y est pensée comme une succession de segments verticaux, dont le contrôle est possible à l'aide d'un ensemble de contraintes sur les variables de l'écriture musicale.

Nous devons toutefois reconnaître que cette approche ne permet de traiter que des variations de timbre relativement lentes, et rien ne nous laisse aujourd'hui prévoir la finesse temporelle que nous pourrons atteindre par ce biais. Une autre vision du temps, horizontale et continue, semble donc nécessaire, et implique l'élaboration d'un modèle temporel du timbre. Vaste problème, qui reporte la complexité de l'orchestration sur celle du jeu instrumental. Si une recherche future s'engage dans cette voie, il lui faudra sans doute abandonner, ne serait-ce que temporairement, l'approche combinatoire, verticale, de l'orchestration pour se concentrer sur une modélisation des enchaînements de timbres au cours du temps.

Quatrième partie

Annexes

Annexe A

Descripteurs : extraction, agrégation, comparaison

Nous explicitons dans cette annexe les méthodes d'extraction des descripteurs introduits en 8.1.2. Nous définissons également les fonctions d'agrégation et fonctions de comparaison pour chacun d'entre eux.

Principaux partiels résolus (MRP)

Extraction L'extraction des MRP se fait en plusieurs étapes résumées par la figure A.1. Dans un premier temps, les partiels du signal sont calculés fenêtre par fenêtre grâce à une FFT et une extraction de pics spectraux. Ensuite, une procédure de suivi de partiels permet de ne retenir que les partiels présents sur l'intégralité du son. Ces deux premières étapes sont réalisées par le programe *pm2* [Rod97] qui sert en partie de noyau au logiciel AudioSculpt [BR05].

Dans un troisième temps, un modèle perceptif est utilisé pour sélectionner les partiels résolus par le système auditif, ainsi qu'en estimer la correction d'amplitude et de fréquence opérée par l'ensemble de la chaîne perceptive. Notre modèle auditif utilise une mesure ERB (*Equivalent Rectangular Bandwidth* [GM00]) pour simuler les bandes critiques de l'oreille interne et identifier les partiels résolus.

Le nombre de MRP varie selon le type de cible. Pour les cibles harmoniques, une quinzaine de partiels suffisent en général. Les cibles plus complexes en nécessitent souvent davantage (entre 20 et 30).

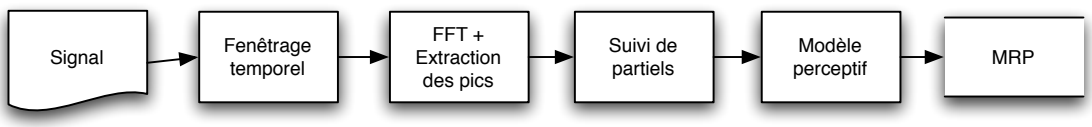

Fig. A.1 – Processus d'extraction des principaux partiels résolus (MRP)

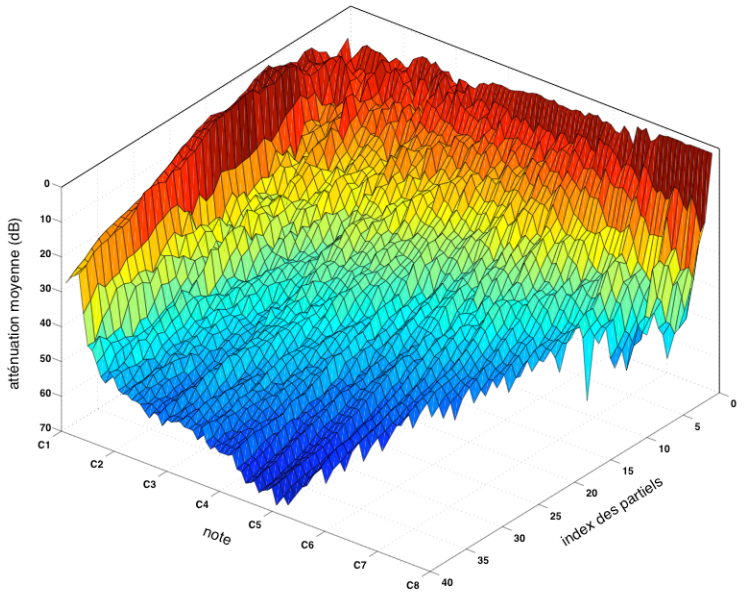

FIG. A.2 – Enveloppes spectrales moyennes des les sons entretenus de SOL

Agrégation Les MRP ne sont calculés que pour la cible. Les sons de la base d'échantillons instrumentaux étant harmoniques (voir section 8.1.1) et leur enveloppe spectrale — à l'exception du registre grave — généralement décroissante (voir figure A.2), leurs MRP se résument la plupart du temps aux premiers partiels. Tirant parti de ce constat, nous introduisons une méthode d'agrégation efficace en complexité, permettant de calculer la contribution des partiels d'un mélange instrumental aux MRP de la cible, mais nécessitant un prétraitement préalable des sons de la base.

Pour chaque son nous identifions les partiels qu'il partage avec la cible. Le critére d'identification est un seuil de tolérance sur les fréquences et correspond environ à un huitième de ton, soit un écart maximal de fréquence inférieur à 1.5%. Par exemple, si la cible possède son $3^{\text{ème}}$ MRP à 500 Hz, tous les sons de la base ayant un partiel entre 493 Hz et 507 Hz verront ce partiel identifié avec le $3^{\text{ème}}$ MRP de la cible. (On notera par la suite $f_1 \sim f_2$ pour désigner deux fréquences dans un rapport inférieur au seuil de tolérance.)

Soit T la cible à orchestrer et $\{f_k^T\}$ et $\{a_k^T\}$ les vecteurs de fréquences et amplitudes des MRP de T. Soit S un son de la base de données dont les partiels ont respectivement pour fréquences et amplitudes $\{f_k^S\}$ et $\{a_k^S\}$. La contributiton de S aux $j^{\text{ème}}$ MRP de T sera alors :

$$\begin{cases} a_{k_0}^i \text{ si } \exists k_0, f_{k_0}^S \sim f_j^T \\ 0 \text{ sinon} \end{cases}$$

Plus simplement, on retiendra qu'on associe à chaque son S_i de la base de données un vecteur d'amplitudes $\left\{MRP_S^T(j)\right\}$ de même taille que $\{a_k^T\}$ exprimant la contribution de S à chaque MRP de T, cette contribution dépendant de l'identification des fréquences des partiels de S et des MRP de T.

