ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

UNIVERSITÉ DU QUÉBEC

MÉMOIRE PRÉSENTÉ À

L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

COMME EXIGENCE PARTIELLE

À L'OBTENTION DE LA

MAÎTRISE EN GÉNIE

M.Ing.

PAR

PASCAL CÔTÉ

OPTIMISATION MULTI-OBJECTIVE DES PROBLÈMES COMBINATOIRES : APPLICATION À LA GÉNÉRATION DES HORAIRES D'EXAMENS FINAUX

MONTRÉAL, LE 1 NOVEMBRE 2004

CE MÉMOIRE A ÉTÉ ÉVALUÉ

PAR UN JURY COMPOSÉ DE :

M. Tony Wong, directeur de recherche
Département de génie de la production automatisée à l'École de technologie supérieure

M. Robert Sabourin, codirecteur
Département de génie de la production automatisée à l'École de technologie supérieure

M. Pascal Bigras, président du jury
Département de génie de la production automatisée à l'École de technologie supérieure

M. Thien-My Dao, examinateur externe
Département de génie mécanique à l'École de technologie supérieure

IL A FAIT L'OBJET D'UNE PRÉSENTATION DEVANT JURY ET PUBLIC

LE 24 SEPTEMBRE 2004

À L'ÉCOLE DE TECHNOLOGIE SUPÉRIEURE

OPTIMISATION MULTI-OBJECTIVE DES PROBLÈMES COMBINATOIRES : APPLICATION À LA GÉNÉRATION DES HORAIRES D'EXAMENS FINAUX

Pascal Côté

SOMMAIRE

Les problèmes d'optimisation combinatoire discrète (POCD) sont des problèmes très difficiles à résoudre. La nature discrète des variables forme un espace non dérivable qui rendent inutiles les techniques basées sur le gradient. Le problème de la production des horaires est représentatif d'une famille de POCD. Il renferme un ensemble d'objectifs conflictuels, un ensemble de contraintes non linéaires et un nombre de combinaisons potentielles très élevé. De plus, un certain nombre d'institutions académiques produisent encore des horaires d'une manière manuelle ou semi-automatique. L'automatisation peut donc éliminer les aspects déplaisants de cette tâche. Ce travail porte sur l'optimisation combinatoire par algorithmes évolutifs et, plus précisément, sur les problèmes de création d'horaires des examens finaux (PCHE). Dans un premier temps, les POCD mono-critère et multi-critères sont décrits d'une manière formelle afin d'en établir les principales caractéristiques. Les méthodes qui ont été proposées pour la résolution d'un PCHE, tels que le recuit simulé, la fouille Tabou et les algorithmes génétiques ont fait l'objet d'une revue de la littérature. Afin de faire un lien avec les méthodes de résolutions multi-critères, il sera prouvé qu'un PCHE est lui aussi un POCD multi-critères. Jusqu'à maintenant, ce sont principalement les modèles mono-critères qui sont utilisés lors de la résolution de ce type de problème. Ainsi, l'étude qui a été entreprise dans ce travail s'est concentrée sur les différentes techniques d'optimisation multi-critères envisagables pour la résolution d'un PCHE. Parmi les algorithmes évolutifs multi-critères les plus populaires, ceux de la famille NSGA de Deb et du SPEA de Zitzler ont été susceptibles d'obtenir de bons résultats. Ces techniques, vues en détail dans ce travail, ont été implantées sur un problème de création des horaires d'examens finaux. Suite à une première série d'expérimentations, les algorithmes NSGA-II et SPEA-II se sont montrés inappropriés pour la résolution d'un PCHE à cause de la gestion des contraintes et à la diversité des solutions. L'ensemble des problèmes rencontrés lors de l'utilisation de ces algorithmes a permis la conception d'une nouvelle approche hybride. Cet algorithme évolutif hybride possède une structure semblable à celle de l'algorithme SPEA-II de Zitzler. La différence majeure est le remplacement de l'opérateur de croisement par deux fouilles Tabou. La première fouille est appliquée afin de réduire les violations de contraintes alors que la deuxième fouille Tabou permet l'optimisation de l'étalement temporel des examens des élèves. Les résultats obtenus avec l'ensemble des bases de données publiques ont démontré qu'en moyenne l'algorithme développé fonctionne très bien et présente un bon potentiel pour la réalisation de travaux futurs.

APPLICATION OF A HYBRID MULTI-OBJECTIVE EVOLUTIONARY ALGORITHM TO THE FINAL EXAM PROXIMITY PROBLEM

Pascal Côté

ABSTRACT

Final exam proximity problem is a hard combinatorial optimization problem. The decision variables formed a discretized search space that does not possess gradient or Hessian operators. The general exam proximity problem contains conflicting objectives, nonlinear side constraints and a very complex solution-space landscape. For the academic institutions that produce timetables in a semi-manually or manually way, the complete automation of the procedure is, obviously, a great improvement. This work presents a hybrid multi-objective evolutionary algorithm to solve the exam proximity problem. The problem has been investigated by many researchers. However, most of the techniques are based on a single-objective approach. Since the final exam proximity problem contains conflicting objectives, a multi-objective approach is more appropriated. The first part of this work presents the mathematical definitions and a formal description of single and multi-objective combinatorial optimization problems. We reviewed the previous works done on the exam proximity problem by presenting the standard solution techniques such as Tabu Search, Simulated Annealing and Genetic Algorithms. The content of the second part concerns the application of a multi-objective evolutionary algorithm (MOEA). After analyzing some of the most popular MOEAs, we applied two of them, the Elitist Non-Dominated Sorting Genetic Algorithm (NSGA-II) and the Strength Pareto Evolutionary Algorithm (SPEA-II) to the uncapacitated exam proximity problem. These two algorithms failed on our multi-objective model due to the constraint handling and the diversity of the solutions on the Pareto front. The weakness of these algorithms gave us the opportunity to develop a new algorithm. Thus, a Hybrid Multi-Objective Evolutionary Algorithm is used to tackle the uncapacitated and the capacitated exam proximity problem. In this hybridization, local search operators are used instead of the traditional genetic recombination operators. One of the local search operators is designed to repair unfeasible timetables produced by the initialization procedure and the mutation operator. The other search operator implements a simplified Variable Neighborhood Descent meta-heuristic and its role is to improve the proximity cost. The resulting non dominated timetables are compared to other optimization methods using several public domain datasets. Without special fine-tuning, the hybrid algorithm was able to produce timetables ranking first and second in most of the datasets.

REMERCIEMENTS

Je tiens en premier lieu à remercier mon directeur, monsieur Tony Wong. Durant la dernière année de mon baccalauréat, j'ai eu la chance d'effectuer un stage sous sa supervision et depuis ce temps je n'ai cessé d'apprécier ses conseils que je trouve toujours importants. En plus d'être mon directeur, le professeur Wong m'a enseigné lors de mes études de premier cycle et je peux affirmer sans l'ombre d'un doute, qu'en plus d'être un chercheur et un directeur émérite, le professeur Wong est un excellent pédagogue. Je tiens à le remercier tout spécialement pour avoir cru en mes compétences et de m'avoir encouragé à poursuivre mes études au niveau doctoral. Je remercie l'école de technologie supérieure (ÉTS) ainsi que le fond Nature et Technologie du Québec (NATEQ) pour leur financement. Je tiens également à remercier mon co-directeur Robert Sabourin pour sa participation à ce travail de recherche. Je veux aussi remercier mes parents pour leur aide constante tout au long de mes études. Je souhaite remercier ma conjointe pour ses judicieux conseils lors de la rédaction de ce document. Tout au long de mes études de deuxième cycle j'ai aussi eu la chance de travailler sur un projet connexe qui consistait à implanter un logiciel pour la création de l'horaire des examens finaux de l'ÉTS. Je tiens donc à remercier le personel du Décanat de la gestion des ressources pour leur aide et leur appuis lors de l'implantation de ce logiciel appelé PlaneX. Je tiens tout spécialement à remercier madame Manon Côté pour toutes ses suggestions qui ont fait du logiciel PlaneX un outils très convivial. Pour terminer je tiens à remercier le Doyen de la gestion des ressources, monsieur Paul Gely pour m'avoir fait confiance dans le grand projet PlaneX.

TABLE DES MATIÈRES

TABLE DES TABLEAUX

TABLE DES FIGURES

LISTE DES ABRÉVIATIONS ET DES SIGLES

AEMC	Algorithme évolutif multi-critères
AEMH	Algorithme évolutif multi-critères hybride
AG	Algorithme génétique
LD	Largest Degree
LWD	Largest Weighted Degree
MOEA	Multi-Objective Optimization Problem
NSGA	Non-Dominated Sorting Algorithm
NSGA-II	Elitism Non-Dominated Sorting Algorithm
PCHE	Problème de création d'horaires des examens
POAC	Problème d'optimisation avec contraintes
POCD	Problème d'optimisation combinatoire discret
POSC	Problème d'optimisation sans contrainte
POMAC	Problème d'optimisation multi-critères avec contraintes
POMC	Problème d'optimisation multi-critères
POMSC	Problème d'optimisation multi-critères sans contrainte
PSC	Problème de satisfaction de contraintes
RO	Radom Ordering
RS	Recuit simulé
SD	Saturation Degree
SPEA	Strength Pareto Evolutionary Algorithm
SPEA-II	Strength Pareto Evolutionary Algorithm v.2
FT	Fouille Tabou

CHAPITRE 1

HORAIRES D'EXAMENS ET OPTIMISATION COMBINATOIRE

1.1 Introduction

Le premier chapitre de ce mémoire consiste essentiellement à établir un cadre de travail servant à introduire l'ensemble des méthodes qui seront proposées pour une application concernant l'ordonnancement des horaires d'examens. Dans un premier temps, il sera question des problèmes d'optimisation combinatoire et des horaires d'examens. Les relations qui existent entre les problèmes d'optimisation combinatoire et les horaires d'examens seront montrées en établissant quelques définitions et en élaborant les caractéristiques des horaires d'examens ainsi que des problèmes d'optimisation combinatoire. Afin de bien comprendre la nature des problèmes de création d'horaires des examens (PCHE), les caractéristiques des problèmes d'optimisation mono-critère et multi-crièteres seront aussi présentées. Par la suite plusieurs méthodes de solutions qui ont été appliquées aux PCHE seront vues.

1.2 Problématique des horaires d'examens

Pour un élève, un horaire d'examens représente la distribution des périodes où ce dernier doit se présenter afin de subir une évaluation dans un ou plusieurs cours dans lequel il est inscrit. Pour un gestionnaire responsable de l'ordonnancement, un horaire d'examens représente l'assignation des examens dans les différentes périodes prévues à cette fin. Cette représentation sera utlisée dans le cadre de ce travail de recherche. Un horaire d'examens peut être considéré comme *réalisable* ou *non réalisable*. Un horaire dit *réalisable* doit obligatoirement respecter l'ensemble des contraintes *fortes*. Il existe un certain nombre de contraintes *fortes*. La première, essentielle à tout PCHE, est le respect des conflits d'horaire. Un conflit d'horaire est présent lorsqu'un élève se voit assigner deux examens

à la même période. Il est aussi possible de considérer d'autres contraintes fortes comme le repect de la capacité des locaux et la préassignation des examens. Les préassignations stipulent que certains examens doivent avoir lieu durant certaines périodes spécifiques. Ainsi, dès que l'une ou l'autre de ces contraintes est enfreinte, l'horaire est considéré comme *non réalisable*.

Ce n'est pas toutes les institutions d'enseignement qui ont recours à la création d'un horaire d'examens finaux. Certaines utilisent la dernière semaine de cours pour la tenue des examens. Cette décision évite d'avoir à procéder à la recherche d'un horaire. Par contre, pour les élèves, cette décision implique qu'ils n'auront qu'une seule semaine attribuée pour la tenue de leurs examens. La création d'un horaire d'examen différent de celui des cours permet donc d'ajouter des périodes suplémentaires et ainsi de donner plus de temps d'étude aux élèves. Évidemment, cette tâche doit se faire de façon tout aussi rigoureuse que la création des horaires de cours. Un horaire d'examens qui est bien conçu permettra aux élèves de mieux se préparer et permettra aux administrateurs de créer un horaire qui respecte la capacité d'accueil des locaux ainsi que les différentes contraintes reliées au personel enseignant.

Lorsque la résolution d'un PCHE ne nécessite que des contraintes fortes, le problème posé est un problème de satisfaction de contraintes. Par contre, un PCHE peut aussi être considéré comme un problème d'optimisation combinatoire. Cette situation se présente lorsque le PCHE est soumis à certaines contraintes dites *faibles*. Ces contraintes peuvent, entre autres, être reliées aux proximités. Les proximités dans un horaire d'examens correspondent à l'étalement temporel des examens d'un élève. Dans la plupart des applications, les proximités représentent le critère d'optimisation. Nous pouvons donc considérer le PCHE comme un problème d'optimisation combinatoire. En effet, un horaire est créé en choisissant une assignation des périodes disponibles à l'ensemble des examens. Cette assignation doit permettre la minimisation d'un ou plusieurs critères tout en satisfaisant les contraintes imposées. Le choix d'une telle assignation n'est pas trivial puisque la taille de

l'ensemble des assignations est très grande. Même si cet ensemble est réduit par l'application des contraintes, le choix d'une assignation demeure très difficile. Pour cette raison, il est nécessaire d'apporter une méthode de solution qui est systématique et rigoureuse.

1.3 Caractéristiques des problèmes d'optimisation combinatoire

Les définitions qui suivent serviront à exposer de façon concise les caractéristiques élémentaires des problèmes d'optimisation combinatoire. Ces définitions sont tirées, en partie, des travaux de Gottlieb [15] effectués sur les algorithmes évolutifs servant à la résolution de problèmes d'optimisation combinatoire. Pour faciliter la lecture, les symboles suivants seront utilisés :

a. H_c : Espace de décision réalisable

b. D_i : Domaine associé à la variable x_i

c. f : Fonction d'objectif

d. c_i : Fonction de contrainte

e. $\mathcal{N}^+$: Ensemble des nombres entiers positifs

f. $\mathcal{R}^+$: Ensemble des nombres réels positifs

g. S : Espace de décision

h. U_c : Espace de décision non réalisable

i. $\setminus$: Opérateur de soustraction d'ensembles

Définition 1.1 *Un **domaine** est un ensemble fini ou infiniment dénombrable d'éléments* $D = \{d_1, d_2, \ldots, d_m\}$. *Pour les besoins de ce projet de recherche,* $d_i \in \mathcal{N}^+$.

Définition 1.2 *Soit un ensemble fini de variables de décision* $X = \{x_1, x_2, \ldots, x_m\}$ *ainsi que les domaines* D_i *associés à chaque variable de* X, *un **espace de décision** est noté par*

le produit cartésien de tous les domaines $S = D_1 \times D_2 \times \cdots \times D_m$. *Un élément* $a \in S$ *est appelé une* ***solution.***

Définition 1.3 *Soit* S *un espace de décision et une* ***fonction d'objectif*** $f : S \rightarrow \mathcal{R}^+$. *Une solution* $a \in S$ *est optimale par rapport à* f *si* $f(a) > f(b), \forall\ b \in S$ *dans le cas d'une maximisation et* $f(a) < f(b), \forall\ b \in S$ *dans le cas d'une minimisation.*

Définition 1.4 *Soit* S *un espace de décision et un ensemble* ***fonction de contrainte*** $c_i : S \rightarrow \mathcal{R}^+$, $i = 1, 2, \ldots, K$. *Un* ***espace de décision réalisable*** *est noté par* $H_c = a \in S$ *tel que* $c_i(a)$ *op 0,* $\forall\ i$ *(op* $\triangleq >$ *pour le cas d'une contrainte d'inégalité, op* $\triangleq =$ *pour le cas d'une contrainte d'égalité). Chaque élément de* H_c *est appelé* ***solution réalisable.*** *Un espace de solutions non réalisables est donné par* $U_c = S \backslash H_c$ *et chaque élément de* U_c *est une* ***solution non réalisable.***

Dans ce travail nous considérons deux types de contraintes. Les contraintes d'égalités $c(x) = 0$ et les contraintes d'inégalités $c(x) > 0$. Ces contraintes peuvent être *fortes*, dans le cas où toutes les solutions réalisables doivent respecter les contraintes. Les contraintes peuvent être *faibles* si elles peuvent être relâchées pour accepter une solution comme "réalisable". Notons que les contraintes d'égalité fortes sont les plus restrictives puisqu'elles découpent l'espace de décision de façon très draconienne. Par exemple, dans un problème 2-D, l'espace de décision sera restreint par une droite.

1.4 Structure des problèmes d'optimisation combinatoire

Cette section présente, à l'aide des définitions 1.1 à 1.3, les structures particulières des problèmes d'optimisation combinatoire. Le premier type de problèmes abordé est le problème d'optimisation sans contrainte (POSC). Dans un POSC (problème 1.1), l'optimisation peut s'effectuer en tout point dans l'espace de décision puisqu'il y a absence de contrainte.

Problème 1.1 *Un problème d'optimisation sans contrainte (POSC) est défini par S et f. Il a la forme :*

$$\begin{aligned} \min / \max \quad & f(a)\,; \\ & a \in S\,; \\ & a_{min} \leq a \leq a_{max}\,. \end{aligned}$$

Le problème est résolu par l'obtention de $a \in S$ où a est la solution optimale par rapport à f et a_{min}, a_{max} définissent l'espace de décision.

Parfois un problème n'a pas de critère à optimiser. Les problèmes qui n'ont pas de critère d'optimisation, mais qui possèdent un ensemble de contraintes sont des problèmes de satisfaction des contraintes (PSC). Ils sont souvent utilisés pour vérifier si le problème est soluble. Si $H_c = \{\emptyset\}$, le problème n'a pas de solution et est insoluble. On peut aussi utiliser les PSC (problème 1.2) pour déterminer la région des solutions réalisables dans un espace de décision. Fondamentalement, les PSC sont des problèmes de recherche.

Problème 1.2 *Un problème de satisfaction des contraintes (PSC) est défini par S et $K > 0$. Il a la forme :*

$$\begin{aligned} \textit{satisfaire} \quad & c_i(a), \qquad\qquad i = 1, 2, \ldots, K\,; \\ & a \in S\,; \\ & a_{min} \leq a \leq a_{max}. \end{aligned}$$

Le problème 1.2 est résolu par l'obtention de $a \in H_c$ si $H_c \neq \{\emptyset\}$.

L'ajout au problème 1.2 d'un critère d'optimisation crée un autre type de problème appelé problème d'optimisation combinatoire avec contraintes (POAC). Ce dernier est le cas général des problèmes d'optimisation mono-critère. Les POAC (problème 1.3) sont très importants puisque la majorité des problèmes d'optimisation le sont.

Problème 1.3 *Un problème d'optimisation avec contraintes (POAC) est défini par S, f et $K > 0$ et a la forme :*

$$\begin{array}{lll} \min / \max & f(a), & \\ s.a. & c_i(a) \; op \; 0, & i = 1, 2, \ldots, K\,; \\ & a \in S, & \\ & a_{min} \leq a \leq a_{max}. & \end{array}$$

où $op \in \{\geqslant, =\}$ selon le type de contraintes. Le problème est résolu par l'obtention de $a \in H_c$ où a est la solution optimale par rapport à f. De plus, les K contraintes sont respectées.

1.5 Optimisation multi-critères

Les problèmes d'optimisation multi-critères (POMC) sont une généralisation des problèmes d'optimisation. Dans les POMC, il existe $M > 1$ fonctions d'objectif et deux espaces sont considérés, soit l'espace de désicion et l'espace d'objectif. Dans un POMC, toute solution est à la fois représentée dans l'espace de décision (définitions 1.1 et 1.2) et l'espace d'objectif (définitions 1.5). Les définitions suivantes ainsi que les symboles associés, serviront à formuler d'une manière précise les structures et les caractéristiques des POMC :

a. $\succcurlyeq$: Opérateur de dominance

b. Λ_j : Domaine de la fonction d'objectif j

c. f : Fonction d'objectif

d. $\mathcal{F}$: Ensemble de Pareto

e. O : Espace d'objectif

f. $\mathbf{z}$: Solution représentée dans l'espace d'objectif

Définition 1.5 *Soit un espace de décision S, un ensemble de fonctions d'objectif f_j, $j = 1, \ldots, M$, $M > 1$, le* ***domaine d'une fonction d'objectif*** *est noté par $\Lambda_j = f_j(\mathbf{a}), \forall\ \mathbf{a} \in S$. Un* ***espace d'objectif*** *est noté par $O = \Lambda_1 \times \Lambda_2 \times \ldots \Lambda_M$. Une solution dans l'espace d'objectif est notée $\mathbf{z} \in O$.*

Définition 1.6 *Soit deux solutions $\mathbf{z}_1, \mathbf{z}_2$ définies dans l'espace d'objectif O, le* ***principe de dominance*** *[9] représenté par l'opérateur $\succcurlyeq$ tel que $\mathbf{z}_1 \succcurlyeq \mathbf{z}_2$, signifie que la solution $\mathbf{z}_1$* domine *la solution $\mathbf{z}_2$ si, et seulement si, dans l'espace de décision, il existe a_1 et a_2, les solutions associées à $\mathbf{z}_1$ et $\mathbf{z}_2$ qui satisfont les deux conditions suivantes :*

1. $f_i(a_1) \leq f_i(a_2)$, $i \in 1,2, \ldots, M$
2. $\exists\ i$ *tel que* $f_i(a_1) < f_i(a_2)$

De plus, $\mathbf{z}_1 \not\succcurlyeq \mathbf{z}_2$ signifie que la solution $\mathbf{z}_1$ ne domine pas la solution $\mathbf{z}_2$. À noter que l'opérateur de dominance est non symétrique, c'est-à-dire que si $\mathbf{z}_1 \not\succcurlyeq \mathbf{z}_2$ cela n'implique pas nécessairement que $\mathbf{z}_2 \succcurlyeq \mathbf{z}_1$.