L'agréagation des contributions aux MRP est alors une opération très simple. Soit $(S_i)_{i\in I}$ un ensemble de sons de la base de données et K la mixture résultant de leur mélange. On a

alors :

$$\widehat{MRP}_K^T = \left\{ \max_{i \in I} \left(MRP_{S_i}^T(1) \right), \max_{i \in I} \left(MRP_{S_i}^T(2) \right), \dots \right\} \tag{A.1}$$

Une fonction de maximum est préféré à une addition des amplitudes en raison des phénomènes de phase évoqués en 4.3 et responsables de la non-additivité des spectres.

La figure A.3 illustre les processus de calcul de contributions des MRP ainsi que de l'agrégation des contributions. Sur le diagramme supérieur, on a reporté le spectre d'un son cible à orchestrer (en pointillés) ainsi que les principaux partiels résolus (MRP, en trait plein). Le diagramme médian est un exemple de contribution aux MRP de deux sons de hauteurs respectives C2 et D2. Il y apparaît par exemple que le $4^{\text{ème}}$ partiel du C2 a été identifié au $3^{\text{ème}}$ MRP, que les $9^{\text{ème}}$ partiel de C2 et $8^{\text{ème}}$ partiel de D2 ont été identifiés au $13^{\text{ème}}$ MRP. Les contributions de chaque son se ramènent par ce procédé à des vecteurs d'amplitude de même taille que les MRP et dont les valeurs ne dépendent ques des amplitudes des partiels de ces sons. Enfin, le diagramme inférieur illustre l'agrégation des contributions.

Comparaison Soit D_T^{MRP} la fonction de comparaison associée aux MRP pour la cible T, c'est-à-dire une mesure de distance entre une mixture K de sons instrumentaux et T dans le sous-espace associé aux MRP. Nous la définissons par :

$$D_T^{MRP}(K) = 1 - \cos\left(\widehat{MRP}_K^T, MRP_T^T \right) \tag{A.2}$$

La distance sera donc nulle si les deux vecteurs d'amplitudes sont dans un rapport de proportionalité. Ce degré de liberté est nécessaire pour éviter toute influence du niveau d'enregistrement du son donné en entrée du système. Avec cette méthode, les sons sont donc comparés sur la forme de leurs enveloppes spectrales.

N.B. Si la contribution d'un son aux MRP augmente avec le nombre de partiels identifiés à ceux de la cible, en revanche aucune mesure ne pénalise les partiels non identifiés. La raison principale est qu'il n'y a pas de conduite particulière à tenir en pareil cas. Dans l'exemple de la figure A.3, les $5^{\text{ème}}$ et $8^{\text{ème}}$ partiels de C2 sont bien présents dans la cible — mais d'amplitude trop faible pour être résolus. Il n'y a donc aucune raison pénaliser leur présence. D'autres cas peuvent ce présenter, pour lesquels l'ajout d'un son dans une mixture introduit des partiels absents dans la cible et que la fonction de comparaison A.2 ne peut pas « voir ». C'est là une limite inhérente à la modélisation en MRP de la partie basse fréquence du spectre. Mais ce que nous perdons en précision d'estimation, nous le gagnons en rapidité de calcul : Les contributions des sons de la base de données aux MRP sont calculés une seule fois pour une cible donnée, et l'évaluation des mixtures selon ce critère n'implique que des opérations élémentaires sur des vecteurs d'amplitudes. Une comparaison ayant recours aux informations de d'amplitude *et* de fréquence aurait été beaucoup plus coûteuse en temps de calcul.

Intensité perceptive instantanée (*loudness*)

L'intensité perceptive (ou sonie, ou *loudness*) n'est pas un descripteur explicitement utilisé par notre système dans ses jugements de similarité entre la cible et les propositions d'orchestration. Néanmoins, il intervient dans les fonctions d'agrégation des descripteurs spectraux