Définition 1.7 *Soit un espace d'objectif O, un* ***ensemble Pareto*** *est un ensemble de solutions non dominées possiblement infiniment dénombrable défini par :*

$$\mathcal{F} = \left\{ \left\{ \mathbf{z}^{(1)}, \mathbf{z}^{(2)}, \ldots, \mathbf{z}^{(l)} \right\} \in O \mid \mathbf{z}^{(n)} \not\succcurlyeq \mathbf{z}^{(m)} \wedge \mathbf{z}^{(m)} \not\succcurlyeq \mathbf{z}^{(n)},\ m, n = 1, 2, \ldots, l \right\} \quad (1.1)$$

avec $\mathcal{F} \subset O$. L'ensemble ***optimal de Pareto*** *est alors :*

$$\mathcal{F}_1 = \{ \mathbf{z} \mid \mathbf{z} \succcurlyeq \mathbf{y},\ \forall\ \mathbf{z} \in \mathcal{F}_1 \wedge \mathbf{y} \in O \backslash \mathcal{F}_1 \} \quad (1.2)$$

Tout comme les problèmes mono-critère, les problèmes multi-critères peuvent associer ou non des contraintes. Dans le cas d'un problème d'optimisation multi-critères sans contrainte (problème 1.4), la recherche s'effectuera en tout point dans l'espace de décision. L'ensemble optimal de Pareto formera une frontiere sur les régions limites de l'es-

pace d'objectif. La figure 1 montre la répartition des éléments de l'ensemble $\mathcal{F}_1$ dans l'espace d'objectif pour un POMSC hypothétique 2-D.

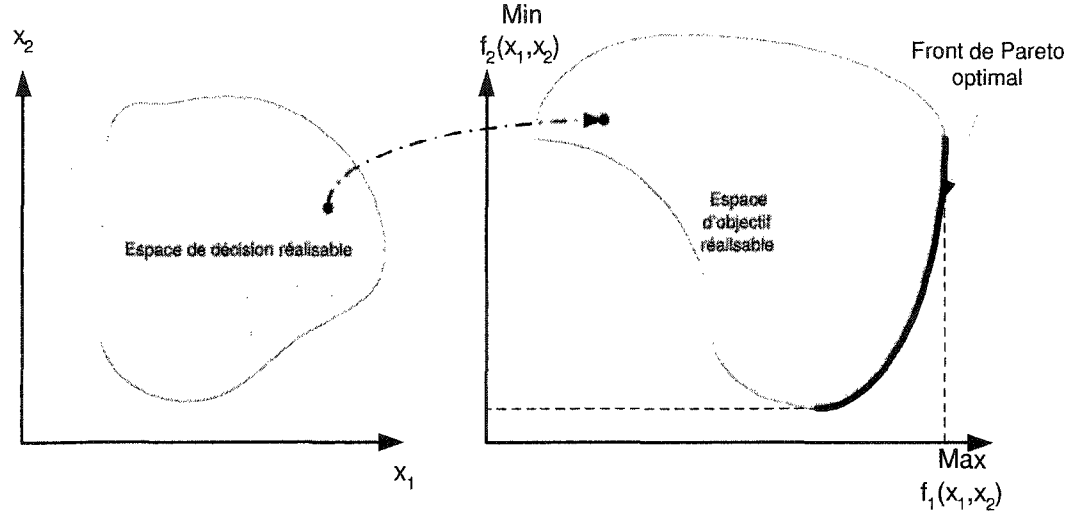

Figure 1 Illustration de l'ensemble optimal de Pareto, problème sans contrainte

Problème 1.4 *Soit S un espace de décision, O un espace d'objectif et f_i, $i = 1, 2, \ldots, M$, $M > 1$, un ensemble de fonctions d'objectif, un problème d'optimisation multi-critères sans contrainte (POMSC) est défini par* $\mathbf{a} \in S$, $\mathbf{a} = [a_1, a_2, \ldots a_n]^T$, *l'espace d'objectif O et l'ensemble des fonctions d'objectif f_i. Il a la forme :*

$$\begin{aligned} \min / \max \quad & f_i(\mathbf{a}), && i = 1,2, \ldots, M; \\ & a_{\min} \leqslant a_j \leqslant a_{\max}, && j = 1,2, \ldots n. \end{aligned}$$

Le problème 1.4 est résolu par l'obtention de l'ensemble optimal de Pareto $\mathcal{F}_1 \subset O$ où $a_{\min}$ et $a_{\max}$ définissent l'espace de décision S et par conséquant l'espace d'objectif O (voir figure 1).

Un problème d'optimisation multi-critères avec contraintes (problème 1.5) est la forme la plus générale des problèmes d'optimisation. La figure 2 montre la répartition des élé-

ments de l'ensemble $\mathcal{F}_1$ dans l'espace d'objectif pour un POMAC hypothétique 2-D avec contrainte.

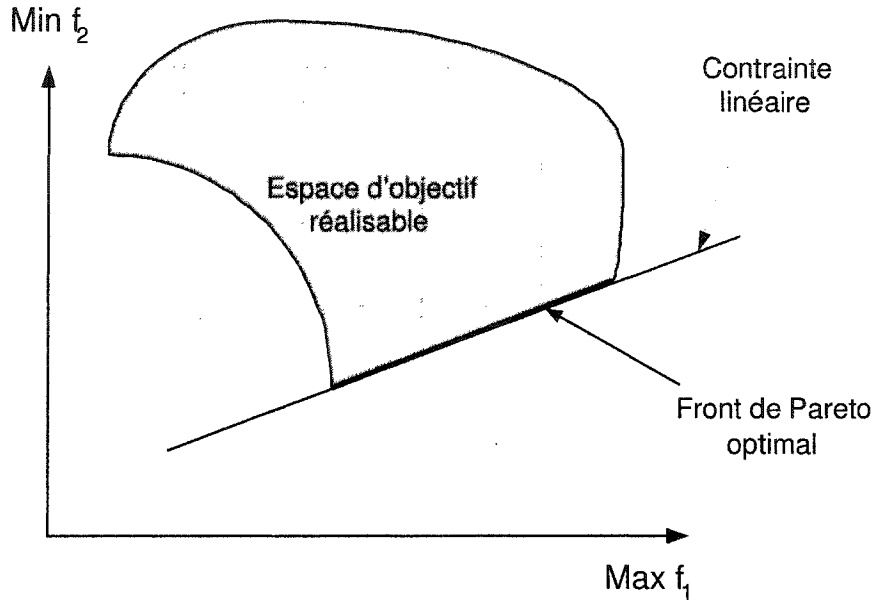

Figure 2 Illustration de l'ensemble optimal de Pareto, problème avec contraintes

Problème 1.5 *Soit S un espace de décision, O un espace d'objectif, f_i, $i = 1, 2, \ldots, M$, $M > 1$, un ensemble de fonctions d'objectif et c_j, $j = 1, 2, \ldots, K$ un ensemble de contraintes, un problème d'optimisation multi-critères avec contrainte (POMAC) est défini par* $\mathbf{a} \in S$, $\mathbf{a} = [a_1, a_2, \ldots a_n]^T$, *l'espace d'objectif O, l'ensemble des fonctions d'objectif f_i et $K > 0$. Il a la forme :*

$$
\begin{array}{lll}
\min / \max & f_i(\mathbf{a}), & i = 1,2, \ldots, M; \\
s.a. & c_k(\mathbf{a}) \; op \; 0, & k = 1,2,\ldots,K; \\
 & a_{\min} \leqslant a_j \leqslant a_{\max}, & j = 1,2, \ldots, n.
\end{array}
$$

où $op \in \{\geqslant, =\}$ *selon le type de contrainte. Le problème 1.5 est résolu par l'obtention de l'ensemble optimal de Pareto* $\mathcal{F}_1 \subset O$ *dont les solutions satisfont les K contraintes.*

1.6 Méthodes pour la résolution des problèmes d'optimisation

Cette partie présente différentes techniques de résolution appliquées aux problèmes d'optimisation combinatoire. Plus précisément, les techniques de fouille par escalade, fouille Tabou, recuit simulé et algorithmes génétiques seront résumées. La fouille par escalade, la fouille Tabou et le recuit simulé sont des techniques basées sur le parcours de l'espace de décision. Ces techniques utilisent deux stratégies pour réaliser le parcours. Ces stratégies sont i) la perturbation ; ii) l'acceptation. La perturbation est la production d'une nouvelle solution $a' \in \mathcal{S}$ (réalisable ou non réalisable) à partir de la solution courante a. Elle est accomplie par la définition d'un voisinage qui comprend l'ensemble des solutions voisines $V(a) \subset \mathcal{S}$. L'objectif de la perturbation est de permettre à la fouille d'exploiter l'espace de décision qui est "près" de la solution courante.

L'acceptation, quant à elle, est une stratégie permettant à une solution $a' \in V(a)$ de devenir la solution courante. Dans le cas de la fouille par escalade, la stratégie d'acceptation est absente. La solution courante a est celle qui satisfait la relation $f(a) < f(a')$ avec $a' \in V(a)$. Autrement dit, la fouille par escalade est de nature gloutonne et elle peut atteindre rapidement un optimum local. Pour la fouille Tabou et le recuit simulé, la solution courante peut être une solution voisine de qualité inférieure. Cette stratégie d'acceptation donne un parcours plus élargi et par le fait même une meilleure exploration de l'espace de décision. Elle est généralement plus efficace puisqu'il est désormais possible de se déplacer en dehors d'un optimium local en acceptant des solutions voisines de qualité inférieure. La stratégie d'accpetation est réalisée, chez la fouille Tabou, par la mémorisation des déplacements effectués. Pour le recuit simulé, elle est réalisée par une acceptation probabiliste des solutions voisines.

Les algorithmes génétiques, quant à eux, utilisent un tout autre paradigme. Ils s'appuient sur un ensemble de solutions candidates et la fouille est réalisée par le croisement et la mutation des solutions. Puisque ces algorithmes considèrent un ensemble de solutions, ils ont la capacité d'exploiter et d'explorer en parallèle plusieurs régions de l'espace de décision.

1.6.1 Fouille par escalade non déterministe

L'algorithme de fouille par escalade (algorithme 1, page 16) est un algorithme non déterministe très simple à implanter. À partir d'une solution $a \in S$, une nouvelle solution sera choisie dans un sous-ensemble de solutions voisines $V(a) \subset S$. Un déplacement peut être effectué d'une solution a vers une solution $a' \in V(a)$ si ce dernier apporte une amélioration à la fonction d'objectif. Les solutions obtenues suite à l'application de cette fouille se retrouvent la plupart du temps dans des optimums locaux. Comme mentionné plus haut, ceci vient du fait que la fouille par escalade est de nature gloutonne, c'est-à-dire qu'elle parcourt l'espace de décision sans stratégie d'acceptation.

1.6.2 Fouille Tabou

La fouille Tabou (algorithme 2, page 17) a été développée par Glover [26]. L'essentiel de l'algorithme est basé sur la méthode de la fouille par escalade. Par contre, pour éviter que l'algorithme s'engouffre dans un optimum local, le choix de la nouvelle solution s'effectue en fonction d'une liste $\mathcal{L}$ de déplacements qui ont déjà été effectués (liste Tabou). Cette liste est mise à jour tout au long de la fouille et est constituée des caractéristiques des déplacements antérieurs. Par contre, un déplacement qui est dans la liste Tabou peut être effectué s'il augmente la qualité de la solution. Cette comparaison est effectuée à l'aide du critère d'aspiration $\mathcal{A}$. Il peut se présenter des situations où aucun déplacement améliorant la valeur de la fonction d'objectif n'est possible à cause de la liste Tabou. Dans ce contexte, le choix de la solution voisine s'effectuera malgré une baisse de qualité.

La fouille Tabou comporte quelques faiblesses. La liste Tabou est un élément difficile à gérer. La mémoire des ordinateurs étant limitée, la taille de la liste doit aussi être bornée. Même pour un problème de faible dimension, l'espace requis peut s'avérer important. De plus, comme la liste doit être parcourue à chaque itération, la taille de la liste aura un impact majeur sur le temps de traitement.

1.6.3 Recuit simulé

L'algorithme du recuit simulé, comme son nom l'indique, est une analogie aux méthodes de recuit effectuées sur les aciers. Pour développer l'algorithme du recuit simulé (algorithme 3, page 18), Kirkpatrik et son équipe se sont inspirés de l'algorithme Métropolis. Cet algorithme utilise des modèles statistiques qui régissent l'agencement des molécules lors du refroidissement d'un acier chauffé à haute température. Le but des recuits est de trouver la température initiale ainsi que la fonction de refroidissement qui optimiseront l'agencement des molécules de façon à augmenter les propriétés mécaniques des aciers [12].

Appliqué dans le domaine de la résolution de problèmes d'optimisation combinatoire, le recuit simulé est semblable à la méthode de fouille par escalade, à la différence que le choix d'une solution peut être valide malgré une baisse de qualité. L'acceptation peut s'effectuer si la relation suivante est vraie :

$$\delta \leq e^{-\Delta f/T_k} \tag{1.3}$$

où δ est un nombre aléatoire compris entre [0,1]. La température T_k diminue selon un cycle de refroidissement qui peut être défini par la fonction :

$$T_k = g(T_0) = T_0/ln(k) \tag{1.4}$$

Un désavantage important quant à l'utilisation de cette technique est l'ajustement des paramètres. En plus d'avoir à déternimer quel cycle de froidissement s'applique le mieux, ces cycles requièrent l'utilisation de plusieurs paramètres. Comme le cycle de refroidissment est un élément très important dans cette fouille, l'ajustement des paramètres doit donc être fait avec attention.

1.6.4 Types de voisinage utilisés

Le principe de voisinage est très important dans la fouille locale. Concernant les techniques présentées à la section 1.6, seul les algorithmes génétiques qui évoluent à l'aide d'un ensemble de solutions n'utilisent pas le voisinage proprement dit. Certaines techniques de voisinage sont fort utiles pour l'optimisation des horaires d'examens. Nombre de travaux ont entrepris la résolution du problème en plusieurs étapes, comme par exemple en trouvant tout d'abord un horaire sans conflit, puis en optimisant les proximités. Il est primordial que l'optimisation des proximités se fasse en respectant les conflits d'horaire pour ne pas détruire le travail de la première étape. Les types de voisinage présentés dans les paragraphes qui suivent permettent d'obéir à cette règle.

1.6.4.1 Voisinage par échange et déplacement d'examens

Le type de voisinage le plus simple est le voisinage par déplacement d'un examen. Ce type de voisinage s'effectue en déplaçant un examen dans une autre période disponible. Généralement, même pour des problèmes dont la taille est restreinte, ce type de voisinage ne permet pas une convergence rapide puisque le déplacement dans l'espace des solutions est faible. De plus, les préassignations ainsi que les conflits d'horaire peuvent aussi rendre le déplacement d'un seul examen impossible. Une version similaire à ce voisinage est l'échange d'examens. Ce voisinage consiste à interchanger les périodes de deux examens. Cette échange doit toujours se faire en respectant la contrainte des conflits d'horaire. Cette méthode permet une plus grande perturbation que la méthode par déplacement d'examen

et en échangant deux examens de périodes, la capacité des locaux est beaucoup plus facile à satisfaire.

1.6.4.2 Voisinage par échange de périodes

Un autre type de voisinage utilisé dans le domaine de l'ordonnancement des examens est celui par échange de périodes. Cette méthode consiste à choisir au hasard deux périodes $\{ t_0 , t_1 \} \in \mathcal{T}, t_0 \neq t_1$ puis à déplacer l'ensemble des examens assignés à la période t_0 à la période t_1 et vice-versa. Bien que cette méthode permet la permutation des périodes tout en respectant les conflits d'horaire, le voisinage par échange de périodes comporte deux faiblesses. Tout d'abord le déplacement d'un nombre important d'examens ne permet pas une recherche très raffinée. Les chances d'obtenir la solution optimale d'un minimum locale ou globale sont donc réduites. La deuxième faiblesse se présente lorsque certains examens sont sujets à des préassignations. Dans ce contexte, il est possible que le déplacement ne soit pas valide, car il peut exister un ou plusieurs examens ne pouvant être assignés à la période t'. Cette contrainte diminue considérablement l'efficacité de ce voisinage.

1.6.4.3 Voisinage par chaîne de Kempe

Une autre méthode intéressante est le voisinage par chaîne de Kempe [33]. Une chaîne de Kempe est représentée par un sous-graphe non orienté $D \subset G$ où G représente le graphe non orienté de l'ensemble des examens. Les examens du sous-graphe D sont assignés soit à la période t_0 ou à la période t_1, $t_0 \neq t_1$ (figure 3). Une chaîne de Kempe se construit donc à l'aide du triplet (e, t_0, t_1) où e est un examen préalablement sélectionné, t_0 la période où cet examen est assigné et t_1 une période où cet examen pourra être assigné. La chaîne de Kempe D est obtenue en utilisant la relation suivante :

$$\begin{aligned} V(D) &= \{V_{t_0}\} \cup \{V_{t_1}\} \\ E(D) &= \left\{(u,w) : (u,w) \in E(G),\ u \in V_{t_i} \wedge w \in V_{t_{(i+1) \bmod 2}}\right\} \end{aligned} \tag{1.5}$$

Dans l'équation (1.5), $E(G)$ représente l'ensemble des arcs et V_i l'ensemble des examens

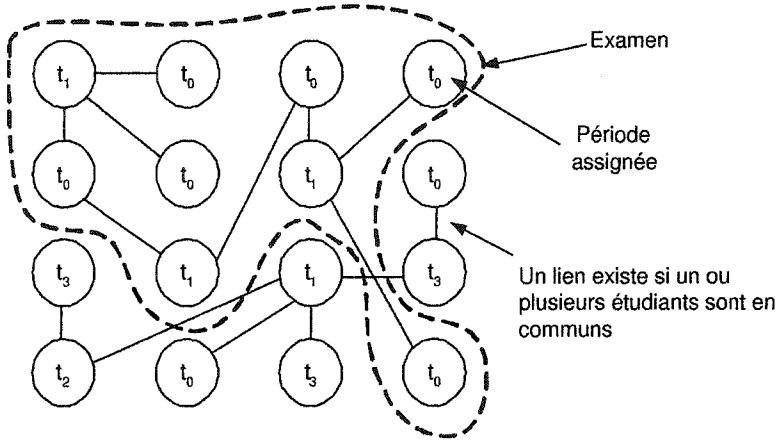

Figure 3 Construction d'une chaîne de Kempe

assignés à la périodes i. Une chaîne de Kempe est le sous-graphe dans lequel l'examen e est atteignable et qui n'admet que deux périodes t_0 et t_1. Avec ce sous-ensemble d'examens aucun conflit supplémentaire ne sera créé suite à l'échange des périodes t_0 et t_1 pour tous les examens impliqués dans la chaîne de Kempe.

ALGORITHME 1

Fouille par escalade, version non déterministe

Données :
a_1 : Solution initiale.
a_i : Solution à l'itération i.

début
  Création de la solution initiale a_1
  pour *chaque itération* i **faire**
    Créer le sous-ensemble $V(a_i)$
    répéter
      Choix aléatoire d'une solution $a'_i \in V(a_i)$
      Δf = f(a'_i) - f(a_i)
      si $\Delta f \geq 0$ *// Dans le cas d'une maximisation* **alors**
        $a_{i+1} = a'_i$
    jusqu'à *ce que la solution soit acceptée*
fin

ALGORITHME 2

Fouille Tabou

Données :
a_1 : Solution initiale.
a_i : Solution à l'itération i.
$\mathcal{L}$: Liste Tabou.

début
  Création de la solution initiale a_1
  pour *chaque itération* i **faire**
    Créer le sous-ensemble $V(a_i)$
    répéter
      Choix aléatoire d'une solution $a_i' \in V(a_i)$
      Δf = f(a_i') - f(a_i)
      si $a_i' \notin \mathcal{L}$ ***ET*** $\Delta f \geq 0$ *// Dans le cas d'une maximisation* **alors**
        $a_{i+1} = a_i'$
      sinon si $f(a')$-$f(a) \geq \mathcal{A}(f(a'))$ **alors**
        $a_{i+1} = a_i'$
    jusqu'à *ce que la solution soit acceptée*
    Mettre à jour la liste $\mathcal{L}$
fin

ALGORITHME 3

Recuit Simulé

Données :
a_1 : Solution initiale.
a_i : Solution à l'itération i.
T_i : Température à l'itération i.
g : Fonction de refroidissement.

début
  Création de la solution initiale a_1
  Initialisation de la température T_1
  pour *chaque itération* i **faire**
    Créer le sous-ensemble $V(a_i)$
    répéter
      Choix aléatoire d'une solution $a_i' \in V(a_i)$
      Δf = f(a_i') - f(a_i)
      si $\Delta f \geq 0$ *//Dans le cas d'une maximisation* **alors**
        $a_{i+1} = a_i'$
      sinon si rand*(0,1)* $\leq e^{-\Delta f/T_i}$ **alors**
        $a_{i+1} = a_i'$
    jusqu'à *ce que la solution soit acceptée*
    Mettre à jour la température $T_{i+1} = g(T_1)$
fin

1.6.5 Les algorithmes génétiques

Depuis les dix dernières années, les algorithmes génétiques (AG) n'ont cessé d'augmenter en popularité. Cette méthode (algorithme 4, page 23), basée sur les principes empiriques de l'évolution, est très utile pour la résolution de problèmes d'optimisation. Les AG peuvent être vus comme une série de transformations appliquées à un sous-ensemble de solutions. Les solutions sont aussi appellées chromosomes. Le croisement représente l'échange des éléments, que l'on appelle gènes, constituants l'ensemble des $n > 1$ solutions, tandis que la mutation représente l'altération des éléments constituant une seule solution. Ainsi, l'exploitation de l'espace de décision est réalisée par le croisement des solutions sélectionnées, alors que la mutation des solutions individuelles permet l'exploration de l'espace de décision. Un nouvel ensemble de solutions candidates est produit par l'application successive de la mutation et du croisement. Comme l'algorithme s'opère sur un ensemble de solutions, il est en mesure d'obtenir plus d'une solutions réalisables en un seul lancement de l'algorithme.

1.6.5.1 Évaluation des chromosomes

L'évaluation des chromosomes consiste à attribuer une valeur d'adaptation à chaque individu. Cette valeur permettra d'établir une comparaison lors de la sélection des individus. Un exemple d'implantation de l'opérateur d'assignation de la valeur d'adaptation est de borner la valeur entre [0,1]. Dans le cas d'une minimisation, si $0 \leq f(a) \leq \infty$, l'application peut être réalisée par la relation :

$$\mathrm{F}(a) = \frac{1}{1 + f(a)} \tag{1.6}$$

1.6.5.2 Sélection

Les deux principales implantations Λ de l'opérateur de sélection S sont le tournoi et la roulette. Dans le cas du tournoi binaire, les solutions sont choisies de façon aléatoire parmi une sous-population qui est construite en effectuant des compétitions entre les solutions de la population. Goldberg et Deb ont montré que cet opérateur converge plus rapidement que tout autre opérateur utilisé pour la sélection [9]. Pour l'utilisation de la sélection par roulette, la solution est choisie, par exemple, en fonction d'une probabilité p_S qui varie entre 0 et $\sum_{j=0}^{N} \mathrm{F}(a_j)$ où N est la taille de la population et $\mathrm{F}(a_j)$ est la valeur d'adaptation de la solution a_j calculée selon la relation (1.6).

1.6.5.3 Croisement

Outre les croisements réels, les deux principales implantations Γ de l'opérateur de croisement X sont le croisement uniforme et par points de coupure. Dans le cas des croisements uniformes, chaque valeur associée aux gènes de l'enfant est constituée par la valeur d'un des deux gènes des parents. Le choix de la valeur de chaque gène est fait selon une probabilité p_C comprise entre [0,1] (figure 4). Pour les croisements par points de coupure (figure 5), les chromosomes sont découpés de façon aléatoire en un ou plusieurs points afin de reconstituer le nouveau chromosome. Il existe un bon nombre de méthodes de croisement pour les modèles en nombre réels [16] [23]. Par contre, comme ce travail porte essentiellement sur des modèles discrets, seuls les techniques de croisement et de mutation applicables à ces modèles seront résumées.

1.6.5.4 Mutation

L'implantation Π de l'opérateur de mutation M consiste à modifier un gène du chromosome de façon aléatoire selon une probabilité p_M comprise entre [0,1]. La mutation favorisera une plus grande diversité à l'intérieur de la population. Pour des modèles discrets,

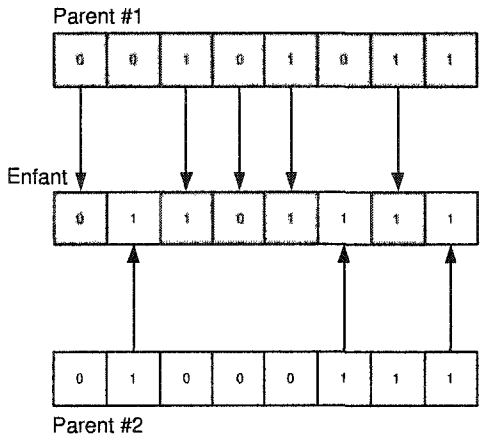

Figure 4 Croisement uniforme

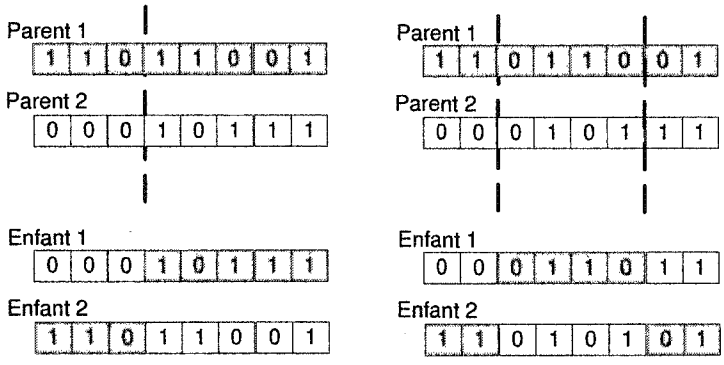

Figure 5 Croisement par points de coupure

la mutation s'effectue d'une manière semblable en changeant le domaine des valeurs attribuées aux gènes.

1.6.5.5 Insertion et élitisme

Il existe, en général, deux façons de procéder à l'insertion des individus produits lors de l'application des opérateurs génétiques de croisement et de mutation. Tout d'abord, l'implantation de l'opérateur d'insertion pourra être totale lorsque la population est remplacée par de nouveaux chromosomes. Une deuxième méthode, en régime permanent

("*steady-state*"), consiste à insérer immédiatement dans la population les chromosomes qui viennent d'être croisés et mutés. Pour l'élitisme, la principale implantation est celle où les meilleurs individus sont conservés à chaque itération. Le nombre d'élites est défini par $\epsilon\%$ du nombre total d'individus dans la population. Une autre forme d'élitisme est celle où, lors de la création de deux enfants par deux parents, une comparaison est faite entre ces quatre individus et les deux meilleurs sont conservés.

1.6.5.6 Choix des paramètres

Beaucoup d'efforts ont été déployés dans le domaine des AG afin de démontrer de façon analytique ou empirique les paramètres optimaux. Les travaux de Eiben et *al.* [21] font état d'une multitude de techniques pour l'obtention de ces paramètres. Entre autres, le taux de mutation P_m proposé est défini par $1/L$ où L est la longueur du chromosome, c'est-à-dire le nombre d'éléments constituants un chromosome. Une version dynamique équivalente peut aussi être implantée. Dans l'équation (1.7), le taux de mutation décroît en fonction des générations ce qui permet une recherche plus vaste au départ et plus raffinée à la fin. Pour le taux de croisement, les valeurs les plus souvent préconisées sont $p_c \in [0.6, 0.95]$. Eiben et *al.* [21] précisent que le taux de croisement ne doit pas être inférieur à 0.6.

$$p_m = \left(2 + \frac{L-2}{T}t\right)^{-1} \tag{1.7}$$

ALGORITHME 4

Algorithme Génétique

Données :
p_c : Probabilité de croisement.
p_m : Probabilité de mutation.
p_s : Probabilité de sélection.
E : Opérateur d'élitisme.
F : Opérateur d'assignation de la valeur d'adaptation.
I : Opérateur d'insertion.
M : Opérateur de mutation.
S : Opérateur de sélection.
X : Opérateur de croisement.
X : Population de l'algorithme génétique.
ϵ : Proportion d'élites conservés.
Ψ : Type d'implantation de l'opérateur d'insertion.
Π : Type d'implantation de l'opérateur de mutation.
Γ : Type d'implantation de l'opérateur de croisement.
Λ : Type d'implantation de l'opérateur de sélection.
Φ : Type d'implantation de l'opérateur d'assignation de la valeur d'adaptation.
Θ : Type d'implantation de l'opérateur d'élitisme.

début
  Initialisation de la population X_1
  pour *chaque itération* i **faire**
    Évaluation des chromosomes - $X_i = \mathrm{F}_{\Phi}(X_i)$
    Sélection - $X_i = \mathrm{S}_{\Lambda/p_s}(X_i)$
    Croisement - $X_i = \mathrm{X}_{\Gamma/p_c}(X_i)$
    Mutation - $X_i = \mathrm{M}_{\Pi/p_m}(X_i)$
    Élitisme - $X_i = \mathrm{E}_{\Theta/\epsilon}(X_i)$
    Insertion - $X_{i+1} = \mathrm{I}_{\Psi}(X_i)$
fin

1.7 Méthodes d'initialisation utilisées

Pour la majorité des algorithmes, le choix d'une ou plusieurs solutions initiales est primordial. Un choix judicieux permettra un meilleur positionnement de la fouille lors de son démarrage. Cette section présentera quelques techniques d'initialisation de solutions qui sont utilisées dans le domaine de la création des horaires d'examens.

Dans la plupart des cas, les techniques d'initialisation sont utilisées afin de créer une solution de départ réalisable, c'est-à-dire sans conflit d'horaire. Les quatre méthodes proposées reposent toutes sur le même principe. L'assignation des périodes se fait de façon séquentielle en sélectionnant les examens l'un après l'autre selon différents critères qui peuvent être :

a. *Largest Degree* (LD)

   Assigner les examens dans les périodes en commençant par les examens qui sont en relation avec le plus d'examens. Ou bien en termes de coloration de graphe, colorier les noeuds en commençant par ceux reliés par le plus grand nombre d'arcs.

b. *Saturation Degree* (SD)

   Communément appelée *D-Satur*, commencer par les examens ayant le choix de périodes possibles le plus restreint.

c. *Largest Weighted Degree* (LWD)

   Assigner les examens dans les périodes en commençant par les examens qui ont le plus d'élèves en conflits. Cette méthode est semblable au LD à la seule différence que la pondération des arcs est prise en considération.

d. *Random Ordering* (RO)

   Choix aléatoire.

D'après les expérimentations de Carter [29], les méthodes par *Largest degree* et *Saturation degree* donnent en moyenne les meilleurs résultats pour l'initialisation d'horaires d'examens.

1.8 Conclusion

Le problème de création des horaires d'examens est un problème d'optimisation combinatoire très difficile à résoudre. Les modèles utilisés se regroupent principalement parmi les problèmes de satisfaction de contraintes qui n'utilisent que des contraintes fortes et les problèmes d'optimisation combinatoire utilisant des contraintes fortes et un critère d'optimisation basé sur les contraintes faibles. Ces modèles demandent tous deux des méthodes de solutions rigoureuses. Parmi les méthodes utilisées pour la résolution des problèmes d'optimisation combinatoire on note la fouille en escalade, la fouille Tabou, le recuit simulé et les algorithmes génétiques. Un survol des principales caractéristiques de ces méthodes de solutions a donc été fait. Ces quatre techniques ont toutes été implantées pour la résolution des PCHE. Ces implantations seront présentées dans le prochain chapitre avec la revue de la littérature.

CHAPITRE 2

MODÈLES DU PROBLÈME ET REVUE DE LA LITTÉRATURE

2.1 Introduction

Depuis plus d'une quarantaine d'années le problème de création des horaires d'examens finaux suscite beaucoup d'intérêts chez certains chercheurs et a fait l'objet de nombreuses publications. La revue de la littérature présentée dans cette section relate les principaux travaux effectués sur l'ordonnancement des horaires d'examens depuis les six dernières années. Mais avant d'entreprendre la revue de la littérature, un nombre de modèles mathématiques consernant les PCHE seront étudiés. Cette formulation mathématique permettra de bien comprendre les différents modèles qui ont été proposés dans la littérature. On compte, entre autre, quatre problèmes standards, soit les modèles P1, P2, P3 et P4. Ces modèles servent à évaluer les performances des différents algorithmes. Ils seront tous présentés en détail dans ce chapitre. Pour terminer, voici la liste des symboles qui seront utilisés dans ce chapitre :

a. τ_{ij} : Quantité binaire désignant si l'élève i est inscrit à l'examen j

b. ϵ_{jk} : Quantité binaire désignant si l'examen j est assigné à la période k

c. η_{ij} : Quantité représentant le nombre d'élèves en commun entre les examens i et j

d. ζ_{jl} : Quantité binaire désignant si l'examen j est assigné au local l

e. α_{jk} : Quantité binaire désignant si l'examen j doit être préassigné à la période k

f. Cp_m : Capacité maximale d'accueil du local m

g. $\mathcal{E}$: Ensemble d'examens à assigner

h. $\mathcal{S}$: Ensemble d'élèves

i. $\mathcal{T}$: Ensemble de périodes disponibles pour les examens

j. $\| \star \|$: Cardinalité d'un ensemble

2.2 Problème de coloration des graphes

La forme la plus simple d'un PCHE est équivalente à celle de la coloration des graphes [27]. Ce problème (problème 2.1) consiste à colorier les noeuds d'un graphe G de façon à ce que deux noeuds reliés par un arc ne soient jamais de la même couleur. L'objectif de ce problème est la minimisation du nombre de couleurs. Dans le contexte des horaires, les noeuds de G représentent les examens alors que les couleurs représentent les périodes consacrées à la tenue des examens.

Problème 2.1 *Le modèle du problème de coloration des graphes se définit par :*

$$\begin{aligned} \min \quad & f = \|\mathcal{K}\| ; \\ s.a. \quad & \sum_{k \in \mathcal{N}} x_{ik} = 1 ; \\ & x_{ik} + x_{jk} = 1, \quad \forall\, i,j \in \mathcal{Q} \text{ tel que } E_{ij} \text{ existe}, \forall\, k \in \mathcal{N}. \end{aligned}$$

où $x_{ik} \in \{0,1\}$ tel que $x_{ik} = 1$ si le neoud i est de couleur k sinon $x_{ik} = 0$, $\mathcal{K}$ représentant l'ensemble des couleurs et $\mathcal{Q}$ l'ensemble des noeuds.

La première contrainte du problème 2.1 indique qu'un noeud doit être assigné d'une seule et unique couleur alors que la seconde contrainte indique qu'il est impossible que deux noeuds reliés par un arc soient de la même couleur. Par analogie, en posant les examens comme des noeuds et les couleurs comme des périodes, chaque examen doit être assigné à une seule période. De plus, deux examens qui ont des élèves en commun sont reliés par un arc. Ces derniers ne doivent pas être assignés à une même période. Plus formellement, ce modèle simpliste est représenté par les définitons suivantes :

Définition 2.1 *Soit $\mathcal{E} = \{e_1, e_2, \ldots, e_N\}$ un ensemble fini d'examens, et $\mathcal{S} = \{s_1, s_2, \ldots, s_M\}$ un ensemble fini d'élèves, $\tau_{ij} = 1$ si l'élève $i \in \mathcal{S}$ est inscrit à l'examen $j \in \mathcal{E}$, sinon $\tau_{ij} = 0$.*

Définition 2.2 *Soit* $\mathcal{E} = \{e_1, e_2, \ldots, e_N\}$ *un ensemble fini d'examens, et* $\mathcal{S} = \{s_1, s_2, \ldots, s_M\}$ *un ensemble fini d'élèves, le graphe non orienté et non pondéré* $G(E, V)$ *est défini par* V_i, $i \in \mathcal{E}$, *qui est l'ensemble des examens et* E_{km} *existe si* $\exists\, j \in \mathcal{S}$ *tel que* $\tau_{jk}+\tau_{jm}>1$.

Le PCHE établi par la définition 2.2 reste incomplet car plusieurs contraintes telles que la capacité des locaux ou la disponibilité des professeurs ne sont pas considérées. Certaines modifications devront être effectuées afin de rendre conforme le modèle aux exigences d'un PCHE plus concret.

2.3 Objectifs et contraintes

La présentation concise de l'ensemble des objectifs et des contraintes que l'on retrouve dans un PCHE nécessite quelques définitions essentielles. En plus des variables de décision énoncées dans les définitions 2.1 et 2.2, nous devons dorénavant tenir compte des préassignations, de la capacité des locaux et du nombre d'élèves inscrits aux examens.

Définition 2.3 *Soit* $\mathcal{T} = \{1,2, \ldots, T\}$, $\mathcal{T} \subseteq \mathbb{N}^+$, *l'ensemble des périodes et* $\mathcal{E} = \{e_1, e_2, \ldots, e_N\}$ *un ensemble fini d'examens,* $\epsilon_{jk} = 1$ *si l'examen* j *est assigné à la période* k, *sinon* $\epsilon_{jk} = 0$.

Définition 2.4 *Soit* $\mathcal{T} = \{1,2, \ldots, T\}$, $\mathcal{T} \subseteq \mathbb{N}^+$ *l'ensemble des périodes et* $\mathcal{E} = \{e_1, e_2, \ldots, e_N\}$ *un ensemble fini d'examens,* $\alpha_{jk} = 1$ *si l'examen* j *doit être préassigné à la période* k, *sinon* $\alpha_{jk} = 0$.

Définition 2.5 *Soit* $\mathcal{E} = \{e_1, e_2, \ldots, e_N\}$ *un ensemble fini d'examens et* $\mathcal{L} = \{l_1, l_2, \ldots, l_L\}$ *un ensemble fini de locaux ayant une capacité maximum* Cp_l, $\zeta_{jl} = 1$ *si l'examen* j *est assigné au local* l, *sinon* $\zeta_{jl} = 0$.

Définition 2.6 *Soit $\mathcal{E} = \{e_1, e_2, \ldots, e_N\}$ un ensemble fini d'examens et $\mathcal{S} = \{s_1, s_2, \ldots, s_M\}$ un ensemble fini d'élèves, le graphe non orienté et pondéré $G(E, V)$ est défini par V_i, $i \in \mathcal{E}$ et par E_{km} un arc joignant les examens k et m. E_{km} existe si $\exists\, i \in \mathcal{S}$ tel que $\tau_{ik}+\tau_{im}>1$ et sera pondéré par $\eta_{km} = \sum_{i \in \mathcal{S}} \tau_{ik} + \tau_{im}$ qui représente le nombre d'élèves en commun entre les examens k et m.*

En s'appuyant sur ces définitions, il est possible d'expliciter les différents objectifs et contraintes de la littérature. Plus précisémment, les objectifs reliés à la minimisation du nombre de périodes des horaires, l'optimisation de l'étalement temporel, les contraintes regroupant les conflits d'horaire, la capacité d'accueil des salles et enfin la préassignation des examens seront exposés.

2.3.1 Conflits d'horaire (f_1 ,c_1)

Les conflits d'horaire dans un PCHE sont présents lorsqu'un élève est assigné à deux examens durant la même période. En ce qui concerne le traitement de ces conflits, deux approches sont possibles. Premièrement, les conflits peuvent faire partie de la fonction d'objectif (équation 2.1) et être pondérés par un poids plus important. Deuxièmement, il est possible de les considérer comme une contrainte (équation 2.2), ce qui implique que toutes les solutions avec conflits devront être rejetées.

$$min \quad f_1 = \frac{1}{2} \sum_{k=1}^{\|\mathcal{T}\|} \sum_{i=1}^{\|\mathcal{E}\|} \sum_{j=1}^{\|\mathcal{E}\|} \eta_{ij} \epsilon_{ik} \epsilon_{jk} \tag{2.1}$$

$$c_1 : \quad \sum_{k=1}^{\|\mathcal{T}\|} \sum_{i=1}^{\|\mathcal{E}\|} \sum_{j=1}^{\|\mathcal{E}\|} \eta_{ij} \epsilon_{ik} \epsilon_{jk} = 0 \tag{2.2}$$

Évidemment, considérer les conflits comme des contraintes est beaucoup plus logique, mais dans la plupart des cas le respect de la contrainte demande un plus grand effort de recherche.

2.3.2 Étalement temporel des examens (f_2 à f_7)

Dans bien des PCHE, l'étalement temporel représente la qualité des horaires produits. Du point de vue des élèves, plus l'étalement temporel entre deux examens successifs est long, plus ils auront du temps à consacrer à leurs études. En posant n comme étant le nombre de périodes par jour et que les périodes sont numérotées de façon contiguë, la modélisation de l'ensemble des objectifs est définie par les équations suivantes :

$$min \quad f_2 = \frac{1}{2} \sum_{k=1}^{\|\mathcal{T}\|-1} \sum_{i=1}^{\|\mathcal{E}\|} \sum_{j=1}^{\|\mathcal{E}\|} \eta_{ij} \epsilon_{ik} \epsilon_{jk+1}, \quad \forall\, k \; tel \; que \; k \mod n \neq 0 \tag{2.3}$$

$$min \quad f_N = \frac{1}{2} \sum_{k=1}^{\|\mathcal{T}\|-(N-1)} \sum_{i=1}^{\|\mathcal{E}\|} \sum_{j=1}^{\|\mathcal{E}\|} \eta_{ij} \epsilon_{ik} \epsilon_{jk+(N-1)}, \quad N = 3...6 \tag{2.4}$$

$$min \quad f_7 = \frac{1}{2} \sum_{k=1}^{\|\mathcal{T}\|-1} \sum_{i=1}^{\|\mathcal{E}\|} \sum_{j=1}^{\|\mathcal{E}\|} \eta_{ij} \epsilon_{ik} \epsilon_{jk+1}, \quad \forall\, k \; tel \; que \; k \mod n = 0 \tag{2.5}$$

Parmi les objectifs ci-dessus, l'objectif f_2 représente le nombre d'élèves ayant deux examens consécutifs dans la même journée . Les fonctions d'objectifs f_N ($N = 3 \ldots 6$) indiquent le nombre d'élèves ayant de une à quatre périodes libres entre deux examens. Il est possible d'ajouter l'objectif f_7 qui représente les élèves ayant un examen le soir et le lendemain matin lorsque l'on ne considère pas la période nocturne comme une période libre normale. L'approche souvent préconisée par les chercheurs est de créer une seule fonction d'objectif en concaténant et en pondérant certaines des fonctions d'objectifs présentées ci-haut. Le choix des fonctions utilisées dépend généralement des exigences des institutions.

2.3.3 Capacité des locaux (f_8, c_2)

Dans un PCHE l'assignation des locaux devient un problème connexe. Dans la plupart des cas, l'opération sera effectuée à postériori. Par contre, il est important de tenir compte de la capacité maximale des locaux lors de la recherche d'un horaire.

$$c_2 : \quad \sum_{i=1}^{\|\mathcal{S}\|} \sum_{j=1}^{\|\mathcal{E}\|} \tau_{ij} \epsilon_{jk} \quad \leq \quad \sum_{m=1}^{\|\mathcal{L}\|} Cp_m, \quad \forall k \in \mathcal{T} \tag{2.6}$$

Comme le montre l'équation (2.6), cette contrainte impose que le nombre total d'élèves par période ne dépasse pas la capacité totale des locaux. Par contre, lorsque le nombre d'élèves approche la capacité maximale des locaux, le post-traitement peut s'avérer difficile, voire même impossible. Il est toujours possible d'inclure l'assignation des locaux à même le modèle du PCHE. L'objectif est de minimiser la perte d'espace tout en respectant la capacité des locaux :

$$min \quad f_8 = \sum_{j=1}^{\|\mathcal{E}\|} \sum_{m=1}^{\|\mathcal{L}\|} \epsilon_{jk} \zeta_{jm} (Cp_m - \sum_{i=1}^{\|\mathcal{S}\|} \tau_{ij}), \quad \forall k \in \mathcal{T} \tag{2.7}$$

Dans l'équation 2.7, le terme à l'intérieur des parenthèses définit la différence entre la capacité maximale du local m et le nombre d'élèves inscrits à l'examen j. Cette valeur est calculée si, et seulement si $\epsilon_{jk} = 1$ et $\zeta_{jm} = 1$, donc si l'examen j est assigné au local m.