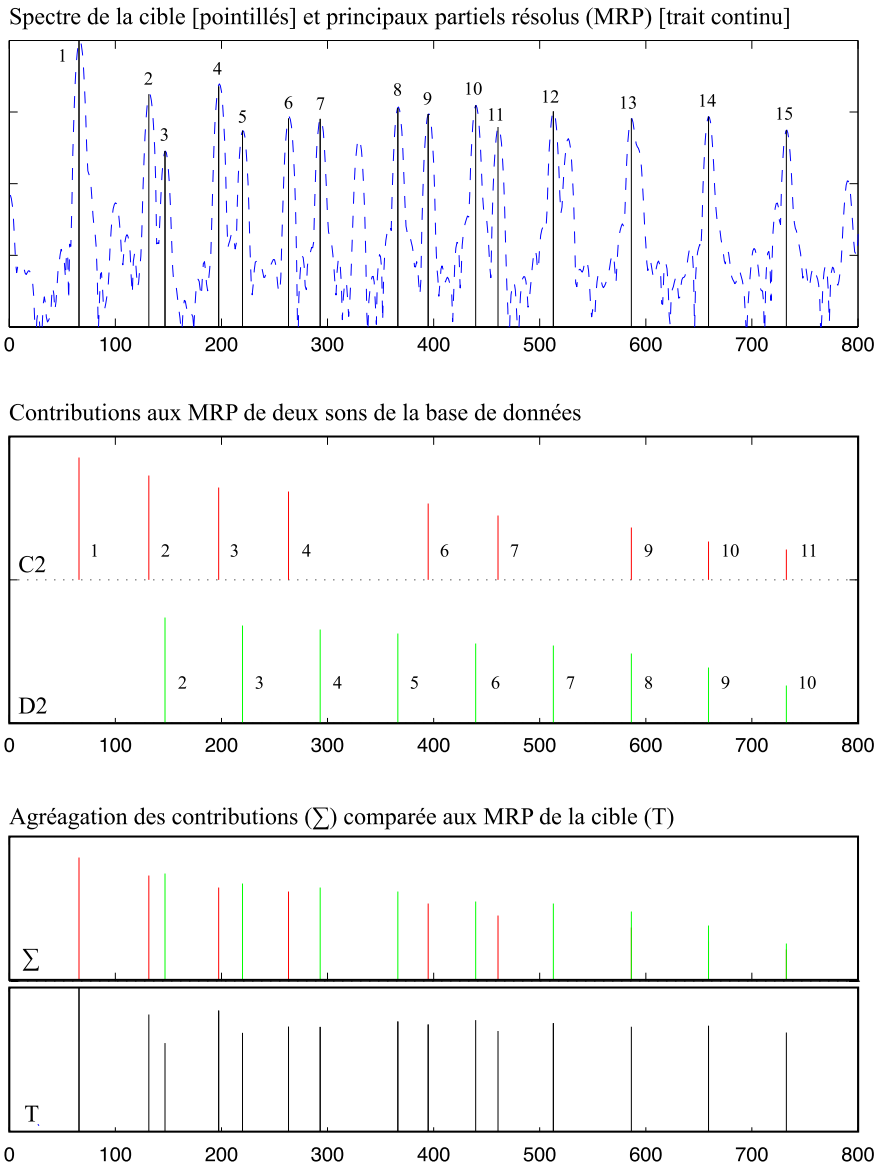

FIG. A.3 – Exemple d'extraction et d'agrégation des MRP

FIG. A.4 – Processus d'extraction de l'intensité perceptive (*loudness)*

décrits en infra. Nous ne décrivons donc dans ce paragraphe que sa méthode d'extraction. Il est calculé pour tous les sons de la base de données mais n'est pas extrait de la cible, puisque nous considérons l'intensité de cette dernière comme non significative.

L'extraction de l'intensité perceptive est schématisée figure A.4. Pour chaque fenêtre temporelle le spectre instantané du signal est extrait par FFT. Un modèle perceptif calcule ensuite l'intensité perceptive instantanée pour chaque fenêtre. Ce modèle applique d'abord un premier filtre simulant la fonction de transfert de l'oreille moyenne [MGB97], puis utilise une mesure ERB [GM00]) pour simuler les bandes critiques de l'oreille interne et calculer l'intensité dans chaque bande. L'intensité totale instantanée est obtenue par sommation de l'instensité dans les bandes critiques. Cette extraction est réalisée par le programme *ircamdescriptor* [Pee04]. Le lecteur trouvera davantage de précision sur la définition et le calcul de l'intensité perceptive dans [MGB97].

Centroïde spectral

Le centroïde spectral est communément défini comme le barycentre spectral du son. D'un point de vue perceptif, il est corrélé à la brillance.

Extraction Le calcul du centroïde (ou premier moment spectral) se fait un deux étapes. Dans un premier temps, le centroïde instantané du signal (défini comme le barycentre du spectre d'amplitude) est extrait pour chaque fenêtre temporelle. La moyenne des centroïdes instantanés pondérée par l'intensité perceptive instantanée définit alors le centroïde global. La figure A.5 résume ce processus.

Agrégation L'estimation du centroïde d'un mélange nécessite la connaissance non seulement des centroïdes de chaque élément du mélange mais aussi de leur intensités. Soit $(s_i)_i$ les sons de la base de données, décrits par leurs centroïdes $(sc_i)_i$ et leurs intensités perceptives $(L_i)_i$. Soit $K = (s_{i_1}, \ldots, s_{i_K})$ une configuration, on a alors :

$$\widehat{sc}_K = \frac{\sum_{i_1}^{i_K} L_i^{1/0.6} sc_i}{\sum_{i_1}^{i_K} L_i^{1/0.6}} \tag{A.3}$$

L'exposant 1/0.6 résulte d'une approximation de la relation entre l'intensité perceptive et le spectre d'amplitude $(f_i, a_i)_i$ du signal, qu'on peut exprimer ainsi :

$$L \simeq \left(\lambda \sum_i a_i \right)^{0.6}$$

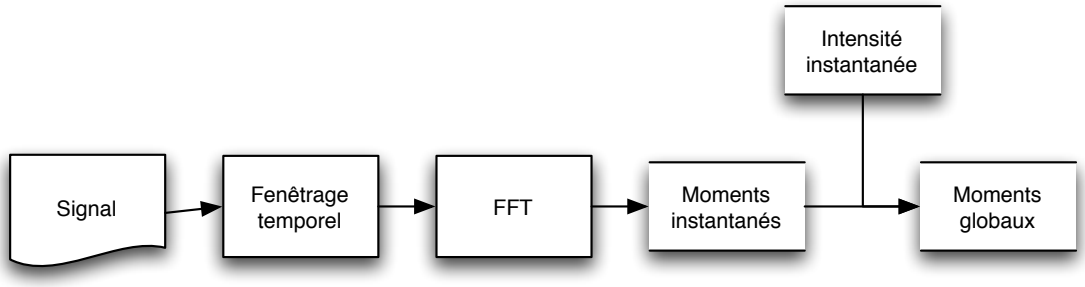

FIG. A.5 – Processus d'extraction des moments spectraux

Où λ est un facteur de proportionalité dont la valeur n'intervient pas dans l'équation A.3.

Comparaison Soient T la cible à orchestrer, sc_T son centroïde et K une proposition d'orchestration. Nous choisissons comme fonction de comparaison relative au centroïde spectral une simple distance relative :

$$D_T^{sc}(K) = \frac{|\widehat{sc}_K - sc_T|}{sc_T} \tag{A.4}$$

Etendue spectrale (*spread*)

L'étendue spectrale (ou second moment spectral) est l'écart-type du spectre d'amplitude. D'un point de vue perceptif, il traduit la « largeur » (ou « l'épaisseur ») spectrale du son. Même si on constate en pratique une corrélation significative entre le centroïde et l'étendue, ces deux descripteurs conservent toutefois une certaine indépendance. Ainsi, une flûte piccolo dans le registre aigu peut avoir un son très brillant et très pincé, alors qu'un cor dans le grave a un son sourd et large (par comparaison à un orgue par exemple).

Extraction Le processus d'extraction du spread est similaire à celui du centroïde (voir section A et figure A.5).