2.3.4 Préassignations des examens (c_3)

Il arrive souvent pour des raisons administratives que certains examens doivent être donnés à une période bien spécifique ou dans une période faisant partie d'un ensemble de périodes possibles.

$$c_3 : \quad \epsilon_{jk} \leq \alpha_{ik}, \quad \forall j \in \mathcal{E}, \quad \forall k \in \mathcal{T} \tag{2.8}$$

La contrainte (2.8) indique que toutes les préassignations, représentées par la matrice α, doivent être respectées. Cette matrice est constituée d'éléments $\alpha_{ik} \in \{0,1\}$ égale à 1 si l'examen j peut être donné à la période k, sinon égale à 0. Si un examen i ne présente aucune préassignation, alors $\alpha_{ik} = 1, \forall\, k \in \mathcal{T}$.

2.3.5 Nombre de périodes à l'horaire (f_9)

Le nombre de périodes à l'horaire peut être considéré comme un paramètre à optimiser. En utilisant la même démarche que les autres objectifs, l'équation (2.9) indique que l'objectif est la minimisation de la cardinalité de l'ensemble des périodes.

$$min \quad f_9 = \|\mathcal{T}\| \tag{2.9}$$

Le fait de considérer la longueur de l'horaire comme un paramètre à optimiser rend beaucoup plus difficile la résolution du problème et justifie davantage l'utilisation de méthodes multi-critères puisqu'en minimisant le nombre de périodes on augmente les chances d'avoir des conflits.

En résumé, un grand nombre de modèles peuvent être réalisés en intégrant différentes combinaisons de fonctions d'objectifs et de contraites. Dans la plupart des travaux réalisés, les modèles utilisés sont mono-objectifs. On retrouve une combinaison linéaire de critères étant pondérés par des poids. Afin de comparer les différents algorithmes publiés, certains critères de performance ont été employés. La prochaine section présente ces critères.

2.4 Critères de performance et problèmes standards

Pour évaluer et comparer les algorithmes il est important d'établir certains critères de performance. Les critères présentés dans cette section sont reliés à la qualité de l'horaire produit, c'est-à-dire le nombre de conflits, le nombre d'élèves ayant deux examens consécutifs, etc. Il existe de nombreuses façons d'évaluer la qualité des horaires. Chaque ins-

titution d'enseignement a ses propres objectifs et contraintes. Par contre, des problèmes types ont été développés afin de comparer les performances des différents algorithmes. Ces problèmes types sont classifiés et catalogués par Merlot et *al.* [5]. Ils sont énumérés ci-dessous :

a. **P1** : Minimiser f_9 Sujet à c_1

Ce problème consiste à trouver un horaire réalisable (sans conflit) en ne minimisant que le nombre de périodes.

b. **P2** : Minimiser $f = \sum f_{i+1} w_i$ Sujet à c_1

La fonction d'objectif consiste à minimiser le nombre d'élèves ayant deux examens distants de $i - 1$ périodes, i = 1...5 avec w_1 = 16, w_2 = 8, w_3 = 4, w_4 = 2 et w_5 = 1 [29] qui désignent la pondération accordée aux différents critères. Le résultat final de la fonction d'objectif doit être divisé par le nombre d'élèves afin d'obtenir une moyenne. La capacité des locaux n'est pas prise en compte dans ce problème.

c. **P3** : Minimiser f_2 Sujet à c_1 et c_2

Ce problème consite à minimiser le nombre d'élèves ayant deux examens consécutifs [28]. Par contre, la contrainte de capacité totale des locaux par période doit être respectée. La semaine d'examens commencent un lundi matin. Trois périodes sont disponibles à chaque jour sauf le samedi ou seul la période du matin est disponible et aucun examen n'est donné le dimanche. Les élèves ayant un examen le soir et le lendemain matin ne sont pas inclus dans la fonction d'objectif.

d. **P4** : Minimiser f_2 Sujet à c_1 et c_2 et c_3

Encore une fois dans ce problème le but est de minimiser le nombre d'élèves ayant deux examens consécutifs. Cette fois-ci, la contrainte de pré-assignation des examens est active en plus de la capacité totale des locaux par période et des conflits d'horaire.

Des bases de données publiques sont disponibles pour valider les performances des algorithmes développés [6]. Le tableau I montre les caractéristiques de la plupart des bases de données disponibles. La troisième colonne de ce tableau donne le nombre d'examens à ordonnancer. La quatrième et la cinquième colonnes indiquent le nombre d'élèves ainsi que le nombre total d'élèves par examen, c'est-à-dire les inscriptions-cours. Ces données seront utlisées afin de valider les algorithmes développés dans ce travail de recherche. Le tableau II donne les résultats (nombre de périodes) obtenus par les différents auteurs concernant le problèmes P1. Le tableau III donne les résultats (fonction d'objectif modèle P2) obtenus par les différents auteurs concernant le problèmes P2. Finalement, le tableau IV quant à lui, montre les résultats obtenus par les différents auteurs concernant le problème P3 (fonction d'objectif modèle P3). Dans ces tableaux les nombres en caractère gras désignent les meilleurs résultats.

Tableau I

Caractéristiques des bases de données publiques [6]

Données	Institution d'enseignement	Examens	Étudiants	Inscriptions
CAR-F-92	Carleton University, Ottawa	543	18 419	55 552
CAR-S-92	Carleton University, Ottawa	682	16 925	56 877
EAR-F-83	Earl Haig Collegiate Institute, Toronto	189	1 125	8 109
HEC-S-92	École des hautes études commerciales, Montréal	80	2 823	10 632
KFU-S-93	King Fahd University, Dharan	461	5 349	25 113
LSE-F-91	London School of Economics	381	2 726	10 918
MEL-F-01	University of Melbourne	521	20 656	62 247
MEL-S-01	University of Melbourne	562	19 816	60 637
NOT-F-94	University of Nottignam	800	7 896	33 997
RYE-S-93	Ryeson University, Toronto	481	11 483	45 051
STA-f-83	St Andrew's Junior High School, Toronto	138	611	5 751
TRE-S-92	Trent University, Peterborough, Ontario	261	4 360	14 901
UTA-S-92	Faculty of Arts and Sciences, University of Toronto	638	21 266	58 979
UTE-S-92	Faculty of Engineering, University of Toronto	184	2 750	11 793
YOR-F-83	York Mills Collegiate Institute, Toronto	180	941	6 034

Tableau II

Résultats de la fonction d'objectif pour les bases de données publiques selon le modèle P1 (minimisation de la longueur des horaires)

Données	Merlot [5]	Carter [29]	Caramia [8]
CAR-F-92	31	**28**	**28**
CAR-S-91	30	**28**	**28**
EAR-F-83	24	**22**	**22**
HEC-S-92	18	**17**	**17**
KFU-S-93	21	**19**	**19**
LSE-F-91	18	**17**	**17**
MEL-F-01	28	—	—
MEL-S-01	31	—	—
NOT-F-94	23	—	—
RYE-F-92	22	**21**	**21**
STA-F-83	**13**	**13**	**13**
TRE-S-92	21	**20**	**20**
UTA-S-92	32	32	**30**
UTE-S-92	11	**10**	**10**
YOR-F-83	23	**19**	**19**

Tableau III

Résultats de la fonction d'objectif pour les bases de données publiques selon le modèle P2 (optimisation de l'étalement temporel)

Données		Cart.[29]	DiG.[11]	Cara.[8]	Bur.[28]	Mer.[5]	Paq. [7]
CAR-F-92	Meilleure	10.5	5.2	6.0	**4.0**	4.3	—
32 périodes	Moyenne	15.0	5.6	—	**4.1**	4.4	—
CAR-S-91	Meilleure	7.1	6.2	6.6	**4.6**	5.1	—
35 périodes	Moyenne	8.4	6.5	—	**4.7**	5.2	—
EAR-F-83	Meilleure	36.4	45.7	**29.3**	35.1	35.1	40.5
24 périodes	Moyenne	40.9	46.7	—	**35.4**	**35.4**	45.8
HEC-S-92	Meilleure	10.6	12.4	**9.2**	11.3	10.6	10.8
18 périodes	Moyenne	15.0	12.6	—	11.5	**10.7**	12.0
KFU-S-93	Meilleure	14.0	18.0	13.8	13.7	**13.5**	16.5
20 périodes	Moyenne	18.8	19.5	—	**13.9**	14.0	18.3
LSE-F-91	Meilleure	10.5	15.5	**9.6**	10.6	10.5	13.2
18 périodes	Moyenne	12.4	15.9	—	**10.8**	11.0	15.5
MEL-F-01	Meilleure	—	—	—	—	**2.9**	—
28 périodes	Moyenne	—	—	—	—	**3.0**	—
MEL-S-01	Meilleure	—	—	—	—	**2.5**	—
31 périodes	Moyenne	—	—	—	—	**2.5**	—
NOT-F-94	Meilleure	—	—	—	—	**7.0**	—
23 périodes	Moyenne	— —	—	—	—	**7.1**	—
RYE-F-92	Meilleure	7.3	—	**6.8**	—	8.4	—
23 périodes	Moyenne	**8.7**	—	—	—	**8.7**	—
STA-F-83	Meilleure	161.5	160.8	158.2	168.3	**157.3**	158.1
13 périodes	Moyenne	167.1	166.8	—	168.7	**157.4**	159.3
TRE-S-92	Meilleure	9.6	10.0	9.4	**8.2**	8.4	9.3
23 périodes	Moyenne	10.8	10.5	—	**8.4**	8.6	10.2
UTA-S-92	Meilleure	3.5	4.2	3.5	**3.2**	3.5	—
35 périodes	Moyenne	4.8	4.5	—	**3.2**	3.6	—
UTE-S-92	Meilleure	25.8	29.0	**24.4**	25.1	25.1	27.8
10 périodes	Moyenne	30.8	31.3	—	**25.2**	**25.2**	29.4
YOR-F-83	Meilleure	41.7	41.0	**36.2**	36.8	37.4	38.9
21 périodes	Moyenne	45.6	42.1	—	**37.3**	37.9	41.7

Tableau IV

Résultats de la fonction d'objectif pour les bases de données publiques selon le modèle P3 (optimisation de l'étalement temporel avec capacité des locaux)

Données			DiG.[11]	Cara.[8]	Bur.[28]	Mer.[5]
CAR-F-92	Périodes 40	Meilleure	331	424	268	**158**
	Capacité 2000	Moyenne	—	443	—	**212.8**
CAR-S-91	Périodes 51	Meilleure	81	88	74	**31**
	Capacité 1550	Moyenne	—	98	—	**47**
KFU-S-93	Périodes 20	Meilleure	974	512	912	**247**
	Capacité 1995	Moyenne	—	597	—	**282.8**
NOT-F-94	Périodes 23	Meilleure	123	—	269	**83**
	Capacité 1550	Moyenne	134	—	—	**105**
NOT-F-94	Périodes 26	Meilleure	11	44	53	**2**
	Capacité 1550	Moyenne	**13**	—	—	15.6
TRE-S-92	Périodes 35	Meilleure	4 5	2	53 3	**0**
	Capacité 655	Moyenne	—	—	—	**0.4**
UTA-S-92	Périodes 38	Meilleure	772	554	680	**334**
	Capacité 2800	Moyenne	—	625	—	**393.4**

2.5 L'état de l'art

Depuis les années 60, l'automatisation du processus de création des horaires d'examens finaux intéresse de nombreux chercheurs en recherche opérationnelle. Le tableau V synthétise quelques publications récentes qui ont été réalisées sur les modèles standards présentés à la section 2.4. La prochaine section consiste en une revue littéraire des principales méthodes utilisées ainsi que celles montrées dans les tableaux II, III et IV.

Tableau V

Représentation des modèles utilisés par différents auteurs

	$P1$	$P2$	$P3$	$P4$
Basé sur les algorithmes génétiques				
Burke [19]				•
Burke [28]			•	
Basé sur la fouille Tabou				
Gaspero [11]		•	•	•
Paquete [7]		•		
White [18]		•		
Basé sur le recuit simulé				
Merlot [5]	•	•	•	•
Méthodes constructives				
Caramia [8]	•	•		
Carter [29]	•	•		

2.5.1 Méthodes constructives

Ces méthodes sont appliquées pour la résolution des problèmes de création des horaires d'examens depuis fort longtemps. Caramia et *al.* [8] ont développé une série d'heuristiques capables de résoudre les problèmes P1, P2 et P3 avec d'excellents résultats. Dans une première phase, les auteurs appliquent une heuristique afin de créer une première solution réalisable en assignant les examens qui présentent de fortes chances de créer des conflits. Dans un deuxième temps, une autre heuristique est appliquée pour optimiser les

proximités sans réduire la taille de l'horaire. Si cette méthode échoue, une période supplémentaire est ajoutée et l'heuristique est appliquée de nouveau. Les résultats publiés par les auteurs sur les bases de données publiques sont excellents. Par contre, les performances de l'algorithme décroît à mesure que la taille du problème augmente (voir tableau III, page 37).

Une autre étude importante dans ce domaine est celle effectuée par Carter et Laporte [29]. Ces derniers ont fait l'étude de l'ensemble des méthodes d'initialisation présentées à la section 1.7. Afin d'améliorer les peformances des algorithmes, les auteurs ont ajouté deux techniques supplémentaires soit le *backtracking* et l'utilisation d'une clique maximale. Dans le contexte des horaires d'examens, une clique est représentée par un groupe d'examens fortement connectés, c'est-à-dire avec beaucoup d'élèves en commmuns. Le *backtracking* est utilisé lorsqu'un examen ne peut être assigné à aucune période. Dans ce cas, un retour en arrière doit être fait, c'est-à-dire que certaines assignations déjà effectuées sont annulées. L'utilisation d'une clique maximale quant à elle permet de séparer le problème en deux parties. Tout d'abord, il est possible de commencer l'assignation par le groupe d'examens le plus difficile à assigner, en d'autre termes, de déterminer le plus grand groupe d'examens reliés par des élèves communs.

La première remarque importante est l'influence de la base de données utilisée. Les auteurs ont fait remarquer qu'avec certaines données beaucoup plus faciles à résoudre, l'utilisation des techniques d'initialisation et de *backtracking* n'ont aucune influence. Par contre, pour des problème plus difficiles ce choix est très critique. Dans le cas des problèmes plus difficiles, l'utilisation du *backtracking* réduit en moyenne le nombre de périodes d'un horaire d'examens de 50%. Selon Carter et *al.* la technique d'initialisation donnant les meilleurs résultats est le *saturation degree* (**SD**). L'ajout de la détermination d'une clique maximale semble importante pour les bases de données qui représentent de vrai PCHE. Les auteurs font mention que les données synthétiques générées de façon aléatoire qui sont utilisées pour l'ajustement des algorithmes présentent un grand nombre de cliques de

taille plus ou moins semblable. Dans ce contexte, l'utilisation de la clique maximale ne présente pas davantage ce qui n'est pas le cas des vraies données où le nombre de cliques est beaucoup plus faible. Les auteurs ont été les premiers à présenter des résultats sur les problèmes standards P1 et P2. Malgré le fait que les auteurs proposent un grand nombre de combinaisons d'heuristiques les résultats demeurent dans la moyenne. De plus, le choix de la combinaison d'heuristiques appropriée pour une base de données spécifique ne semble pas être trivial.

2.5.2 Fouille Tabou

Dans le domaine de la génération automatique des horaires d'examens, la fouille Tabou (FT) est l'une des méthodes les plus citées. De nombreux travaux ont été réalisés à l'aide de cette fouille locale. White et *al.* [18] ont travaillé à la résolution du problème P2 à l'aide de deux FT. La première fouille sert à assigner les examens qui n'ont pu être traités suite à l'initialisation de la solution par la technique *Largest Degree*. Une fois la solution initiale obtenue, une deuxième FT est appliquée pour l'optimisation de proximités. Une des caractéristiques importantes de leur travail est le calcul de la *tenure* Tabou. La *tenure* Tabou désigne la durée à laquelle une solution est présente dans la liste Tabou. Les auteurs proposent deux méthodes basées sur les caractéristiques des bases de données afin de calculer la *tenure* Tabou. Le nombre de liens (examens ayant des élèves en commun), le nombre d'inscriptions-cours ainsi que la pondération des liens entre les examens sont les caractéristiques utilisées. Cet ajustement de paramètres est un avantage très intéressant, car dans les publications concernant les bases de données standard, on note que les meilleurs résultats ne sont pas obtenus systématiquement par le même chercheur. Ceci implique que les bases de données ont possiblement des caractéristiques bien différentes qui ont une influence sur les performances des algorithmes. Malheureusement White et *al.* n'ont donné des résultats que sur deux bases de données publiques.

Paquete et *al.* [7] ont, pour leur part, implanté une FT où les contraintes sont considérées comme des objectifs. Ils accordent ensuite une priorité aux objectifs selon un ordre lexicographique. Ces derniers ont réussi à obtenir de très bons résultats sur le problème P2. De plus, comme la distinction entre les différents types de proximités est faite selon un ordre de priorité (lexicographique) ceci élimine l'attribution à priori d'une pondération des critères. Un des désavantages notés est que le paramètre servant au calcul de la *tenure* Tabou semble avoir une influence importante sur la qualité des solutions et dans certains cas, aucune solution réalisable n'est trouvée.

Un autre travail très représentatif de ce domaine est celui de Di Gaspero [11] et *al.*. Ils utilisent à chaque itération deux listes qui sont mises à jour. Ces listes sont appelées liste des violations (*VL*) et la liste des violations fortes (*HVL*). Ces listes sont utilisées afin de déterminer les solutions voisines. La liste *VL* est constituée de l'ensemble des examens créant des violations de contraintes faibles ou fortes. La liste *HVL* quant à elle est constituée d'examens ne créant que des violations fortes (conflits d'horaire). Afin de déterminer une solution voisine, un examen est selectionné dans les listes *HVL* ou *VL* par une fouille exhaustive. La FT établie par les auteurs considère les solutions irréalisables (avec conflits). C'est à l'intérieur de la fonction d'objectif qu'ils différencient les contraintes fortes et les faibles en les pondérant d'une manière dynamique basée sur la méthode de *Shifting penalty* implantée originalement par Gendreau et *al.* [32]. Les auteurs ont aussi eu recours à l'utilisation d'une méthode d'initialisation dont la nature n'est pas mentionnée. Les résultats obtenus peuvent se comparer à ceux de Carter et *al.* [29] bien qu'ils sont, en général, relativement plus faibles. L'utilisation des deux listes *VL* et *HVL* permet d'augmenter les efforts de recherche pour la satisfaction des contraintes fortes. Par contre, les contraintes fortes et faibles sont malgré tout pondérées dans la fonction d'objectif. L'ajustement des poids devient donc très critique.

2.5.3 Le recuit simulé

La méthode du recuit simulé (RS), n'est pas la plus populaire dans le domaine de création des horaires. Une des contributions les plus significatives, dans le contexte de création des horaires d'examens, est celle de Merlot et *al.* [5] qui ont étudié les problèmes P1 à P4. L'approche de Merlot est une méthode hybride. Par contre, le plus gros du travail est réalisé par un RS. La méthode se divise en trois étapes, soit la recherche d'une solution initiale réalisable par satisfaction de contraintes, l'optimisation de l'étalement temporel de l'horaire obtenu (recuit simulé) et le dernier raffinement (fouille par escalade). La satisfaction de contraintes qui consitue la première étape de l'approche consiste à assigner à chaque examen une période parmi une liste de périodes où l'assignation satisfait l'ensemble des contraintes. Les listes de tous les examens sont mises à jour à chaque assignation. Lorsque la liste d'un examen est vide, c'est-à-dire qu'il ne peut être assigné en respect des contraintes, un retour en arrière est effectué.

Pour ce qui est du RS à chaque itération un seul voisin est choisi. Avant de comparer la valeur de la fonction d'objectif, la solution doit d'abord être réalisable (capacité des locaux dans le cas de P3 et pré-assignation des examens dans le cas de P4). Si la solution est réalisable alors la valeur de la solution est comparée avec la solution actuelle ou avec la probabilité de remplacement. En procédant ainsi l'algorithme reste dans le domaine réalisable. Suite à l'application du RS, une troisème méthode est ajoutée : la fouille par escalade. À chaque itération de la fouille par escalade, toutes les solutions voisines sont évaluées et l'algorithme progresse en choisisant la meilleure. Cette application permet un dernier raffinement. Les auteurs notent qu'une dizaine d'itérations semblent suffisantes. Merlot a grandement contribué à la résolution des problèmes standards. En plus d'obtenir d'excellents résultats pour le modèle P1 et P2, Merlot a implanté avec succès son algorithme sur les modèle P3 et P4 pour une très grande partie des bases de données disponibles. Par contre la faiblesse de son algorithme réside dans l'obtention d'une solution initiale sans conflit. Le tableau II (page 36) montre qu'en général plus la taille du problème augmente,

plus le nombre de périodes nécessaires pour trouver un horaire sans conflit s'éloigne de la moyenne obtenue par les autres chercheurs.

2.5.4 Méthodes évolutives

La popularité toujours grandissante des algorithmes génétiques (AG) a aussi laissé sa marque dans le domaine de la création des horaires d'examens. Un bon nombre de travaux ont été publiés sur ce sujet. Par contre, l'utilisation des AG simples ne fait pas l'unanimité puisque la plupart des travaux portant sur cet algorithme utilisent divers moyens pour augmenter leur performance. Pour un AG simple, les performances sont plus faibles comparativement à une fouille Tabou et ce, autant pour la qualité des horaires que pour la vitesse de convergence [20]. Un travail important dans cet axe de recherche est celui de Burke et *al.* [28]. Leur application se classe plutôt dans les techniques appelées *memetic algorithm*, c'est-à-dire une méthode évolutive utilisant une fouille locale. Dans le cas de Burke et *al.*, ils utilisent une population initiale créée à partir d'une heuristique qui assigne les examens dans les périodes sélectionnées par l'utilisation d'une roulette biaisée par pondérations. Ces pondérations sont basées sur les violations des différentes contraintes. Par la suite, les solutions sont choisies au hasard, puis mutées. Pour terminer, une fouille en escalade est appliquée pour optimiser la qualité des solutions. Toutes les solutions affectées par ces opérateurs sont placées dans la prochaine population. Le cycle se répète jusqu'à l'obtention d'un horaire satisfaisant. Comme le montre les tableaux III et IV, des résultats intéressants ont été obtenus sur les modèles P2 et P3. De plus, les opérateurs utilisés, sélection, mutation et fouille par escalde sont très faciles à implanter et ne requièrent que très peu de paramètres. Malgré tout, l'algorithme ne fonctionne qu'avec des horaires d'examens dont la taille est fixe et la fonction d'objectif contient des pondérations dont les valeurs n'ont pas été expliquées.