Agrégation Soit K une proposition d'orchestration de centroïde $\widehat{sc}_K$. on a alors :

$$\widehat{ss}_K = \sqrt{\frac{\sum_i L_i^{1/0.6}(sc_i^2 + ss_i^2)}{\sum_i L_i^{1/0.6}} - \widehat{sc}_K^2} \tag{A.5}$$

Comparaison Soit T la cible à orchestrer d'étendue spectrale ss_T. Soit K une proposition d'orchestration d'étendue $\widehat{ss}_K$. La fonction de comparaison s'écrit :

$$D_T^{ss}(K) = \frac{|\widehat{ss}_K - ss_T|}{ss_T} \tag{A.6}$$

Annexe B

Norme induite par une configuration

Etant donné un point $x = (x_1, \ldots, x_N) \in C$ de l'espace des critères, nous avons défini au paragraphe 8.5.1 la *norme induite* par x (notée $\|\cdot\|_x$) comme la norme pondérée de Tchebycheff vérifiant :

$$\forall (i,j) \in \{1;N\}^2,\ \lambda_i x_i = \lambda_j x_j \tag{B.1}$$

La propriété 8.2 garantit que la valeur de la norme induite par une configuration x est inférieure à toutes les autres normes pondérées de Tchebycheff pour x. C'est la norme qui optimise le classement x sur l'espace de recherche. La figure illustre cette propriété. L'espace des critéres y est « ordonné » jusqu'à x (représentée par un carré noir) selon trois normes pondérées de Tchebycheff. Les solutions en rouge ont un rang inférieur à x, elles sont donc préférées à x pour la norme considérée. De façon évidente, seule la norme plaçant la solution x sur le sommet d'un hyper-rectangle correspondant à une surface isométrique permet de minimiser le nombre de solutions de meilleur rang que x. Cette position correspond à l'équation B.1 de la norme induite.

La propriété 8.2 assure en outre que si une solution x minimise $\|\cdot\|_x$, alors x est non-dominée et vice-versa. La figure B.2 permet de s'en convaincre aisément. Seule la norme induite par une configuration efficiente permet de classer cette dernière en premier lorsque l'espace de recherche est ordonné selon cette norme. C'est cette propriété qui a permis à Chen & Liu [CL94] et Steuer [Ste86] d'établir l'équivalence entre un problème multicritère et un ensemble de problèmes mono-critère « scalarisés » (voir sections 5.7 et 8.5.1).

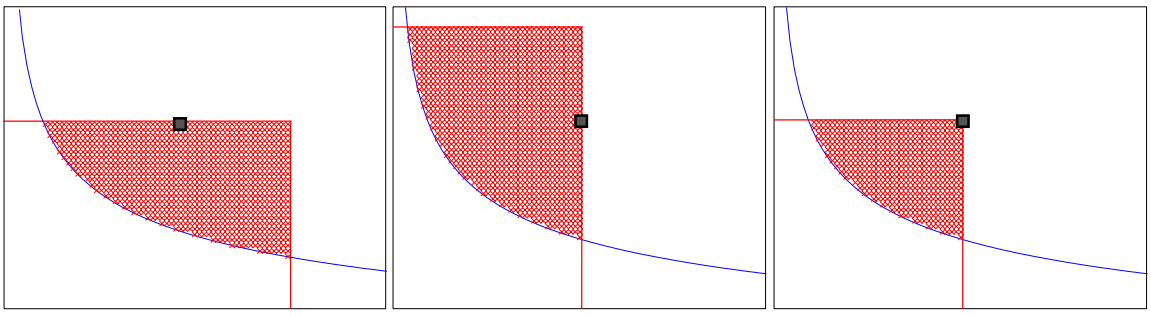

FIG. B.1 – Classement d'une configuration dominée selon trois normes de Tchebycheff

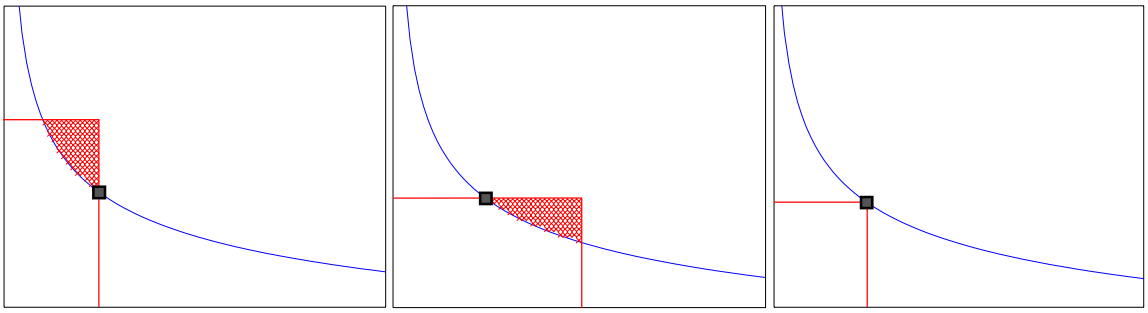

FIG. B.2 – Classement d'une configuration efficiente selon trois normes de Tchebycheff

Calcul des poids

Lorsque l'algorithme de recherche d'orchestrations se termine, le choix par le compositeur d'une solution efficiente permet de « deviner » les préférences implicites de ce dernier en termes d'écoute du timbre. Le calcul de la norme induite par la solution choisie révèle les poids associés à chaque critère, traduisant leur importance relative dans le jugement de similarité entre la cible et la solution.

Soit un point x situé sur un sommet de la boule de rayon d_x (au sens d'une norme de Tchebycheff pondérée). La somme des poids devant être égale à 1, on a en deux dimensions :

$$\begin{cases} \lambda_1 x_1 = d_x \\ \lambda_2 x_2 = d_x \\ \lambda_1 + \lambda_2 = 1 \end{cases} \tag{B.2}$$

Trivialement, les solutions s'écrivent :

$$\begin{cases} \lambda_1 = x_2/S \\ \lambda_2 = x_1/S \end{cases} \text{ avec } S = x_1 + x_2 \tag{B.3}$$

De même, en trois dimensions, le problème s'écrit :

$$\begin{cases} \lambda_1 x_1 = d_x \\ \lambda_2 x_2 = d_x \\ \lambda_3 x_3 = d_x \\ \lambda_1 + \lambda_2 + \lambda_3 = 1 \end{cases} \tag{B.4}$$

Et les solutions sont :

$$\begin{cases} \lambda_1 = x_2 x_3/S \\ \lambda_2 = x_1 x_3/S \\ \lambda_3 = x_1 x_2/S \end{cases} \text{ avec } S = x_1x_2 + x_1x_3 + x_2x_3 \tag{B.5}$$

Le calcul se généralise aisément aux dimensions supérieures :

$$\lambda_k = \frac{\prod\limits_{j \neq k} x_j}{\sum\limits_{i=1}^{K} \prod\limits_{j \neq i} x_j} \text{ , } k = 1, ..., K \tag{B.6}$$