2.6 Conclusion

On remarque qu'avec l'ensemble des objectifs et des contraintes qui ont été énoncés à la section 2.3, un grand nombre de modèles peuvent être utilisés. Pour cette raison, il peut être difficile de comparer les algorithmes entre eux si ces derniers ont été modifiés et ajustés pour obtenir de bonnes performances sur les données et sur un modèle propre à leurs institutions. Par contre, des modèles standard (P1 à P4) ont été instaurés et des données publiques ont été publiées afin de permettre une comparaison plus juste. Plusieurs chercheurs ont appliqué leurs méthodes sur ces bases de données, on note, entre autres, que Merlot et *al.* [5] utilisent une méthode hybride incluant un recuit simulé, Burke et *al.* [28] utilisent un algorithme évolutif hybride et DiGaspero et *al.* [11] ont développé une fouille Tabou. Toutes ces implantations ont montré que les techniques du recuit simulé et la fouille Tabou donnent de très bons résultats. Les algorithmes génétiques quant à eux demandent quelques modifications, comme l'ajout d'opérateurs de fouille locale, afin d'obtenir des performances acceptables. Ces approches hybrides ont quand même l'avantage d'être facilement applicable pour l'implantation de méthodes multi-critères. Il est clair que les objectifs du modèle P1 et P2 sont conflictuels. En minimisant le nombre de périodes les chances d'augmentation du taux de proximités seront plus élevées. Le prochain chapitre portera donc sur l'application d'algorithmes d'optimisation multi-critères basés sur des méthodes évolutives.

CHAPITRE 3

ALGORITHMES ÉVOLUTIFS MULTI-CRITÈRES EXISTANTS

3.1 Introduction

Bien peu de travaux ont été réalisés sur l'optimisation multi-critères dans le domaine des horaires d'examens. Aucun travail n'a encore été réalisé jusqu'à maintenant sur l'optimisation multi-critères des horaires d'examens en utilisant les bases de données publiques. C'est pour cette raison, entre autre, que cette étude portera sur l'utilisation d'algorithmes évolutifs multi-critères (AEMC) déjà existants. Une revue des principales techniques d'optimisation multi-critères sera effectuée dans ce chapitre. Par la suite, une première série de tests seront effectués sur les bases de données publiques. Afin de comparer les résultats avec ceux déjà publiés, un modèle multi-critères appelé P1-P2 sera proposé. Les résultats obtenus montront que les AEMC les plus populaires tels que le NSGA-II et le SPEA-II sont inéfficaces pour la résolution d'un PCHE de forme mutli-critères. Les causes de cet échec seront mises en évidence afin de guider la conception d'un éventuel algorithme multi-critères. Dans ce contexte, ce chapitre permettra donc de justifier la conception d'un nouvel algorithme évolutif multi-critères.

3.2 Justification du choix d'une méthode évolutive multi-critères

Les problèmes d'optimisation multi-critères (POMC) sont une généralisation des problèmes d'optimisation. Le but est de trouver l'ensemble de Pareto optimal comprenant des solutions non dominées (définition 1.6). Les PCHE sont fondamentalement des problèmes d'optimisation multi-critères. Tout comme la plupart des problèmes d'optimisation, ils doivent traiter avec plusieurs objectifs. Les techniques d'optimisation mono-critère quant à elles, assemblent tous les objectifs dans une même fonction. Si l'on suppose un problème d'optimisation sans contrainte défini par min $\{f_1, f_2, \ldots, f_m\}$ $m > 1$ la fonction

d'objectif utilisée par un algorithme mono-critère sera $\sum f_i w_i, i = 1 \ldots m$. Le vecteur W = $\{w_1, w_2, \ldots, w_m\}$ est le poids qui est accordé à chaque objectif. Ce vecteur de poids ne représente qu'une mesure de préférence. Si l'on prend par exemple les poids w_1 = 16 et w_2 = 8 du problème P2 (section 2.4), ceci implique que la fonction f_1 est deux fois plus importante que f_2 et que la valeur finale de f_1 risque d'être deux fois plus faible que celle de f_2. Par contre, si les critères de préférence changent, un nouveau vecteur W doit être identifié et une nouvelle recherche doit être faite. Une autre différence est que dans un problème d'optimisation multi-critères l'objectif est double. Contrairement aux techniques d'optimisation mono-critère où le seul objectif à atteindre est la maximisation ou la minimisation de la fonction d'objectif, pour un problème multi-critères l'obtention de l'ensemble de Pareto optimal du problème et la diversité des solutions qui en font partie sont deux objectifs indissociables. Cette diversité est un objectif important, car elle permet d'effectuer une meilleure décision à posteriori. En considérant qu'un PCHE est un problème multi-critères, l'application d'une méthode de résolution multi-critères semble logique. Les méthodes évolutives offrent aussi l'avantage d'utiliser une population d'individus. Dans une application multi-critères il sera donc possible de faire converger l'ensemble de la population vers un ensemble de Pareto en effectuant un seul lancement de l'algorithme. Ceci éliminera du même coup toute pondération des fonctions d'objectifs appliquée à priori. De plus, les résultats obtenus par Burke [28] montrent que les algorithmes génétiques peuvent être aussi performants que tout autre algorithme tels que le recuit simulé ou la fouille Tabou.

3.3 Algorithmes évolutifs multi-critères

Les algorithmes évolutifs multi-critères (AEMC) sont bien adaptés à la résolution d'un problème multi-critères puisqu'ils permettent de trouver, en un seul lancement, un ensemble de solutions non dominées. De nombreuses techniques telles que NGSA [30] , NSGA-II [14], SPEA [25] et SPEA-II [13] sont de bons exemples d'AEMC parus dans

la littérature. Les sections qui suivent présentent ces différentes techniques ainsi que leurs caractéristiques.

3.3.1 NSGA

L'algorithme NSGA proposé par Deb et Srinivas [30] est semblable aux algorithmes génétiques qui ont été présentés à la section 1.6.5. Aucun choix n'est imposé sur les opérateurs génétiques de mutation, croisement, sélection et insertion. La seule différence se trouve au niveau de l'implantation de l'opérateur d'assignation de la valeur d'adaptation des individus.

ALGORITHME 5

Procédure d'assignation de la valeur d'adaptation du NSGA [30]

Données :

$\mathcal{P}$:	Population à trier.
$\mathcal{F}_k$:	Ensemble de Pareto.
A :	Opérateur de fonction de partage.
T :	Opérateur de tri par ensembles de Pareto.
ω_i :	Valeur d'adaptation de la solution i.
σ_{share} :	Paramètre servant à la gestion de la diversité.
ε :	Petit nombre positif.
$\Omega_{\min}$:	Valeur d'adaptation minimum des solutions d'un ensemble de Pareto.
$\mathbf{z}_k^{(i)}$:	Solution i faisant partie de l'ensemble de Pareto k.

début
- Initialiser σ_{share}, ε, $\Omega_{\min} = \|\mathcal{P}\| + \varepsilon$ et $k = 1$
- Appliquer $\mathcal{P}' = \mathrm{T}(\mathcal{P})$
- **pour** *chaque* $\mathcal{F}_k \in \mathcal{P}'$ **faire**
  - **pour** *chaque* $\mathbf{z}_k^{(i)} \in \mathcal{F}_k$ **faire**
    - Assigner $\omega_i = \Omega_{\min} - \varepsilon$
    - Modifier $\omega_i = \mathrm{A}(\mathbf{z}_k^{(i)})$
  - Mettre à jour $\Omega_{\min} = \min(\omega_i : \forall\, i \in \mathcal{F}_k)$
  - k=k+1

fin

Cette assignation, faite par l'opérateur T (algorithme 5), repose sur le rang de l'ensemble de Pareto dans lequel la solution est incluse. Dans cette procédure, toutes les solutions du premier ensemble auront comme valeur d'adaptation $\omega_i = ||\mathcal{P}|| \ \forall i \in \mathcal{F}_1$. Pour tous les autres ensembles la valeur d'adaptation sera égale à la plus petite valeur de la solution du ensemble qui le précède retranché de ε, un petit nombre positif. De cette façon, il n'y aura jamais deux solutions faisant partie de deux ensembles différents ayant la même valeur d'adaptation. Le principal désavantage de cet algorithme est l'absence d'élitisme. Les prochaines sections présenteront différents algorithmes qui permettent l'ajout de cet opérateur.

3.3.2 NSGA-II

La présence d'élites augmente les chances de créer de meilleurs enfants [9]. Ceci implique qu'un algorithme utilisant l'élitisme convergera plus rapidement. Dans le cas des problèmes d'optimisation mono-critère les élites se repèrent facilement. Ils ont la meilleure valeur par rapport à la fonction d'objectif. Pour les problèmes d'optimisation multi-critères cette affirmation n'est plus valide, car il y a présence de plusieurs fonctions d'objectifs. Afin de gérer les élites, l'algorithme NSGA-II de Deb [14] utilise le rang de l'ensemble de Pareto où se trouve la solution. Donc les élites seront identifés comme étant les solutions faisant partie de l'ensemble de Pareto $\mathcal{F}_1$. Pour gérer la diversité au niveau des ensembles de Pareto, l'algorithme utilise une implantation de l'opérateur de sélection bien particulière, soit le *Crowded Tournament* (algorithme 6, page 52).

Définition 3.1 *Soit une population $\mathcal{P}$, deux solutions $\mathbf{z}_m^{(i)} \in \mathcal{F}_m$ et $\mathbf{z}_n^{(j)} \in \mathcal{F}_n$, $\{\mathcal{F}_m, \mathcal{F}_n\} \in \mathcal{P}$, la solution $\mathbf{z}_m^{(i)}$ remportera le tournoi si :*

1. $m < n$

   OU

2. $m = n$ *et* $d_1 > d_2$

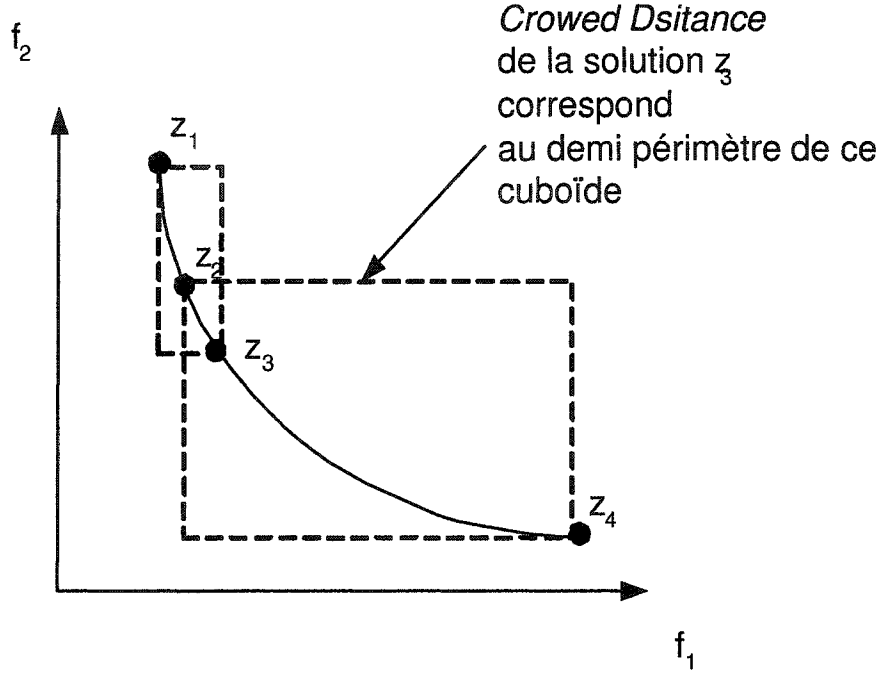

Figure 6 Cas hypothétique de l'assignation de la *Crowded Distance*

Dans la procédure d'assignation de la *Crowded Distance*, notée par $\mathrm{D} : \mathcal{P} \rightarrow \mathbb{R}^{+}$, la variable d_i évalue la densité des solutions présentes autour d'une solution i. L'assignation de cette valeur aura comme effet de diminuer les chances de survie d'une solution i présente dans une région où plusieurs autres solutions y sont concentrées. Concrètement, comme le montre la figure 6, le fait de trier le vecteur d'indice $I^{(j)}$ et d'assigner une distance très grande aux premières et dernières solutions permet de donner priorité aux solutions extrêmes d'un ensemble de Pareto pour chaque fonction d'objectif (voir algorithme 6, page 52). Pour les solutions intermédiaires, toujours grâce au vecteur de tri, la distance est donnée par le demi périmètre du cuboïde entourant ces solutions. Dans l'exemple montré à la figure 6, la solution z_2 est celle qui aura plus de chance d'être rejetée étant donné que le demi périmètre de son cuboïde est le plus petit. Il est aussi important de noter que la densité est mesurée dans l'espace d'objectif et non dans l'espace de décision et que la méthode inclue directement une normalisation, ce qui est indispensable lors des calculs de distances.

Finalement l'algorithme NSGA-II (algorithme 7, page 53) utilise deux populations. La population $\mathcal{Q}$, constituée de l'ensemble des individus qui ont été créés par l'application de l'opérateur de sélection Λ_1 (définition 3.1) et une deuxième population, $\mathcal{P}$ de taille N. À chaque itération les deux populations, de taille N, seront combinées dans une même population $\mathcal{R}$ et triées de façon à obtenir les ensembles de Pareto. Par la suite, une nouvelle population $\mathcal{P}_{t+1}$ de taille N sera constituée des meilleurs ensembles de Pareto de la population $\mathcal{R}$. Pour y arriver, les solutions des ensembles de Pareto $\mathcal{F}_k \subset \mathcal{R}_t$ seront incluses dans la population jusqu'à ce que la taille devienne supérieure ou égale à N. Si, suite à l'ajout du dernier ensemble de Pareto possible, la taille de la population est supérieure à N, alors le dernier ensemble sera trié selon la *crowded Distance* et les solutions ayant les plus petites distances seront éliminées jusqu'à ce que la taille de $\mathcal{P}_{t+1}$ soit égale à N. L'opérateur de sélection, croisement et mutation seront alors appliqués sur $\mathcal{P}_{t+1}$ pour créer la nouvelle population $\mathcal{Q}_{t+1}$.

ALGORITHME 6

Opérateur d'assignation de la *Crowded Distance* [14]

Données :
I : Vecteur d'indice.
$\mathcal{F}_k$: Ensemble de Pareto.
d_i : *Crowded Distance* de la solution i.
f_i : Fonction d'objectif.
$f_m^{\max}$: Valeur maximale de la fonction d'objectif m.
$f_m^{\min}$: Valeur minimale de la fonction d'objectif m.

début
  Initialiser $l = \|\mathcal{F}\|$
  pour *chaque* $\mathbf{z}^{(i)} \in \mathcal{F}$ **faire**
    Initialiser $d_i = 0$
  pour *chaque fonction d'objectif* f_j **faire**
    Trouver le vecteur d'indice $I^{(j)}$ = Trier(f_j,<)
  pour *chaque fonction d'objectif* f_j **faire**
    Assigner $d_1^m = d_l^m = \infty$
    pour *chaque* $\mathbf{z}^{(i)}$ $i = 2 \ldots (l-1)$ **faire**
      Calculer
      $d_{I_j^m} = d_{I_j^m} + \frac{f_m^{I_{j+1}^m} - f_m^{I_{j-1}^m}}{f_m^{\max} - f_m^{\min}}$
fin

ALGORITHME 7

NSGA-II [14]

Données :

D : Opérateur d'assignation de *Crowded Distance*.
$\mathcal{F}_k$: Ensemble de Pareto.
M : Opérateur de mutation.
$\mathcal{P}$: Population d'individus.
$\mathcal{Q}$: Population des enfants.
$\mathcal{R}$: Population enfants - individus.
S : Opérateur de sélection.
T : Opérateur de tri par ensembles de Pareto.
X : Opérateur de croisement.
$\mathbf{z}_k^{(i)}$: Solution i faisant partie de l'ensemble de Pareto k .

début

Créer une population initiale $\mathcal{P}_0$ et $\mathcal{Q}_0$
pour *chaque itération* t **faire**
  $\mathcal{R}_t = \mathcal{P}_t \cup \mathcal{Q}_t$
  Appliquer $\mathcal{R}' = \mathrm{T}(\mathcal{R})$
  Assigner $\mathcal{P}_{t+1} = \emptyset$
  tant que $\|\mathcal{P}_{t+1}\| < N$ **faire**
    $k = k + 1$ et $\mathcal{P}_{t+1} = \mathcal{P}_{t+1} \cup \mathcal{F}_k$
  Calculer $\mathrm{D}(\mathbf{z}_k^{(i)})\ \forall\, i \in \mathcal{F}_k$
  Trier($\mathcal{F}_k, <_d$)
  $i = 1$
  répéter
    $\mathcal{F}_k = \mathcal{F}_k \setminus \mathbf{z}_k^{(i)}$
    $i = i + 1$
  jusqu'à *ce que* $\|\mathcal{P}_{t+1}\| = N$
  Appliquer
  $\mathcal{Q}_{t+1} = \mathrm{M}(\mathrm{X}(\mathrm{S}(\mathcal{P}_{t+1})))$

fin

3.3.3 SPEA

Dans un tout autre ordre d'idée, Zitzler et Thiele [22] proposent un algorithme évolutif utilisant une population externe $\mathcal{Q}$ de taille $\overline{N}$ qui sera appelée archive. Cette population sera composée des élites. L'algorithme utilise aussi une population $\mathcal{P}$ de taile N. L'algorithme, qui est totalement différent de ceux proposés par Deb, calcule, à chaque itération, l'ensemble de Pareto $\mathcal{F}_1$ de $\mathcal{P}_t$. Les solutions non dominées sont alors copiées dans $\mathcal{Q}$. Par la suite, toutes les solutions dominées de $\mathcal{Q}$ sont excluses. En appliquant une implantation de type tournoi binaire sur la population $\{\mathcal{P}_t \cup \mathcal{Q}\}$ la population $\mathcal{P}_{t+1}$ sera alors créée.

ALGORITHME 8

Opérateur de mise à jour de l'algorithme SPEA [25]

Données :
$\mathcal{Q}$: Population des élites.
C : Ensemble des regroupements C_i.
$\overline{N}$: Nombre d'élites désiré.

début
  Initialiser $C = \{C_1, C_2, \ldots, C_i\} \mid C_i = \mathbf{z}_i, i \in \mathcal{Q}$
  répéter
    pour *chaque paire de regroupement* $\{C_j, C_k\} \subset C$ **faire**
      Calculer $dc_{jk}(C_j, C_k)$
    Identifier $\{C_j, C_k\} \subset C \mid dc_{jk} = \min(dc_{jk} : \forall\, j, k \in C)$
    Retirer $C = C \setminus \{C_j\}$ et $C = C \setminus \{C_k\}$
    Ajouter $C = C \cup \{C_j \cup C_k\}$
  jusqu'à *ce que* $\|C\| \leq \overline{N}$
  Initialiser $\mathcal{Q} = \{\emptyset\}$
  pour *chaque* $C_i \in C$ **faire**
    Appliquer $C_i = \mathbf{z}_q \mid d_{jk} = \min(d_{jk} : \forall\, j, k \in C_i)$
    Mettre à jour $\mathcal{Q} = \mathcal{Q} \cup C_i$
fin

Un des aspects importants de l'algorithme est la mise à jour de la population d'élites $\mathcal{Q}$. Lors de la copie des solutions non dominées $\mathcal{F}_1 \in \mathcal{P}$ dans $\mathcal{Q}$, si le nombre de solutions

de l'ensemble $\mathcal{F}_1 > \overline{N}$ alors un opérateur de mise à jour (algorithme 8), noté par R : $\mathcal{P} \rightarrow \mathcal{P}$, sera appliqué afin de réduire la taille de $||\mathcal{Q}|| = \overline{N}$. Initialement, cet opérateur crée un regroupement pour chaque solution. Toutes les distances entre chaque paire de regroupements (C_1,C_2) sont ensuite calculées. Cette distance est calculée par rapport à la distance moyenne de toutes les solutions comprises dans le regroupement (algorithme 8). Par la suite, l'opérateur fera l'union des deux regroupements ayant la plus petite distance. Cet union sera effectué jusqu'à ce que le nombre de regroupements soit égal à $\overline{N}$. Dès lors, une seule solution sera choisie par regroupement et la population $\mathcal{Q}$ pourra être constituée. La solution choisie est déterminée par la plus petite distance moyenne entre les solutions du regroupemement.

L'implantation de l'opérateur d'assignation de la valeur d'adaptation s'effectue en deux étapes. Tout d'abord, à l'aide de l'équation (3.1) la valeur d'adaptation ω_i est calculée pour chaque individu $i \in \mathcal{Q}$. La variable n_i détermine le nombre de solutions dominées dans $\mathcal{P}_t$ par la solution $i \in \mathcal{Q}$.

$$\omega_i = \frac{n_i}{N+1} \quad i \in \mathcal{Q} \tag{3.1}$$

$$\omega_j = 1 + \sum_{i \in Q \mid i \succcurlyeq j} \omega_i \quad i \in \mathcal{Q}, j \in \mathcal{P}_t \tag{3.2}$$

Pour les membres de la population $\mathcal{P}_t$ la valeur d'adaptation se calcule selon la relation (3.2). De cette façon, lors du tournoi binaire, les élites seront assurés d'avoir une valeur d'adaptation inférieure à 1 et les membres de $\mathcal{P}_t$ auront une valeur supérieure à 1. Le SPEA (algorithme 9) comporte certaines faiblesses. Tout d'abord, étant donné que seul l'ensemble de Pareto $\mathcal{F}_1$ doit être identifié, il peut arriver que deux solutions $\{\mathbf{z}_i, \mathbf{z}_j\} \notin \mathcal{F}_1$ tel que $\mathbf{z}_i \succcurlyeq \mathbf{z}_j$ possèdent une valeur d'adaptation identique. C'est le cas, entre autres, lorsque ces deux solutions sont dominées par les mêmes solutions $\mathbf{z}_k \in \mathcal{F}_1$. Le deuxième point faible vient de la manière dont l'algorithme traite la diversité. En aucun cas, lors du calcul de la valeur d'adaptation et la mise à jour de l'archive, les solutions extrêmes de

l'ensemble de Pareto auront priorité sur les autres. Ceci pourrait éventuellement réduire la couverture de la population dans l'espace d'objectif. La prochaine méthode présentée sera donc le SPEA-II qui tient compte des faiblesses mentionnées ci-haut.

ALGORITHME 9

SPEA [22]

Données :
M : Opérateur de mutation.
N : Taille de la population $\mathcal{Q}$.
$\overline{N}$: Taille de la population $\mathcal{P}$.
$\mathcal{F}_1$: Ensemble optimal de Pareto.
$\mathcal{P}$: Population d'individus.
$\mathcal{Q}$: Population des élites.
R^1 : Opérateur de mise à jour de l'archive.
S : Opérateur de sélection.
X : Opérateur de croisement.
ω_i : Valeur d'adaptation de la solution i.

début
  Initialiser $\mathcal{P}_0$ de taille N et $\mathcal{Q} = \{\emptyset\}$
  pour *chaque itération* t **faire**
    Appliquer $\mathcal{Q} = \mathcal{Q} \cup \mathcal{F}_1, \mathcal{F}_1 \subset \mathcal{P}_t$
    Identifier $\mathcal{F}_1, \mathcal{F}_1 \subset \mathcal{Q}$
    Mettre à jour $\mathcal{Q} = \mathcal{F}_1$
    si $\|Q\| > \overline{N}$ **alors**
      Appliquer $Q = \mathrm{R}^1(\mathcal{Q})$
    Calculer ω_i pour chaque $i \in \mathcal{P}$
    Calculer ω_j pour chaque $j \in \mathcal{Q}$
    Appliquer
    $\mathcal{P}_{t+1} = \mathrm{M}(\mathrm{X}(\mathrm{S}(\mathcal{P}_t \cup Q)))$
fin

3.3.4 SPEA-II

Récemment Zitzler et *al.* [13] ont proposé une version améliorée de leur SPEA. Les modifications majeures se situent au niveau de l'assignation de la valeur d'adaptation et de

la procédure de mise à jour de la population d'élites $\mathcal{Q}$. L'algorithme doit s'assurer que les valeurs d'adaptation de deux solutions $\mathbf{z}_i$, $\mathbf{z}_j$ tel que $\mathbf{z}_i \succcurlyeq \mathbf{z}_j$ soient toujours assignées de sorte que ω_i soit supérieur en tout temps à ω_j. Ainsi, la première étape de l'assignation de la valeur d'adaptation s'effectue sur l'ensemble $\mathcal{P} \cup \mathcal{Q}$. Cette valeur est déterminé en calculant, pour chaque solution $\mathbf{z}_i \in \mathcal{P} \cup \mathcal{Q}$, la valeur $S(i)$ (équation 3.3) qui représente le nombre de solutions que domine $\mathbf{z}_i$.

$$S(i) = \{z_j \mid j \in \{P \cup Q\},\ z_i \succcurlyeq z_j\} \tag{3.3}$$

$$sk(i) = \sum_{j \in \{Q \cup P\} \wedge z_j \succcurlyeq z_i} \|S(j)\| \tag{3.4}$$

ALGORITHME 10

Opérateur de mise à jour de l'algorithme SPEA-II [13]

Données :
$\mathcal{P}$: Population d'individus.
$\mathcal{Q}$: Population des élites.
$\overline{N}$: Nombre d'élites désiré.

début
  si $\|\mathcal{Q}\| < \overline{N}$ **alors**
    Complèter $\mathcal{Q}$ par les $\overline{N} - \|\mathcal{Q}\|$ meilleurs solutions de $\mathcal{P}$
  sinon si $\|\mathcal{Q}\| > \overline{N}$ **alors**
    répéter
      $\mathcal{Q} = \mathcal{Q} \setminus \mathbf{z}_i \mid \mathbf{z}_i \leq_d \mathbf{z}_j \ \forall\, j \in \mathcal{Q}$
    jusqu'à *ce que* $\|\mathcal{Q}\| = \overline{N}$
fin

Dans un deuxième temps, pour chaque solution $\mathbf{z}_i \in \mathcal{P} \cup \mathcal{Q}$ la valeur $sk(i)$ sera calculée. La valeur $sk(i)$ représente le nombre de solutions qui dominent la solution z_i. Ceci implique que toutes les solutions ayant une valeur $sk(i) = 0$ seront non dominées dans $\mathcal{P} \cup \mathcal{Q}$. Par contre, cette affectation n'assure pas une diversité à l'intérieur de la population. C'est

pourquoi Zitzler et *al.* ont ajouté une mesure de densité qui est basée sur la méthode du k^{ieme} voisin. La valeur d'adaptation finale sera donc obtenue par la relation :

$$\omega_i = sk_i + dk_i \tag{3.5}$$

où les deux valeurs $sk(i)$ et $dk(i)$ sont additionnées. La procédure de l'obtention de la mesure de distance $dk(i)$ se divide en quatre étapes :

1. **Calculer** $D(j) = d_{ij} \; \forall \; j \in \{\mathcal{Q} \cup \mathcal{P}_t\}$
2. **Trier**$(D(j), <)$
3. **Initialiser** $\sigma_i^k = D(k)$ où $k = \sqrt{\|\mathcal{Q}\| + \|\mathcal{P}\|}$
4. **Affecter** $dk(i) = \frac{1}{\sigma_i^k + 2}$

Zitzler propose d'utiliser une valeur de $k = \sqrt{\|\mathcal{P}\| + \|\mathcal{Q}\|}$. La valeur de $dk(i)$ étant comprise entre [0,1], toute solution ayant une valeur d'adaptation $\omega_i \leq 1$ sera considérée non dominée. Pour ce qui est de la formation de la population $\mathcal{Q}$, les auteurs ont décidé de garder un nombre constant d'individus. Tout comme le SPEA, à chaque itération toutes les solutions non dominées de $\mathcal{P}$ sont copiées dans $\mathcal{Q}$. Ceci implique encore une fois la présence d'une procédure de mise à jour de l'archive $\mathcal{Q}$. Cette procédure, notée par $\mathrm{R}_2 : \mathcal{P} \rightarrow \mathcal{P}$ introduit un nouvel opérateur de comparaison (définition 3.2). La relation $\mathbf{z}_i \leq_d \mathbf{z}_j \in \mathcal{Q}$ indique que la solution $\mathbf{z}_i$ est plus rapprochée d'une solution (en terme de distance euclédienne dans l'espace d'objectif) que peut l'être $\mathbf{z}_j$ par rapport à toutes les solutions de $\mathcal{Q}$. En se servant de la figure 7, il est possible de montrer un cas hypothétique où la relation $z_1 \leq_d z_3$ sera vraie puisqu'aucune distance euclédienne entre z_3 et les autres solutions n'est plus petite que la distance euclédienne entre z_1 et z_2. Afin d'obtenir une population d'élites de taille $\overline{N}$, la procédure de mise à jour de l'archive (algorithme 10) éliminera les solutions qui possèdent la plus petite distance euclédienne jusqu'à ce que la taille de $\mathcal{Q}$ soit égale à $\overline{N}$. En se servant cette fois-ci de la figure 8 qui présente un cas hypothétique où la taille de $\mathcal{Q}$ doit être égale à quatre, on montre que suite à l'application de procédure de mise à jour, la solution z_2 sera retirée, puis ensuite la solution z_4.

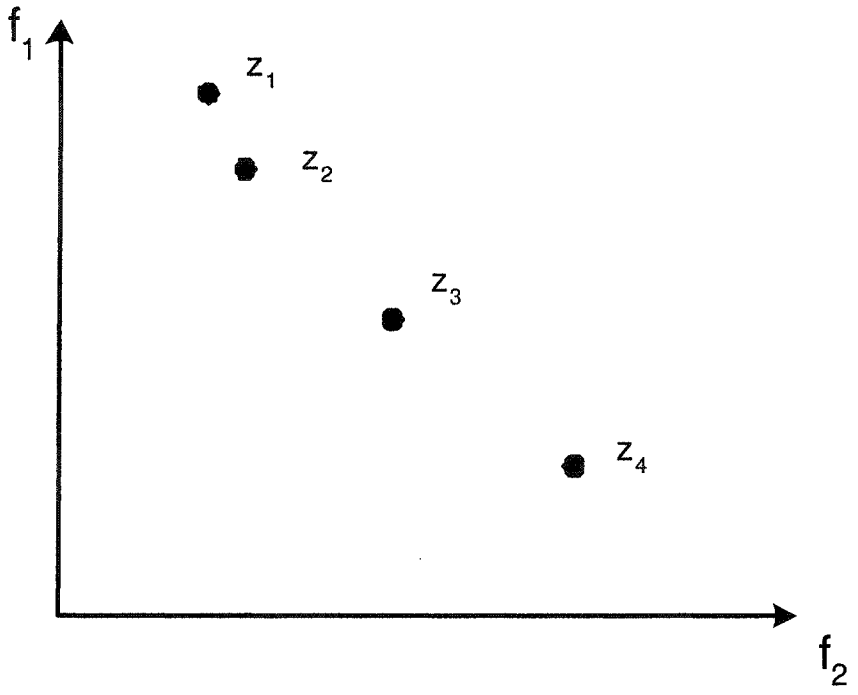

Figure 7 Cas hypothétique de l'assignation de la mesure de distance (SPEA-II)

Définition 3.2 *Soit un ensemble de solutions non dominées $\mathcal{Q}$, la relation $\mathbf{z}_i \leq_d \mathbf{z}_j$ s'applique entre $\{\mathbf{z}_i, \mathbf{z}_j\} \in \mathcal{Q}$ si :*

1. $\forall\, k,\, 0 < k < \|\mathcal{Q}\|,\, \sigma_i^k = \sigma_j^k$

   OU

2. $\exists\, k,\, 0 < k < \|\mathcal{Q}\|, \forall\, l,\, 0 < l < k,\, \sigma_i^l = \sigma_j^k \wedge \sigma_i^k < \sigma_j^k$

Zitzler [13] conclut que le SPEA-II (algorithme 11) donne de meilleurs résultats que son ancêtre le SPEA. Cette différence vient essentiellement de l'assignation de la valeur d'adaptation qui s'effectue dans les deux populations $\mathcal{Q}$ et $\mathcal{P}$ selon la dominance. L'auteur a effectué plusieurs expérimentations et conclut qu'en général le NSGA-II et le SPEA-II donnent en moyenne des résultats semblables. Par contre, le SPEA-II semble donner de meilleurs résultats lorsque le nombre de fonctions d'objectifs est plus élevé.

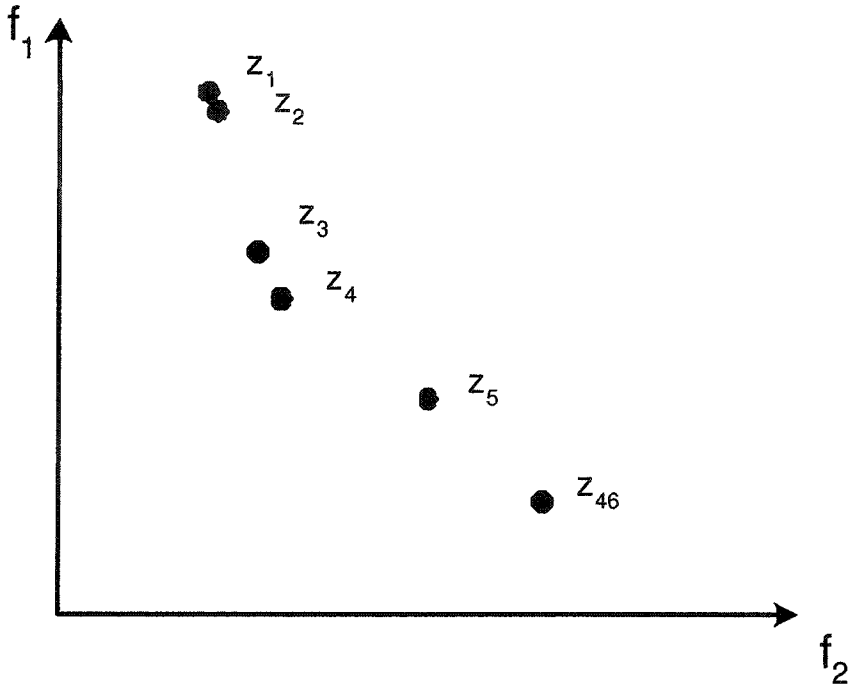

Figure 8 Application de l'opérateur de troncation (SPEA-II)

Cet algorithme sera donc le dernier présenté. Avant de passer au chapitre suivant, quelques concepts importants reliés aux algorithmes évolutifs multi-critères seront présentés.

ALGORITHME 11

SPEA-II [13]

Données :
M : Opérateur de mutation.
N : Taille de la population $\mathcal{Q}$.
$\mathcal{P}$: Population d'individus.
$\mathcal{Q}$: Population des élites.
R^2 : Opérateur de mise à jour de l'archive.
S : Opérateur de sélection.
X : Opérateur de croisement.
ω_i : Valeur d'adaptation de la solution i.

début
Initialiser $\mathcal{P}_0$ de taille N et $\mathcal{Q} = \emptyset$
pour *chaque itération* t **faire**
Calculer ω_i pour chaque $i \in \mathcal{P}_t$
Calculer ω_j pour chaque $j \in \mathcal{Q}_t$
Initialiser $\mathcal{Q}_{t+1} = \mathbf{z}_i \mid \omega_i < 1 : \mathbf{z}_i \in \{\mathcal{Q}_t \cup \mathcal{P}_t\}$
Appliquer $\mathcal{Q}_{t+1} = \mathrm{R}^2(\mathcal{Q}_{t+1})$
si *Si le critère d'arrêt est atteint* **alors**
Retourner $\mathcal{Q}_{t+1}$
sinon
$\mathcal{P}_{t+1} = \mathrm{M}(\mathrm{X}(\mathrm{S}(\mathcal{Q}_t \cup \mathcal{P}_t)))$
fin

3.3.5 Algorithmes évolutifs multi-critères sous contraintes

Dans certains problèmes tel que le problème P2, les fonctions d'objectifs sont soumises à des contraintes fortes qui peuvent être soit de type égalité ou inégalité. Dans le problème P2, toutes les solutions contenant des conflits d'horaire doivent être rejetées. Il existe de nombreuses méthodes servant à la gestion des contraintes. Entre autre, il est possible d'imposer une pénalité forte à toute solution non réalisable. Le désavantage de cette méthode est l'ajout de poids qui demande une phase d'ajustement qui peut s'avérer très difficile. Récemment, Deb et ses élèves [9] ont proposé une nouvelle approche basée sur la définition du principe de dominance.

Définition 3.3 *Soit deux solutions* $\mathbf{z}^{(1)}, \mathbf{z}^{(2)} \in O$, *le* ***principe de dominance sous contraintes*** *représenté par l'opérateur* $\succcurlyeq_c$ *tel que* $\mathbf{z}^{(1)} \succcurlyeq_c \mathbf{z}^{(2)}$ *signifie que la solution* $\mathbf{z}^{(1)}$ domine sous contrainte *la solution* $\mathbf{z}^{(2)}$ *dans l'espace d'objectif si* $\exists\ \mathbf{z}^{(1)}$ *et* $\mathbf{z}^{(2)}$ *tel que :*

1. $\mathbf{z}^{(1)}$ *est réalisable et* $\mathbf{z}^{(2)}$ *est irréalisable, OU*
2. $\mathbf{z}^{(1)}$ *est irréalisable et* $\mathbf{z}^{(2)}$ *est irréalisable mais* $\mathbf{z}^{(1)}$ *obtient la plus petite violation, OU*
3. $\mathbf{z}^{(1)}$ *est réalisable et* $\mathbf{z}^{(2)}$ *est réalisable mais* $\mathbf{z}^{(1)} \succcurlyeq \mathbf{z}^{(2)}$

Avec ce principe de dominance, l'opérateur de sélection de la définition 3.1 peut être appliquée. Comme les ensembles de Pareto seront obtenus à l'aide du principe de dominance sous contraintes, Deb et *al.* ont donné un nom différent à cet opérateur de sélection : *Crowded Constrained Tournament.* Aucun paramètre additionnel n'est requis. De plus, avec la réalisation de ce tournoi, toute solution irréalisable sera toujours dominée par une solution réalisable.

3.4 Application à la création des horaires d'examens

Cette section débutera par la présentation du modèle qui sera utilisé. Par la suite, la modélisation de ce problème sera effectuée afin de rendre possible l'utilisation des algorithmes proposés. Cette section se terminera par une présentation des paramètres de simulation utilisés, des résultats obtenus et d'une discussion.

3.4.1 Modèle utilisé

Il existe quatre types de problèmes qui ont été suggérés dans la littérature soit le P1, P2, P3 et P4. Ces problèmes ont été décrits dans la section 2.4. Par contre, aucun modèle multi-critères n'a été proposé. Afin de permettre à la fois la comparaison des résultats et l'utilisation d'un modèle multi-critères, les problèmes P1 et P2 seront fusionnés afin de créer un nouveau modèle P1-P2 :

Problème 3.1 *Le problème multi-critères P1-P2 est défini par :*

$$\begin{aligned}
&\min \quad f_1 = \|\mathcal{T}\|\,; \\
&\min \quad f_2 = \tfrac{1}{2}\sum_{t=1}^{5} w_t \sum_{k=1}^{\|\mathcal{T}\|-t}\sum_{i=1}^{\|\mathcal{E}\|}\sum_{j=1}^{\|\mathcal{E}\|}\left(\eta_{ij}\epsilon_{ik}\epsilon_{jk+t}\right)\,; \\
&s.a. \quad \sum_{k=1}^{\|\mathcal{T}\|}\sum_{i=1}^{\|\mathcal{E}\|}\sum_{j=1}^{\|\mathcal{E}\|}\eta_{ij}\epsilon_{ik}\epsilon_{jk} = 0.
\end{aligned}$$

où $\mathcal{T}$ représente l'ensemble des périodes de l'horaire, $\mathcal{E}$ définit l'ensemble des examens, w_t = {1,2,4,8,16}, η_{ij} est le nombre d'élèves en commun entre l'examen i et j et ϵ_{jk} = 1 si l'examen j est assigné à période k, sinon ϵ_{jk} = 0

La première fonction d'objectif du problème P1-P2 (problème 3.1) détermine le nombre de périodes qui composent l'horaire. La deuxième fonction d'objectif quant à elle indique le nombre d'élèves ayant $t-1$ périodes libres entre deux examens, t =1...5. La seule contrainte active dans ce modèle stipule que le nombre d'élèves ayant deux examens durant la même période doit être nul.

3.4.2 Représentation des solutions

La façon la plus simple de représenter une solution ou un horaire d'examens, est d'utiliser un tableau unidimensionel où chaque élément représente un examen et la valeur associée à chaque élément est la période où l'examen est assigné. Comme le montre la figure 9, cette méthode permet aussi l'encodage direct d'un chromosome ce qui a pour avantage de pouvoir utiliser les opérateurs génétiques tels que le croisement et la mutation.

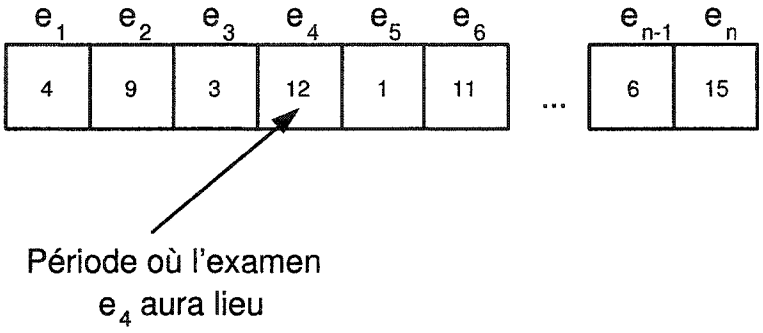

Figure 9 Représentation des solutions

3.4.3 Application de NSGA-II et SPEA-II

Le premier essai effectué avec les algorithmes NSGA-II et SPEA-II a été réalisé en utilisant les paramètres suivants :

a. Taille de la population = 40

b. Nombre d'itérations : 50

c. Croisement uniforme

d. Taux de croisement = 50%

e. Mutation aléatoire

f. Taux de mutation = 1 / L (L = nombre d'examens)

g. Base de données *hec-s-92*

h. Domaine de f_1 entre 17 et 26 périodes

La figure 10 montre la population initiale ainsi que la population finale de l'algorithme. Il est clair que cette implantation présente un problème de diversité au niveau de la première fonction d'objectif (figure 10). Cette lacune vient principalement du fait qu'aucun élément n'assure la diversité au niveau du nombre de périodes des horaires. Comme le montre la figure 11, lorsque deux horaires de longueur différente se croisent, il arrive dans certain cas que la longueur devient identique pour les deux horaires. Ce phénomène semble être très influent puisque lorsque l'algorithme termine, toutes les solutions possèdent le même nombre de périodes. Une modification doit être apportée pour prévenir ce problème.