Table des figures

Liste des tableaux

Liste des algorithmes

Glossaire

approximation Ensemble des configurations efficientes au sein d'une population donnée.

attribut Donnée symbolique décrivant un paramètre musical d'émission du son (hauteur, dynamique, sourdine...). Par opposition aux descripteurs, les attributs sont obtenus par indexation pour les échantillons sonores constituant la connaissance instrumentale de l'outil d'orchestration.

cible Ensemble des descripteurs audio caractérisant le timbre recherché.

configuration Sous-ensemble de la base d'échantillons sonores, constituant une proposition d'orchestration. Ensemble d'échantillons joués simultanément pour produire un timbre. Ensemble de couples variable/valeur dont chaque variable représente un instrument de l'orchestre, à valeurs prises dans un sous-ensemble de la base d'échantillons sonores propre à l'instrument considéré et au problème d'orchestration courant.

configuration efficiente Configuration telle qu'il n'existe, au sein d'une population donnée, aucune configuration la surpassant sur l'ensemble des crièteres.

contrainte Condition devant être vérifiée par les attributs d'une configuration.

critère Distance à la cible selon un descripteur audio unique.

descripteur Donnée numérique (scalaire, vectorielle ou matricielle) décrivant une caractéristique précise du timbre. Elle est calculée directement sur le signal.

fonction d'agrégation Estimateur d'une valeur de descripteur d'une configuration, à partir des valeurs de descripteur de ces composantes.

fonction de comparaison Pour une cible donnée, fonction transformant une valeur de descripteur d'une configuration en une valeur de critère.

fonction de coût Fonction à valeurs réelles mesurant le niveau de violation de contrainte d'une configuration donnée.

front de Pareto Ensemble des configurations efficientes parmi toutes les configurations consistantes de l'espace de recherche.

mode de jeu Technique de jeu enrichissant le vocabulaire sonore des instruments de musique.

modèle d'instrument Modèle probabiliste caractérisant les possibilités sonores d'un instrument, dont les paramètres sont estimés à partir de bases de données d'échantillons.

méthode d'extraction Fonction associée à un descripteur permettant d'en calcuer la valeur à partir du signal audio.

orchestration générative Mixture instrumentale dont les paramètres de timbre ont été définis à partir de données symboliques (par oppostion à une orchestration imitative).

orchestration imitative Mixture instrumentale dont le timbre s'apparente à celui d'un son donné par le compositeur et qu'il cherche à reproduire avec l'orchestre.

Bibliographie

[AADC08] G. Assayag A. Allombert and M. Desainte-Catherine, *De Boxes à Iscore : vers une écriture de l'interaction*, Proceedings of the 13th Journées d'Informatique Musicale (JIM'2008), Albi, France, Mars 2008.

[AD05] Jean-Marc Amiot and Nicolas Durand, *Algorithmes Génétiques*, mars 2005.

[Adl89] Samul Adler (ed.), *The Study of Orchestration*, Norton Company, New York, 1989.

[Ago98] Carlos Agon, OpenMusic *: Un langage visuel pour la composition musicale assistée par ordinateur*, Ph.D. thesis, Université Pierre et Marie Curie (Paris 6), Paris, France, 1998.

[AHA02] Carlos Agon, Karim Haddad, and Gérard Assayag, *Representation and Rendering of Rythmic Structures*, International Conference of Web Delivering of Music (Darmstadt, Allemagne), IEEE Computer Press, 2002.

[BAKC99] F. Ben Abdelaziz, S. Krichen, and J. Chaouachi, *Metaheuristics : Advances and Trends in Local Search Paradigms for Optimization*, ch. A Hybrid Heuristic for Multiobjective Knapsack Problems, pp. 205–212, Kluwer Academic Publishers, 1999.

[Bar85] Jean-Baptiste Barrière (ed.), *Le timbre, métaphore pour la composition*, Christian Bourgois, Paris, 1985.

[BBBK07] Alexander M. Bronstein, Michael M. Bronstein, Alfred M. Bruckstein, and Ron Kimmel, *Paretian Similarity for Partial Comparison of Non-rigid Objects*, SSVM, 2007, pp. 264–275.

[BBHL99] Guillaume Ballet, Riccardo Borghesi, Peter Hoffmann, and Fabien Lévy, *Studio Online 3.0 : An Internet "Killer Application" for Remote Access to IRCAM Sounds and Processing tools*, Actes des Journées d'informatique musicale (Paris), 1999.

[Bel01] Alan Belkin, *L'art orchestral*, 2001, `http://www.musique.umontreal.ca/personnel/belkin/bk.o.fr/of3.html`.

[Ber55] Hector Berlioz, *Traité d'instrumentation et d'orchestration*, Henri Lemoine (2e ed., Paris, 1855.

[Beu00] Anthony Beurivé, *Un logiciel de composition musicale combinant un modèle spectral, des structures hiérarchiques et des containtes*, Actes des Journées d'informatique musicale (JIM'2000) (Bordeaux), 2000, pp. 21–30.

[Bou00] Richard Boulanger, *The CSound Book*, MIT Press, 2000.

[BR05] Niels Bogaards and Axel Roebel, *An Interface for Analysis-Driven Sound Processing*, AES 119th Convention (New York, Etats-Unis), Octobre 2005.

[Cas58] Alfredo Casella, *Technique de l'orchestre contemporain*, Ricordi, Paris, 1958.

[CD01] Philippe Codognet and Daniel Diaz, *Yet Another Local Search Method for Constraint Solving*, AAAI Fall Symposium on Using Uncertainty within Computation, Cape Cod, 2001.

[CDT02] Philippe Codognet, Daniel Diaz, and Charlotte Truchet, *The Adaptive Search Method for Constraint Solving and its Application to musical CSPs*, Proceedings of the 1st International Workshop on Heuristics, 2002.

[Cer94] R. Cerf, *Une théorie asymptotique des algorithmes génétiques*, Ph.D. thesis, Université Montpellier II, France, 1994.

[CL94] Y.L Che and C.C. Liu, *Multiobjective VAR planning Using the Goal Attainment Method*, IEEE Proc. on Generation, Transmission and Distribution, mai 1994, pp. 227–232.

[CTA+06] Grégoire Carpentier, Damien Tardieu, Gérard Assayag, Xavier Rodet, and Emmanuel Saint-James, *Imitative and Generative Orchestrations Using Pre-Analyzed Sound Databases*, Proceedings of Sound and Music Computing Conference (SMC) (Marseille, France), mai 2006, pp. 115–122.

[CTA+07] ———, *An Evolutionary Approach to Computer-Aided Orchestration*, EvoMUSART (Valence, Espagne), vol. LNCS4448, Avril 2007, pp. 488–497.

[Deb00] Kalyanmoy Deb, *An Efficient Constraint Handling Method for Genetic Algorithms*, Computer Methods in Applied Mechanics and Engineering (2000), no. 186, 311–338.

[Del04] Olivier Delerue, *Spatialisation du son et programmation par contraintes : le système* MusicSpace, Ph.D. thesis, Université Pierre et Marie Curie (UPMC) – Paris 6, 2004.

[DJ98] R. Dorne and J.K.Hao, *A New Genetic Local-Search Algorithm for Graph Coloring*, Lecture Notes in Computer Science **LNCS 1498** (1998), 745–754.

[DJ02] Kalyanmoy Deb and Sachin Jain, *Running Performance Metrics for Evolutionary Multi-Objective Optimization*, Tech. Report 2002004, Kanpur Genetic Algorithm Laboratory (KanGAL), Indian Institue of Technology, Kanpur, PIN 208 016, Inde, 2002.

[DP91] P. Depalle and G. Poirrot, *SVP : A Modualr System for Analysis, Processing and Synthesis of Sound Signals*, Proceedings of the International Computer Music Conference (ICMC), 1991.

[DPAM00] K. Deb, A. Pratap, S. Agarwal, and T Meyarivan, *A Fast and Elistist Multi-Objective Genetic Algorihm : NSGA-II*, Tech. Report 200001, Kanpur Genetic Algorithm Laboratory (KanGAL), Indian Institue of Technology, Kanpur, PIN 208 016, Inde, 2000.

[Ebc88] Kemal Ebcioglu, *An Expert System for harmonizong Four-part Chorals*, Computer Music Journal **12** (1988), no. 3, 43–51.

[EG00] Matthias Ehrgott and Xavier Gandibleux, *A Survey and Annotated Bibliography of Multioblective Combinatorial Optimization*, OR Spektrum **22** (2000), 425–460.

[Ehr05] Matthias Ehrgott, *Multicriteria Optimization (2nd ed.)*, Springer, 2005.

[Eib01] A.E. Eiben, *Evolutionary Algorithms and Constraint Satisfaction : Definitions, Survey, Methodology, and Research Directions*, Theoretical Aspects of Evolutionary Computing, Springer-Verlag, 2001, pp. 13–58.

[ELT07] Semya Elaoud, Taicir Loukil, and Jacques Teghem, *The Pareto Fitness Genetic Algorithm : Test Function Study*, European Journal of Operational Research (2007), no. 177, 1703–1719.

[Eri75] Robert Erickson, *Sound structure in music*, University of California Press, 1975.

[EvdH96] A.E. Eiben and J.K. van der Hauw, *Graph Coloring with Adaptive Evolutionnary Algorithms*, Tech. Report TR-96-11, Leiden University, 1996.

[FF93] C.M. Fonseca and P.J. Flemming, *Genetic Algorithm for Multiobjective Otimization : Formulation, Discussion and Generalization*, Proceedings of the 5$^{\text{th}}$ International Conference on Genetic Algorithms (San Mateo, CA, USA) (Stephanie Forest, ed.), Morgan Kaufman, 1993, pp. 416–423.

[FF95] ———, *An Overview of Evolutionnary Algorithms in Multiobjective Optimization*, Journal of Evolutionary Computation **3** (1995), no. 1, 1–16.

[GHNO03] Masataka Goto, Hiroki Hashiguchi, Takuichi Nishimur, and Ryuichi Oka, *RWC Music Database : Music Genre Database and Musical Instrument Sound Database*, Proceedings of the 4$^{\text{th}}$ International Conference on Music Information Retrieval (ISMIR 2003), 2003, pp. 229–230.

[GL97] F. Glover and H. Laguna, *Tabu Search*, Kluwer Academic Publishers, 1997.

[GM00] Brian R. Glasberg and Brian C.J. Moore, *A Model for Prediction of Thresholds Loudness and Partial Loudness*, Hearing Research **47** (2000), 103–138.

[Gol89] D.E. Goldberg, *Genetic Algorithms in Search, optimization and Machine Learning*, Addison-Wesley, 1989.

[Got04] Masataka Goto, *Development of the RWC Music Database*, Proceedings of the 18$^{\text{th}}$ International Congress on Acoustics (ICA 2004), vol. 1, avril 2004, pp. 553–556.

[Gro05] C. Grosan, *Evolutionary optimization : Algorithms and applications*, Tech. report, Dept. Of Computer Science, Babes-Bolyai University of Cluj-Napoca, Roumanie, 2005.

[Gro06] C. Grosan, *Multiobjective Adaptive Representation Evolutionary Algorithm (MAREA) — A New Evolutionary Algorithm for Multiobjective Optimization*, Proceedings of 9$^{\text{th}}$ World on-line Conference on Soft Computing in Industrial Application, Applied Soft Computing Technologies : The Challenge of Complexity, Advances in Soft Computing (Allemagne), Springer-Verlag, 2006, pp. 113–121.

[HGH98] Jin-Kao Hao, Philippe Galinier, and Michel Habib, *Métaheuristiques pour l'optimisation combinatoire et l'affectation sous contraintes*, Journal of Heuristics (1998).

[HJ97] P.H. Hansen and A. Jaszkiewicz, *Evaluating Quality of Approximations to the Non-dominated Set*, Tech. Report IMM-REP-1998-7, Dept. Of Mathematical Modelling, Technical University of Denmark, 1997.

[HNG94] J. Horn, N. Nafpliotis, and D.E. Goldberg, *A Niched Pareto Genetic Algorithm for Multiobjective Optimization*, Proceedings of the First IEEE Conference on

Evolutionary Computation, IEEE World Congress on Computational Intelligence (Piscataway, New Jersey), IEEE Service Center, 1994, pp. 82–87.

[Hol75] J.H. Holland, *Adaptation in natural and Artificial Systems*, Ph.D. thesis, 1975.

[Hum05] Thomas A. Hummel, *Simulation of Human Voice Timbre by Orchestration of Acoustic Music Instruments*, Proceedings of International Computer Music Conference (ICMC), 2005.