3.4.4 Amélioration de la diversité

Plusieurs possibilités peuvent être envisagées afin de garder une certaine diversité au niveau des différentes valeurs de f_1. Tout d'abord, il est possible de restreindre le croisement à des solutions possédant la même valeur de f_1. Le problème avec cette approche est que les échanges génétiques n'étant possibles qu'entre des petits groupes formés d'individus à l'intérieur de la population risque de faire converger prématurément l'algorithme. Une autre solution envisagable est d'ajouter à chaque chromosome un gène de contrôle. Ce gène enregistrera le nombre de périodes dans l'horaire et la valeur des gènes sera contrôlée par ce dernier. Plus précisément, en posant $\mathcal{E}$ comme étant l'ensemble des examens et $\mathcal{T}$ l'ensemble des périodes, le modèle de solution se définit par :

$$Z_1 = \|\mathcal{T}\| \tag{3.6}$$

$$Z_i = p_j \quad j \in \mathcal{T}, \quad i = 2 \ldots \|\mathcal{E}\| + 1 \tag{3.7}$$

Afin de permettre l'application de l'opérateur de croisement sur des solutions possédant des valeurs de f_1 différentes sans interférer avec le gène de contrôle, une heuristique

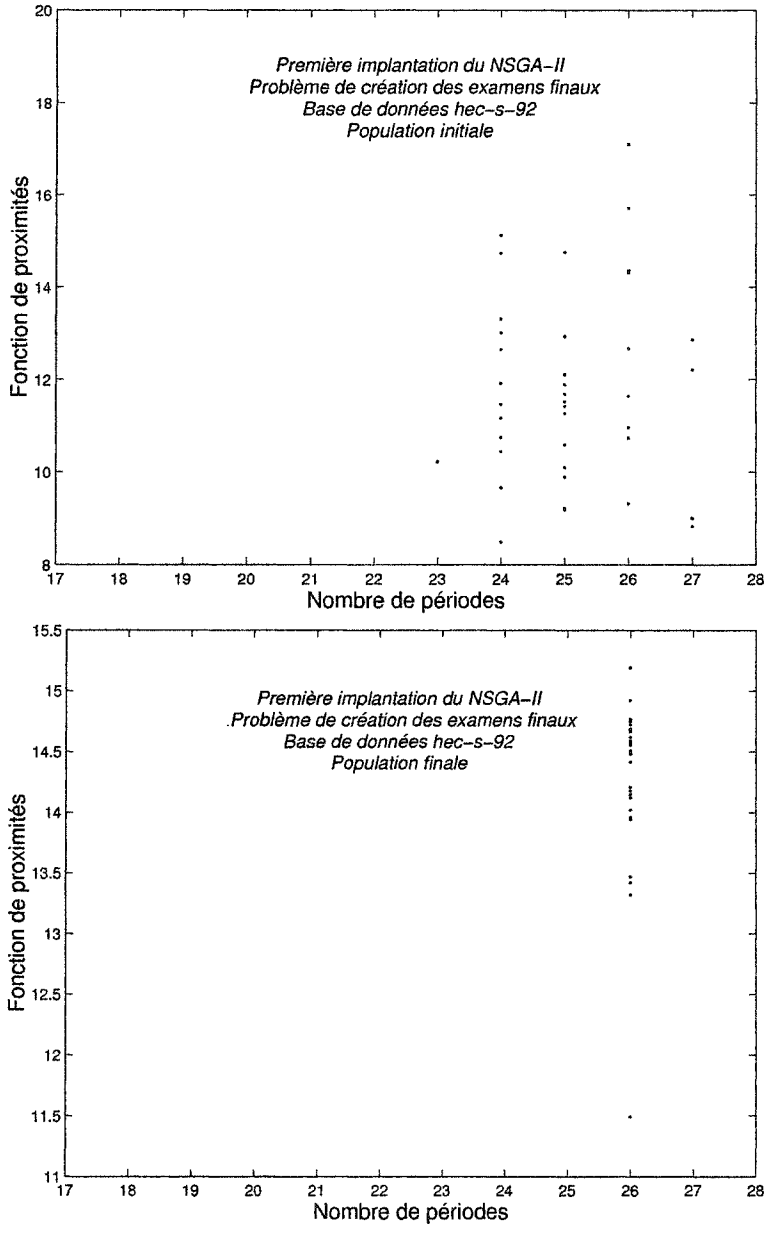

Figure 10 Premier essai de création d'un horaire d'examens

d'ajustement doit être ajoutée. La façon la plus simple d'ajuster un chromosome est de parcourir tous ces gènes et de réassigner les examens affectés à des périodes en dehors de la valeur du gène de contrôle. Cette réassignation s'effectue en minimisant l'impact au niveau des proximités. Avec cette heuristique, il sera même possible d'injecter une légère

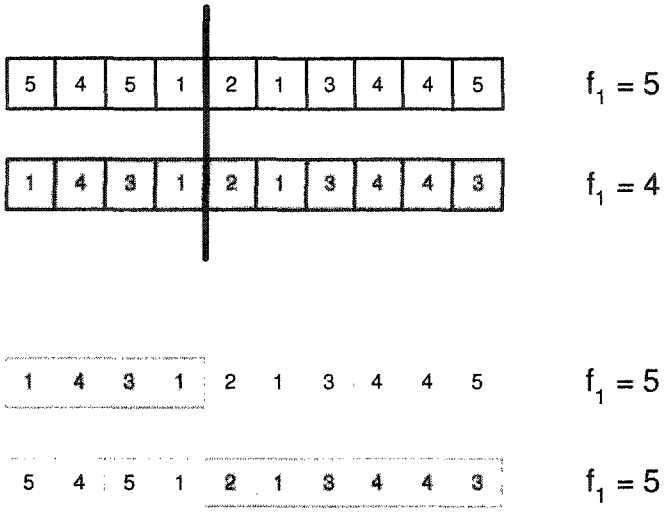

Figure 11 Croisement de deux horaires de longueur différente

mutation sur le gène de contrôle afin d'augmenter l'espace de recherche. La prochaine implantation sera donc effectuée en intégrant ces modifications et en utilisant les mêmes paramètres soit :

a. Taille de la population = 40

b. Nombre d'itérations : 50

c. Croisement uniforme

d. Taux de croisement = 50%

e. Mutation aléatoire

f. Taux de mutation = 1 / L (L = nombre d'examens)

g. Base de données *hec-s-92*

h. Domaine de f_1 entre 17 et 26 périodes

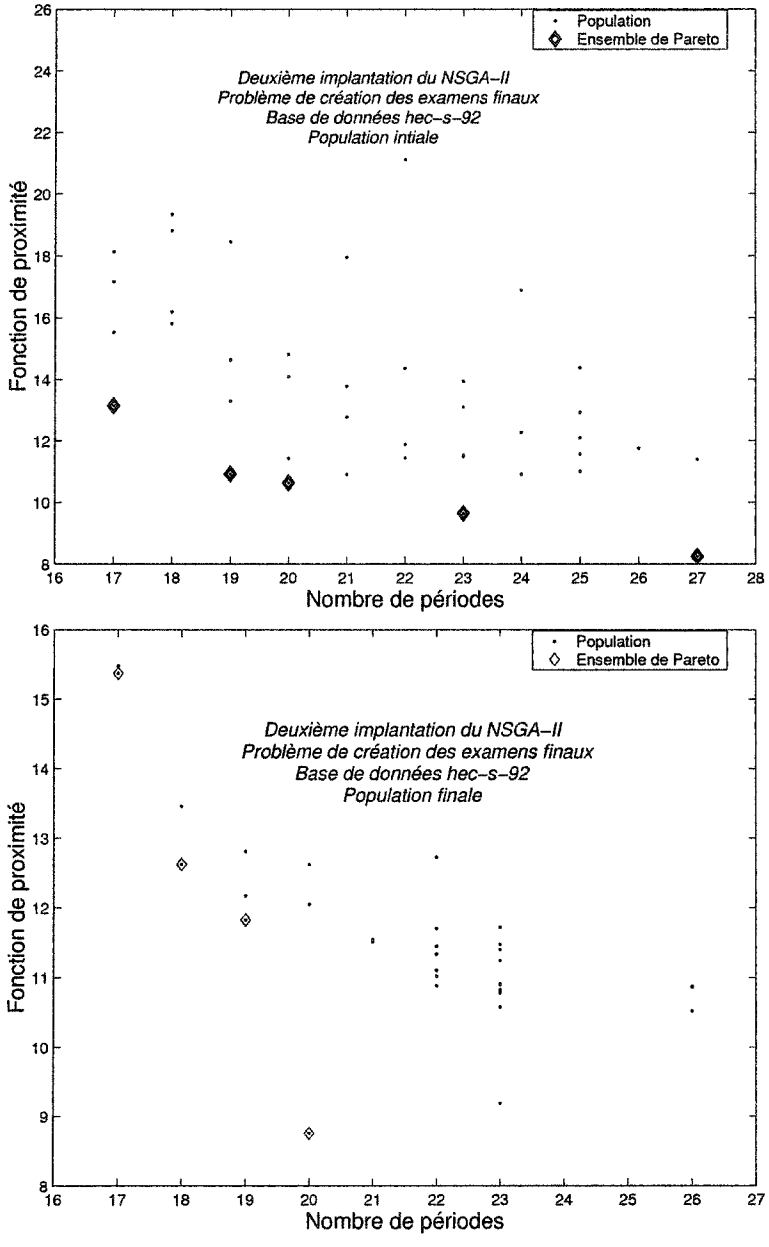

Figure 12 Deuxième essai de création d'un horaire d'examens

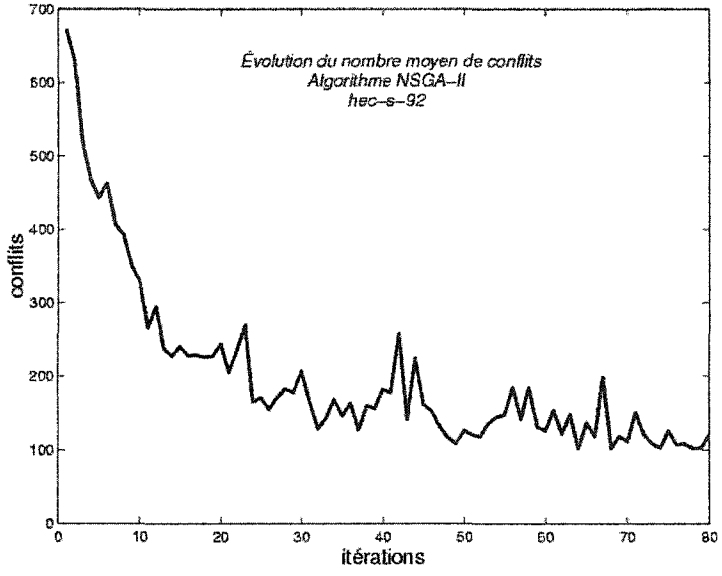

Figure 13 Évolution du nombre moyen de conflits lors du deuxième essai

Comme le montre la figure 12, représentant la population initiale et finale, l'ajout du gène de contrôle améliore la diversité au niveau de l'ensemble de Pareto final. Malgré tout, cette diversité n'est pas encore parfaite. Le problème majeur rencontré, qui était aussi présent lors du premier essai, est le nombre de conflits. Toutes les solutions produites présentent des conflits d'horaire. La figure 13 montre l'évolution du nombre moyen de conflits dans la population. Le *Crowded Constraint Tournament* ne semble pas assez efficace pour gérer à lui seul cette contrainte forte. De plus, comme toutes les solutions sont conflictuelles, avec l'utilisation du *Crowded Constraint Tournament* le nombre de conflits sera le seul facteur qui est optimisé. La figure 12 montre bien que le taux de proximités n'a pas diminué entre la population initiale et la population finale. Encore une fois une modification s'impose.

3.4.5 Minimisation des conflits

Dans la section 1.7, différentes techniques d'initialisation des solutions étaient présentées. Ces techniques permettent de créer des solutions initiales en minimisant le nombre de

conflits. La technique la plus performante était celle de l'assignation des examens par ordre de périodes disponibles (*D-Satur*). Le troisième essai consiste donc à initialiser toutes les solutions avec cette heuristique.

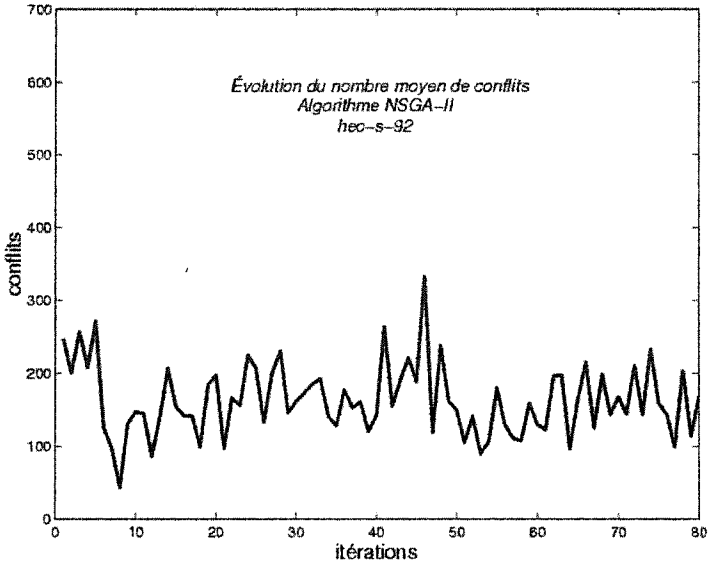

Figure 14 Évolution du nombre moyen de conflits lors du troisième essai

La figure 14 montre l'évolution du nombre moyen de conflits avec l'utilisation de l'initialisation des solutions selon l'assignation *D-Satur* [29]. Encore une fois l'algorithme n'arrive pas à réduire le nombre de conflits. Le seul impact notable est la disparition de la zone de transition entre la première et la dixième itération (voir figure 13, page 69).

3.4.6 Discussion

Les essais effectués précédemment montrent que les algorithmes NSGA-II et SPEA-II ne sont pas adéquats pour résoudre les problèmes de création des horaires d'examens. Tout d'abord la gestion des conflits d'horaire ne peut être traitée en utilisant uniquement le croisement et l'assignation de la valeur d'adaptation. Malgré la pression exercée par le

Crowded Constraint Tournament, des conflits sont toujours présents après avoir exécuté les algorithmes NSGA-II ou SPEA-II.

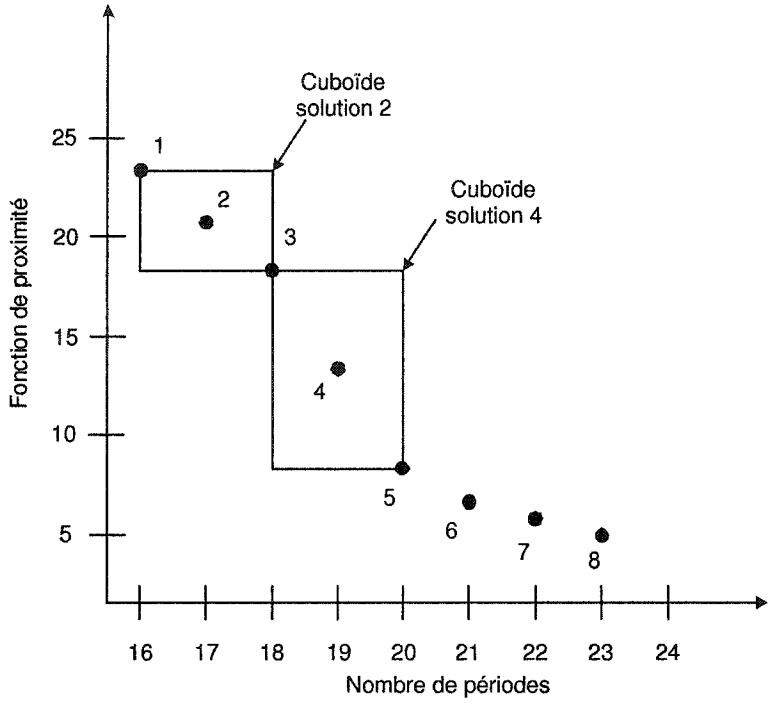

Figure 15 Cas hypothétique du calcul de la *Crowded Distance* pour un PCHE

Un autre problème important est la gestion de la diversité dans les ensembles de Pareto. Étant donné que l'ensemble de Pareto est discret et que le domaine de la fonction f_1 représentant le nombre de périodes est restreint, l'ensemble de Pareto final devrait posséder une solution pour chaque valeur de f_1. Les résultats ont montré qu'en aucun cas l'ensemble de Pareto final obtenu ne respectait cette affirmation. Le problème vient de la manière dont la diversité est traitée dans les algorithmes. Pour le NSGA-II la diversité s'évalue par la *Crowded Distance*. Cette distance, comme il a été présenté à la section 3.3.2, représente le demi périmètre dans l'espace d'objectif formé par le cuboïde reliant les solutions voisines à une solution dont il faut calculer la distance. Le problème dans le modèle utilisé est que la distance entre deux valeurs de f_1 consécutives est toujours la même. La *Crowded Distance* sera alors calculée uniquement par rapport à f_2. La figure 15 illustre ce phéno-

mène. Dans cet exemple hypothétique, la solution 2 possèderait une *Crowded Distance* plus faible que la solution 4 et, par le fait même, une chance plus faible de survie. Par contre, la solution 2 est aussi importante que la solution 4 pour la définition de l'ensemble de Pareto. Un phénomème similaire apparaît lors du calcul de la valeur d'adaptation des solutions dans l'algorithme SPEA-II. Encore une fois la distance n'aura de différence que sur la valeur de f_2 et des solutions importantes dans la définition de l'ensemble de Pareto pourront être exclues.

Une autre remarque intéressante est que le nombre de solutions dans les ensembles de Pareto sera restreint automatiquement par le domaine de la fonction f_1. En supposant que la diversité au niveau des ensembles de Pareto est parfaite, il ne pourra pas y avoir plus de solutions dans un ensemble que d'éléments dans le domaine de f_1. Concrètement, en prenant l'exemple de la base de données *hec-s-92*, dont le nombre de périodes minimales ne peut probablement pas être inférieur à 17. Cette valeur est uniquement basée sur des résultats expérimentaux. Un résumé des valeurs obtenues pour l'optimisation du nombre de périodes est disponible, entre autres, dans le papier de Merlot et al. [5]. et en fixant la borne supérieur à 27 périodes, le domaine de f_1 ne présentera que 11 valeurs. Dans ce cas précis les ensembles de Pareto ne pourront pas posséder plus de 11 solutions différentes. En augmentant alors la taille de la population, le nombre d'ensembles de Pareto augmentera. Ce nombre peut devenir potentiellement très élevé et les chances que des solutions non dominées se croisent avec des solutions dominées sont très élevées. Pour toutes ces raisons l'études des algorithmes SPEA-II et NSGA-II est en partie abandonnée.

3.5 Conclusion

Suite à plusieurs tentatives infructueuses dans le but de résoudre un PCHE, les méthodes évolutives multi-critères strictement basées sur les algorithmes génétiques, telles que NSGA-II et SPEA-II, ont été abandonnées. Ces méthodes sont inefficaces pour la résolution d'un PCHE multi-critères où l'on désire minimiser la longueur des horaires ainsi que les proxi-

mités. Les résultats obtenus ne peuvent être comparés à ceux publiés, puisque les algorithmes n'arrivent pas à trouver de solutions réalisables. Il sera donc nécessaire de réviser la méthodologie utilisée pour arriver à obtenir un AEMC capable de bien performer sur un problème de création des horaires d'examens. Les points suivants devront guider la conception d'un éventuel algorithme :

1. Trouver un opérateur autre que le croisement capable d'assurer une convergence de manière à obtenir des résultats comparables à ceux publiés ;
2. Déterminer une technique fiable pour la gestion des conflits d'horaire ;
3. Posséder un opérateur ou quelconque heuristique capable d'assurer une diversité parfaite au niveau des solutions composant les ensembles de Pareto.

Le prochain chapitre porte sur la conception d'un nouvel AEMC capable de répondre aux exigences mentionnées ci-haut.

CHAPITRE 4

CONCEPTION D'UN ALGORITHME ÉVOLUTIF HYBRIDE

4.1 Introduction

Puisque les algorithmes NSGA-II et SPEA-II ont tous deux échoués lors de la résolution d'un PCHE multi-critères, la conception d'un nouvel AEMC peut être envisagée. Ce chapitre présente donc les étapes nécessaires à l'élaboration d'un tel algorithme. Pour commencer, la structure générale de l'algorithme et de ses principales composantes seront présentées. Par la suite, l'ensemble des éléments utilisés pour arriver à la conception de l'algorithme seront vues en détails. Une série d'expérimentations permettra de justifier les opérateurs choisis et d'ajuster leurs paramètres. La première application réalisée porte sur le modèle mutli-critère P1-P2 vu au chapitre précédent. Une deuxième application sera effectuée sur un nouveau modèle, soit le P1-P3 qui tient compte de la capacité des locaux. Contrairement au NSGA-II et SPEA-II, l'algorithme présentera de bonnes performances. La comparaison des résultats obtenus avec ceux déjà publiés mettront en évidence la pertinence de l'algorithme conçu. Les expérimentations effectuées avec les algorithmes NSGA-II et SPEA-II n'auront donc pas été faites en vain puisqu'elles auront permis d'identifier les faiblesses d'un algorithme évolutif appliqué sur un PCHE multi-critères et ainsi guider la conception du nouvel algorithme.

4.2 Présentation de l'algorithme

La structure de l'algorithme présentée provient de nombreux essais infructueux ainsi que des observations faites sur le comportement des algorithme NSGA-II et SPEA-II. Seule la structure de l'algorithme final sera présentée. Évidemment le choix des opérateurs et des paramètres sera jusitifé.

4.2.1 Structure générale

La figure 16 montre les étapes de l'algorithme évolutif. Tout d'abord l'application de deux fouilles locales sont requises, d'une part pour diminuer les conflits d'horaire et d'autre part pour traiter les proximités. Ensuite un calcul servant à l'assignation de la valeur d'adaptation est entrepris. Une stratégie d'archivage est ensuite utilisée afin de conserver les meilleures solutions à chaque itération et un opérateur de sélection permet de construire la prochaine population. L'exploration des solutions est assurée par l'application de l'opérateur de mutation. Les solutions mutées serviront donc de nouveaux points de départ pour les fouilles locales lors de la prochaine itération.

Le résultat de l'algorithme sera l'obtention d'une population consitutée d'un ensemble de solutions non dominées. De cette façon, il est possible à la fois d'exploiter les forces d'une fouille locale combinée aux avantages d'un AEMC. L'algorithme peut être vu aussi comme un algorithme génétique multi-critères où le croisement est remplacé par deux opérateurs de fouille locale donc, un *memetic algorithm* multi-critères.