[IK03] H. Ishibuchi and S. Kaige, *Effects of Repair Procedures on the Performance of EMO Algorithms for Multiobjective 0/1 Knapsack Problems*, Evolutionary Computation **4** (2003), 2254–2261.

[IM96] H. Ishibuchi and T. Murata, *Multi-Objective Genetic local Search Algorithm*, Proc. of 3rd IEEE International Conference on Evolutionary Computation, 1996, pp. 119–124.

[IY02] H. Ishibuchi and T. Yoshida, T. Murata, *Balance between Genetic Search and Local Search in Hybrid Evolutionary Multi-Criterion Optimization Algorithms (2002)*.

[Jas01] Andrzej Jaszkiewicz, *Comparison of Local Search-Based Metaheuristics on the Multiple Objective Knapsack Problem*, Foundations of Computing and Design Sciences **26** (2001), no. 1, 99–120.

[Jas02] ———, *Genetic Local Search for Multiple Objective Combinatorial Optimization*, European Journal of Operational Research **137** (2002), no. 1, 50–71.

[Jot97] Jean-Marc Jot, *Efficient Models for Reverberation and Distance Rendering in Computer Music and Virtual Audio Reality*, International Computer Music Conference (ICMC) (Thessaloniki, Greece), Septembre 1997, pp. 236–243.

[KC99] J.D. Knowles and D.W. Corne, *The Pareto Archived Evolution Strategy : A New Baseline Algorithm for Multiobjective Optimization*, Proceedings of the 1999 Congree on Evolutionary Computation (Piscataway, New Jersey), IEEE Service Center, 1999, pp. 98–105.

[KC00] ———, *A Comparison of Diverse Approaches to Memetic Multiobjective Combinatorial Optimisation*, Proceedings of the 2000 Genetic and Evolutionary Computation Conference Workshop Program (Las Vegas, Nevada, USA), 2000, pp. 103–108.

[KFTZ05] Jochua D. Knowles, Carlos M. Fonseca, Lothar Thiele, and Eckart Zitzler, *A Tutorial on the Performance Assessment of Stochastic Multiobjective Optimizers*, Evolutionary Multi-Criterion Optimization Conference (EMO), 2005.

[Koe43] Charles Koechlin, *Traité de l'Orchestration*, Max Eschig, Paris, 1943, 4 volumes.

[Lau93] M. Laurson, *PWConstraints*, 10imo Colloquio di Informatica Musicale (Associazione di Informatica Musicale, Milano, Italie) (G. Haus and I. Pighi, eds.), 1993, pp. 332–335.

[LD89] M. Laurson and J. Duthen, PatchWork*, a Graphic Language in PreForm*, Proceedings of the International Computer Music Conference (ICMC) (Ohio State University, Etats-Unis), 1989, pp. 172–175.

[LT06] Thibaut Lust and Jacques Teghem, *MEMOX : a Memetic Algorithm Scheme for Multiobjective Optimization*, Proceedings of the 7th International Conference on Multi-Objective Programming and Goal Programming, 2006.

[MGB97] Brian C.J. Moore, Brian R. Glasberg, and Thomas Baer, *A Model for Prediction of Thresholds, Loudness, and Partial Loudness*, Journal of Audio Engeneering Society (1997), 224–240.

[Mic96] Z. Michalewicz, *Denetic Algorithms + Data Structures + Evolution Programs (3^{rd} Ed.)*, Springer, berlin, 1996.

[Mie99] Kaisa Miettinen, *Nonlinear Multiobjective Optimization*, International Series in Operations Research and Management Science, vol. 12, Kluwer Academic Publishers, Dordrecht, 1999.

[MS98] Kim Mariott and Peter Stuckey, *Programming with Constraints : An Introducion*, MIT Press, 1998.

[MT90] S. Martello and P. Toth, *Knapsack Problems : Algorithms and Computer Implementations*, John Wiley & Sons, Chichester, 1990.

[MWD+95] S. McAdamcs, S. Winnsberg, S. Donnadieu, G. De Soete, and J. Krimphoff, *Perceptual Scaling of Synthesized Musical Timbres : Common Dimensions, Specificities, and Latent Subject Classes*, Psychological Research (1995), no. 58, 177–192.

[OFL97] Y. Orlarey, D. Fober, and S. Letz, *L'environnement de composition musicale* ELODY, Actes des Journées d'informatique musicale (JIM'97) (Lyon, France), 1997.

[OGAK05] M. Oltean, C. Grosan, A. Abraham, and M. Koppen, *Multiobjective Optimization Using Adaptive Pareto Archived Evolution Strategy*, Proceedings of the 5^{th} International Conference on Intelligent Systems Design and Applications (ISDA), 2005, pp. 558–563.

[Pee04] Geoffroy Peeters, *A Large Set of Audio Features for Sound Description (Similarity and Classification) in the CUIDADO Project*, Tech. report, IRCAM, 2004.

[Pis55] Walter Piston, *Orchestration*, Norton Company, New York, 1955.

[PMH00] Geoffroy Peeters, Stephen McAdams, and Perfecto Herrera, *Instrument Sound Description in the Context of MPEG-7*, International Computer Music Conference (ICMC) (Berlin, Allemagne), Septembre 2000, pp. 166–169.

[Pop91] S.T. Pope (ed.), *The Well-Tempered Object (Musical Application of Object-Oriented Technology)*, MIT Press, 1991.

[PR01] François Pachet and Pierre Roy, *Musical Harmonization with Constraints : A Survey*, Constraints Journal **6** (2001), no. 1, 7–19.

[Pse03] David Psenicka, *Sporch : An Algorithm for Orchestration Based on Spectral Analyses of Recorded Sounds*, Proceedings of International Computer Music Conference (ICMC), 2003.

[Puc91] Miller Puckette, *Combining Event and Signal Processing in the* MAX *Graphical Programming Environment*, Computer Music Journal **15** (1991), no. 3.

[RAQ+01] C. Rueda, G. Alvarez, L.O. Quesada, G. Tamura, Valencia F.D, J.F. Diaz, and G. Assayag, *Integrating Constraints and Concurrent Objects in Musical Applications : A Calculus ans its Visual Language*, Constraints **6** (2001), no. 1.

[RH05] François Rose and James Hetrick, *Spectral Analysis as a Ressource for Contemporary Orchestration Technique*, Proceedings of Conference on Interdisciplinary Musicology, 2005.

[Ris86] Jean-Clause Risset, *Son musical et perception auditive*, Pour la Science (1986), no. 109.

[RK64] Nicolaï Andre Rimski-Korsakov, *Principles of Orchestration, with Musical Examples Drawn from his own Works*, Maximilian Steinberg, New York, 1964.

[RLBA98] Camilo Rueda, Mikael Laurson, Georges Bloch, and Gérard Assayag, *Integrating Constraints Programming in Visual Musical Composition Languages*, Proceedings of the European Conference on Artificial Intelligence, 1998.

[Rod97] Xavier Rodet, *Musical Sound Signal Analyis/Synthesis : Sinusoidal+Residual and Elementary Waveform Models*, Proceedings of the IEEE Time-Frequency and Time-Scale Workshop (Coventry, Grande Bretagne), 1997.

[Sal79] M.E. Salukwadze, *Vector-Valued Optimization Problems in Control Theory*, Academic Press, 1979.

[Sch11] Arnold Schoenberg, *Harmonielehre*, Universal Edition, Vienne, 1911.

[Sch66] Pierre Schaeffer, *Traité des objets musicaux*, Seuil, Paris, 1966.

[Sch68] J.F. Schouten, *The Perception of Timbre*, Reports of the 6$^{\text{th}}$ Internation Cangress on Acoustics (Tokyo, Japon), no. GP-6-2, 1968, pp. 35–44, 90.

[SD94] N. Srinivas and K. Deb, *Multiobjective Optimization Using Nondominated Sorting in Genetic Algorithms*, Evolutionary Computation **2** (1994), no. 3, 221–248.

[SF97] Eleanor Selfridge-Field (ed.), *Beyond MIDI : The Handbook of Musical Codes*, MIT Press, Cambridge, 1997.

[Sol02] Christine Solnon, *Ants Can Solve Constraint Satisfaction Problems*, IEEE Transactions on Evolutionary Computation **6** (2002), 347–357.

[Ste86] R.E. Steuer, *Multiple Critera Optimization - Theory, Computation and Application*, Wiley, New York, 1986.

[Ste90] G.L. Steele (ed.), *Common Lisp, the Language (2nd Ed.)*, Digital Press, 1990.

[SW00] D. Schwartz and M. Wright, *Extensions and Applications of the SDIF Sound Description Interchange Format*, Proceedings of the International Computer Music Conference (Berlin, Allemagne), 2000.

[Tal99] E.G. Talbi, *Métaheuristiques pour l'optimisation combinatoire multi-objectif : Etat de l'art*, Tech. report, C.N.E.T. (France Telecom), octobre 1999.

[TR07] Damien Tardieu and Xavier Rodet, *An Instrument Timbre Model For Computer Aided Orchestration*, WASPAA (New Paltz, NY, USA), Octobre 2007.

[Tru04] Charlotte Truchet, *Contraintes, recherche locale et composition assistée par ordinateur*, Ph.D. thesis, Univerité Paris 7 Denis Diderot, Paris, 2004.

[Tsa93] E.P. Tsang, *Fondations of Constraint Satisfaction*, Academic Press, 1993.

[UT94] E.L. Ulungu and J. Teghem, *Multi-Objective Combinatorial Optimization : A Survey*, Foundations of Computing and Decision Sciences **20** (1994), no. 2, 149–165.

[VHD03] Michel Vasquez, Djamal Habet, and Audrey Dupont, *Neighborhood Design by Consistency Checking*, Actes du 5$^{\text{e}}$ congrès de la société française de recherche opérationnelle et d'aide à la décision (ROADEF'2003) (Avignon, France), 2003.

[VSL] *Vienna Symphonic Library*, `http://www.vsl.co.at`.

[vV99] D. A. van Veldhuizen, *Multiobjective Evolutionary Algorithms : Classifications, Analyses and New Innovations*, Ph.D. thesis, Department of Electrical and Computer Engineering, Graduate School of Engineering, Air Force Institute of Technology, 1999.

[WFM03] Matthew Wright, Adrian Freed, and Ali Momeni, *OpenSound Control : State of the Art 2003*, Proceedings of the 2003 Conference on New Interfaces for Musical Expression (NIME-03) (Montréal, Canada), 2003.

[WHBH06] L. While, P. Hingston, L. Barone, and S. Huband, *A Faster Algorithm for Calculating Hypervolume*, IEEE Transactions on Evolutionary Computation **10** (2006), no. 1, 29–38.

[Whi93] Darrell Whitley, *A Genetic Algorithm Tutorial*, Tech. Report CS-93-103, Colorado State university, mars 1993.

[YP02] Collette Y. and Siarry P., *Optimisation multiobjectif*, Eyrolles, 2002.

[Zit99] E. Zitzler, *Evolutionary Algorithms for Multiobjective Optimization : Methods and Applications*, Ph.D. thesis, Swiss Federal Institute of Technology (ETH), Zurich, 1999.

[ZLT02] E. Zitzler, M. Laumanns, and L. Thiele, *SPEA2 : Improving the Strength Pareto Evolutionary Algorithm for Multiobjective Optimization*, Evolutionary Methods for Design, Optimisation and Control with Application to Industrial Problems (K.C. Giannakoglou et al., eds.), International Center for Numerical Methods in Engineering (CIMNE), 2002, pp. 95–100.

[ZT98a] E. Zitzler and L. Thiele, *An Evolutionary Algorithm for Multiobjective Optimization : The Strength Pareto Approach*, Tech. Report 43, Computer Engineering and Communication Networks Lab (TIK) — Swiss Federal Institute of Technology (ETH), Gloriastrasse 35, CH-8092 Zurich, Switzerland, 1998.

[ZT98b] ———, *Multiobjective Optimization Using Evolutionary Algorithms — A Comparative Case Study*, Lecture Notes in Computer Science **1498** (1998), 292–302.

[ZT99] ———, *Multiobjective Evolutionary Algorithms : A Comparative Case Study and the Strength Pareto Approach*, IEEE Transactions on Evolutionary Computation **3** (1999), no. 4, 257–271.

[ZTL+03] E. Zitzler, L. Thiele, M. Laumanns, C. M. Foneseca, and V. Grunert da Fonseca, *Performance Assessment of Multiobjective Optimizers : An Analysis and Review*, IEEE Transactions on Evolutionary Computation **7** (2003), no. 2, 117–132.

[Zwi61] E. Zwicker, *Subdivision of the Audible Frequency Range into Critical Bands*, Journal of the Acoustical Society of America (1961), no. 33.