4.2.2 Justification de la structure

Comme l'algorithme proposé est un AEMC il doit, par définition, utiliser un ensemble de solutions pour évoluer. Cette population doit obligatoirement être initialisée dès le départ. La méthode d'initialisation sera justifiée plus tard. La structure générale d'un algorithme évolutif est constituée par l'application d'une série d'opérateurs sur un ensemble de solutions. Cette structure sera donc conservée. Afin de permettre une meilleure exploration des solutions, le croisement sera remplacé par une fouille locale dont la nature sera justifée plus loin. Ce remplacement s'inscrit dans l'optique des travaux de Burke et *al.* [28][19] qui sont les seuls à avoir donné de très bons résultats sur les bases de données publiques en utilisant une méthode évolutive. Ces derniers ont démontré que l'ajout d'une fouille locale permet d'augmenter grandement les performances d'un algorithme évolutif appliqué à l'ordonnancement des examens.

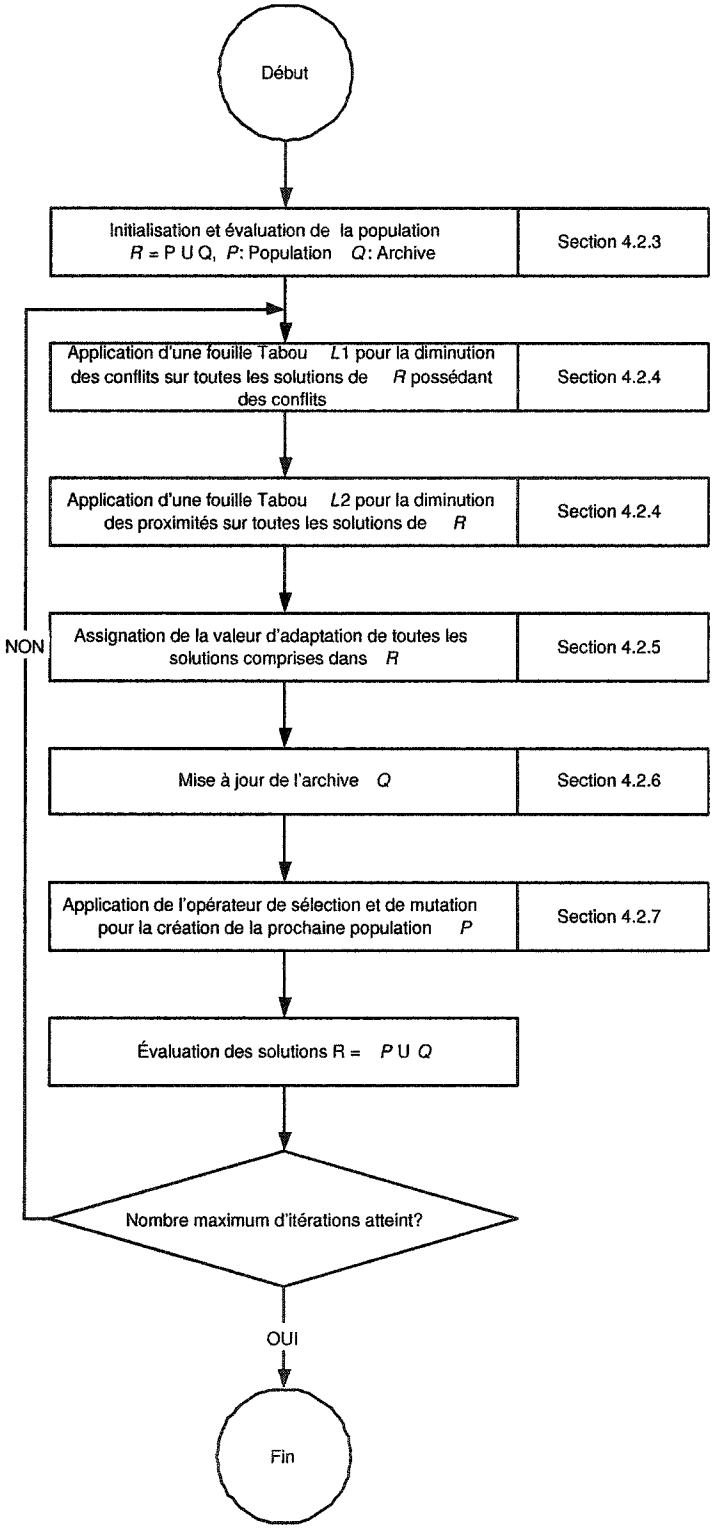

Figure 16 Ordinogramme de l'algorithme évolutif hybride

Une deuxième fouille locale a été ajoutée afin de permettre la réparation des solutions non réalisables. Lors des essais précédents, l'opérateur de sélection s'est avéré inéfficace pour la gestion des conflits d'horaire. Donc pour palier à ce problème, en plus d'utiliser le principe de dominance sous contraintes ainsi que la sélection, une deuxième fouille locale sera ajoutée.

Un autre opérateur important d'un algorithme évolutif est l'élitisme. Bon nombre de travaux font l'éloge de l'utilisation de cet opérateur [9] [13]. C'est donc pourquoi l'algorithme utilisera, tout comme le SPEA-II, une population externe servant à enregister les meilleures solutions obtenues. Une procédure de mise à jour de cette archive devra donc être ajoutée. Pour terminer, l'opérateur de mutation sera toujours présent afin de permettre une plus grande exploration de l'espace de désicion.

4.2.3 Méthode d'initialisation des solutions

Les résultats des travaux de Carter montrent que la technique d'assignation par *Saturation Degree* (SD) assure une convegence rapide au niveau des conflits d'horaire. Par contre, aucun test n'a été effectué quant à l'influence de ces techniques sur les proximités. Le tableau VI montre donc l'impact sur les proximités de l'utilisation des techniques RO et SD sur l'application de la fouille Tabou.

Tableau VI

Influence de la méthode d'initialisation des solutions. Essais effectués sur *hec-s-92*. Nombre de période = 18

Technique	moyenne	écart type	itération
RO	**10.475**	0.091	4
SD	10.636	0.142	1

Afin de bien cerner l'influence de ces techniques, 10 essais ont été effectués sur la base de données *hec-s-92*. Cette base de données a été choisie pour sa petite taille, ce qui permet d'avoir plus rapidement des résultats sur la sensibilité des paramètres. La deuxième et la troisième colonne du tableau VI donnent la moyenne et l'écart type de la fonction de proximités utilisée dans le modèle P2. La dernière colonne de ce tableau donne le nombre moyen d'itérations nécessaire pour obtenir une solution réalisable. Les résultats montrent que l'initialisation par assignation aléatoire présente le taux de proximités le plus faible. Ce phénomène peut s'expliquer par le fait que l'utilisation d'une technique telle que SD entraîne, dès le début, les solutions dans une région de l'espace caractérisé par une faible présence de conflits. C'est pourquoi l'utilisation d'une technique d'initialisation aléatoire de la population permet une meilleure exploration de l'espace de décision. De plus comme le montre le tableau VI, la fouille Tabou appliquée sur les conflits est amplement suffisante pour gérer cette contrainte, car même sans l'utilisation du SD, une solution sans conflit ayant dix-huit périodes est trouvée après 4 itérations. Pour ces raisons la technique d'initialisation retenue sera par assignation aléatoire.

4.2.4 Fouilles locales

Comme il a été mentionné dans les sections précédentes, l'algorithme proposé nécessite l'utilisation de deux fouilles locales. La première pour la minimisation du nombre de conflits et la deuxième pour la minimisation du nombre de proximités.

4.2.4.1 Choix de la méthode

Le choix de la technique utilisée pour effectuer les fouilles locales est basé sur la structure même de l'algorithme évolutif. Les deux fouilles locales qui sont le plus citées sont sans aucun doute la fouille Tabou et le recuit simulé. Dans plusieurs cas d'optimisation d'horaire d'examens, ces techniques ont fait leurs preuves. Pour cette raison ces deux techniques seront retenues. Il est important que la fouille locale soit en mesure de continuer la recherche là où elle l'avait terminée lors de la dernière itération. Dans le cas contraire

tous les efforts effectués lors des itérations subséquentes seraient inutiles. Dans cette optique le choix d'un recuit simulé est rejeté. L'une des caractéristiques du recuit simulé est la phase initiale. Dans cette phase une température élevée permet de couvrir une région plus vaste de l'espace de décision. Ceci a comme conséquence d'effectuer une exploration quasi aléatoire lors des premières époques de l'algorithme. La méthode de la fouille Tabou ne crée pas ce genre de problème, c'est pourquoi elle sera retenue. Le choix des paramètres des fouilles Tabou sera déterminé à la section suivante.

4.2.4.2 Type de voisinage et échantillonage pour les fouilles Tabou

Dans l'algorithme évolutif proposé, étant donné que l'opérateur de croisement a été retiré, l'application de la fouille Tabou devra assurer la convergence de la population. L'ajustement des paramètres de cette fouille aura donc une influence marquante sur la qualité des solutions obtenues. Le voisinage utilisé pour la fouille Tabou servant à diminuer le nombre de conflits est facile à determiner. Une solution voisine est obtenue suite au déplacement d'un examen qui crée des conflits d'horaire dans une autre période. Le voisinage de la deuxième fouille Tabou servant à l'optimisation des proximités est plus délicat. La difficulté majeure lors de ce voisinage est de déterminer une technique capable de diminuer les proximités tout en respectant les conflits d'horaire. Dans la littérature plusieurs méthodes sont proposées. Les trois techniques les plus fréquemment utilisées, soit par échange d'examens, échange de périodes et chaîne de Kempe seront présentées à la section 1.6.4 et feront l'objet d'une étude comparative.

Une autre approche intéressante est de combiner différentes techniques de voisinage. Par exemple la technique par chaîne de Kempe, qui permet une convergence rapide, peut être combinée à un voisinage par déplacement d'examens qui permet un meilleur raffinement. Deux façons de procéder au voisinage multiple seront évaluées. Premièrement, un seul changement est fait, entre chaîne de Kempe et déplacement d'un examen, après 100 itérations sans amélioration de la fonction d'objectif. Deuxièment, une alternance entre chaîne

de Kempe et déplacement d'un examen est effectuée. Plus précisément le voisinage par chaîne de Kempe est d'abord lancé. Après 100 itérations sans amélioration de la fonction d'objectif on procède par déplacement d'un examen. Si avant 100 itérations une amélioration est produite le retour au voisinage par chaîne de Kempe s'effectue. Le cycle s'arrête dès que 100 itérations par déplacement d'un examen ont été faites sans amélioration de la fonction d'objectif.

Tableau VII

Influence des différentes techniques de voisinages choisies. Essai effectué sur *hec-s-92*. Nombre de périodes = 17

Technique	**moyenne**	**écart type**
Échange d'examens	14.8211	0.3537
Échange de périodes	13.2434	0.6456
Chaîne de Kempe	13.0233	0.1354
Voisinage multiple 1	12.7906	0.1644
Voisinage multiple 2	**12.039**	0.0993

Une expérimentation a été faite afin de déterminer le type de voisinage le plus adéquat. Pour y arriver une fouille Tabou a été utilisée. La *tenure* de cette fouille est fixée à 25 itérations. La fouille progresse jusqu'à ce que la valeur de la fonction d'objectif reste inchangée pendant 500 itérations. La solution de départ est initialisée de façon à n'avoir aucun conflit et le critère d'optimisation est la fonction de proximités du modèle P2. Le tableau VII présente les résultats obtenus suite à l'application des 5 combinaisons de voisinages énumérées ci-haut. L'expérimentation a été faite à l'aide de la base de données *hec-s-92*. Le tableau présente les meilleures solutions trouvées ayant 17 périodes. Dix essais ont été lancés pour chaque technique. Il est clair que la technique par alternance entre les différents types de voisinage est de loin la plus intéressante. Le passage entre le voisinage par chaîne de Kempe et le déplacement d'un examen agit comme une perturbation dans le système ce qui permet à l'algorithme d'évoluer davantage.

4.2.5 Valeur d'adaptation et sélection des solutions

Dans la littérature, il existe un grand nombre de techniques servant à l'assignation de la valeur d'adaptation. Dans le cadre de ce projet, les techniques basées sur le tri par ensembles de Pareto (NSGA-II) et par mesure de force (SPEA-II) ont été implantées. Comme il a été vu lors du chapitre précédent, un des problèmes qui a été relevé pour l'algorithme NSGA-II est le nombre d'ensembles de Pareto obtenus. Comme le domaine de la fonction de la minimisation du nombre de périodes est discret et très restreint, le nombre de solutions possibles par ensemble de Pareto est faible. On a donc remarqué que le nombre d'ensembles de Pareto peut potentiellement être très élevé et l'assignation de la valeur d'adaptation par les ensembles de Pareto n'est pas adéquat pour ce type de fonction. Pour cette raison, le calcul de la valeur d'adaptation sera celle utilisée par l'algorithme SPEA-II.

Pour terminer l'étude des opérateurs génétiques utilisés, un choix doit être fait conscernant la méthode de sélection. Le problème à résoudre est sujet à une contrainte soit l'imposition d'un nombre de conflit nul. Dans cette optique, l'opérateur de sélection utilisé sera le tournoi binaire basé sur la valeur d'adaptation du SPEA-II. Pour calculer la valeur d'adaptation le principe de dominance sous contraintes utilisé lors des essais du chapitre précédent sera utilisé.

4.2.6 Mise à jour de l'archive et traitement de la diversité

La mise à jour de l'archive est une opération essentielle. L'archive a comme rôle de garder en mémoire les meilleures solutions obtenues. Dans le cadre de la conception de l'algorithme évolutif multi-critères hybride (AEMH), la mise à jour de l'archive assurera en plus la diversité des solutions. Le problème de la diversité a été soulevé lors des essais faits dans le chapitre précédent. Généralement la diversité est contrôlée par différentes techniques basées sur des mesures de distance. Dans le modèle proposé, la première fonction d'objectif est notée par $f_1 = \{p_1, p_2, p_3, \ldots, p_i\}\ p_i \in \mathcal{N}^+$ où p_i représente le nombre

de périodes compris dans l'horaire d'examens. Le domaine de cette fonction d'objectif est restreint à quelques éléments, ceci rend inéficace les mesures de distances puisque l'objectif est de maintenir des solutions pour chacune des valeurs du domaine de f_1. Ainsi, pour chaque valeur du domaine de f_1 l'archive initiale Q_0 doit être construite selon la définition 4.1.

Définition 4.1 *Soit* $D = \{p_1, p_2, p_3, \ldots, p_i\}$ $p_i \in \mathcal{N}^+$, *le domaine de la fonction d'objectif* f_1, $\mathcal{Q}_0 = \{q_1 \cup q_2 \cup q_3 \ldots \cup q_i\}$, *l'archive initiale où* $q_i = \{x_1^i, x_1^i, \ldots, x_j^i\}$ *représente les solutions initiales du sous-groupe* i *de l'archive, l'initialisation de chaque sous-groupe* $q_i \in \mathcal{Q}_0$ *doit être faite de façon à ce que* $f_1(x) = p_i$, $\forall\, x \in q_i$, $q_i \in \mathcal{Q}_0$.

Avec l'initialisation de l'archive, il est alors possible de conserver des solutions pour chaque valeur du domaine de f_1 en appliquant la règle suivante :

Définition 4.2 *Soit* Q_{t+1}, *l'archive à l'itération* $t+1$, R_t *l'ensemble des solutions obtenues suite à l'application des fouilles locales à l'itération* t *et* ω_i *la valeur d'adaptation d'une solution, une solution* $x_i \in R_t$ *remplacera une solution* $x_j \in Q_{t+1}$ *si et seulement si les deux conditions suivantes sont respectées :*

1. $f_1(x_i) = f_1(x_j)$, $i \in R_t$ *et* $j \in Q_{t+1}$
2. $\omega_i < \omega_j$

La taille ainsi que le nombre de sous-groupes sont fixes dans l'archive. Donc à chaque itération, si un individu dans la population possède une meilleure valeur d'adaptation suite à l'application des fouilles Tabou, il remplacera l'individu inclus dans l'archive qui possède pour la même valeur de f_1, la pire valeur d'adaptation.

4.2.7 Opérateur de mutation

Dans la structure de l'algorithme évolutif proposé, l'opérateur de mutation sert à créer une perturbation permettant aux fouilles Tabou d'évoluer davantage. Généralement, pour un problème de création des horaires d'examens, l'opérateur de mutation consiste à déplacer un ou plusieurs examens dans de nouvelles périodes choisies aléatoirement. Comme l'opérateur de mutation est le seul opérateur génétique assurant la diversité des solutions, le taux de mutation devient un facteur important. Généralement dans la littérature un taux de mutation de $1/L$ est proposé, où L représente la longeur d'un chromosome. Ce taux de mutation sera donc choisi.

4.2.8 Gestion de la capacité des locaux

Il existe plusieurs méthodes servant à la gestion d'une contrainte reliée à la capacité des locaux. Une approche intéressante est celle utilisée, entre autres, par Merlot. Ce dernier utilise, dans son recuit simulé, un test de faisabilité lors du voisinage. À chaque itération une solution voisine est choisie à l'aide d'une chaîne de Kempe. Si la solution est réalisable, c'est-à-dire qu'elle respecte la capacité des locaux, alors l'algorithme la retient, sinon cette dernière est automatiquement rejetée. Il existe d'autres techniques basées sur les opérateurs génétiques pour effectuer la gestion des contraintes. On note, entre autres, le *Constrained Tournament* utilisé au chapitre 3. Par contre, tout comme la gestion des conflits, les fouilles Tabou étant trop puissantes, les opérateurs génétiques n'arrivent jamais à traiter ce type de contraintes. L'ajout de techniques autres que celles basées sur la valeur d'adaptation devient donc essentiel.

Un point important à mentionner concerne l'utilisation d'un test de faisabilité lors du voisinage. Cette méthode permet aux deux fouilles locales de ne jamais créer une solution qui ne respecte pas la capacité des locaux. Comme la fouille locale n'est pas le seul opérateur modifiant les solutions, les opérateurs utilisés devront tous être capables de gérer cette

contrainte. Ces derniers sont la mutation et l'initialisation des solutions. Dans la même optique que les techniques utilisées précédemment, les changements apportés sont :

Mutation : Déplacer un examen de façon aléatoire selon un probabilité P_m si, et seulement si, le déplacement ne crée pas un débordement de la capacité des locaux.

Initialisation : Pour chaque examen, choisir une période de façon aléatoire parmi un ensemble de périodes où l'examen ne crée pas de débordement de la capacité des locaux.

4.3 Protocole d'expérimentation

Les bases de données publiques disponibles sur le site Web de Merlot [6] ont été utilisées afin de réaliser l'ensemble des expérimentations. Pour chaque base de données 5 essais ont été effectués. Les paramètres de simulation sont les suivants :

Paramètres de l'algorithme évolutif

a. Initialisation : Assignation aléatoire

b. Mutation : Mutation sans respect aux conflits d'horaire

c. Taux de mutation : $1/L$ (nombre d'examens)

Paramètres de la fouille Tabou FTP

a. Voisinage : Déplacement d'un examen à chaîne de Kempe

b. Échantillonage : Maximum de 25 échantillons évalués à chaque voisinage

c. Taille de la liste : 25

d. Critère d'arrêt : 25 changements de voisinage

Paramètres de la fouille Tabou FTC

a. Voisinage : Déplacement d'un examen en conflit

b. Échantillonage : Maximum de 500 échantillons évalués à chaque voisinage

c. Taille de la liste : 25

d. Critère d'arrêt : 500 itérations sans amélioration de la fonction d'objectif

Ces paramètres sont indentiques pour l'ensemble des simulations effectuées sur toutes les bases de données ainsi que sur les deux modèles P1-P2 et P1-P3.

4.4 Résultats pour le problème P1-P2

Afin d'évaluer les performances de l'AEMH sur un PCHE, le premier modèle multi-critères utilisé sera le problème P1-P2 (problème 3.1, page 63). Les résultats de l'algorithme appliqué sur ce modèle sont montrés au tableau IX. Les tableaux VIII et X comparent les résultats avec ceux déjà publiés. Les résultats inscrits au tableau VIII, portent sur le problème P1 et sont obtenus en identifiant dans l'archive finale la solution réalisable (sans conflit d'horaire) ayant le moins de périodes à l'horaire (valeur de la fonction f_1). Pour les résultats du tableau X portant sur le problème P2, ceux-ci sont obtenus en choisissant la valeur de f_2 de la solution réalisable de l'archive qui possèdent la valeur de f_1 précisée. Comme exemple, les résultats du problème P2 tirés de la base de données *hec-s-92* doivent être comparés à des solutions possédant 18 périodes. Comme il a été montré précédemment, la stratégie d'archivage assure la conservation d'un certain nombre de solutions pour chaque valeur du domaine de f_1.

4.4.1 Analyse des résultats

Mis à part les travaux de Caramia [8], l'algorithme évolutif proposé présente des performances pratiquement semblables aux meilleurs résultats publiés. Sur les bases de données

sta-f-83 et *not-f-94* les résultats obtenus sont les meilleurs comparativement à ceux présentés par les autres rechercheurs. Le tableau VIII montrant les résultats du problème P1 confirme que la fouille Tabou appliquée pour la gestion des conflits d'horaire remplit son objectif puisque pour la majorité des bases de données l'algorithme obtient les meilleurs résultats. En plus, il est important de noter que pour l'ensemble des bases de données aucun ajustement des paramètres n'a été effectué. Tous les essais, sur toutes les bases de données ont été lancés avec la même configuration. Le fait que les résultats sont très bons pour la base de données *sta-f-83* montre que la configuration de l'algorithme est idéale pour la résolution de ce problème. Par contre, pour les données *uta-s-92* par exemple, les résultats se situent plutôt dans la moyenne. Comme l'objectif de la démarche est de formuler une méthode d'optimisation flexible pour la résolution de n'importe quel type de données, on peut conclure que cet objectif est amplement atteint. L'algorithme peut donc performer d'une façon très acceptable sur la majorité des bases de données provenant d'une institution quelconque. Afin de permettre une meilleure visualisation du comportement de l'algorithme, la prochaine section présentera l'évolution de l'archive tout au long de l'algorithme sous forme de graphique.

Tableau VIII

Résultats de la fonction d'objectif pour les bases de données publiques selon le modèle P1 (mise à jour)

DONNÉES	AEMH	Merlot [5]	Carter [29]	Caramia [8]
CAR-F-92	30	31	**28**	**28**
CAR-S-91	31	30	**28**	**28**
EAR-F-83	**22**	24	**22**	**22**
HEC-S-92	**17**	18	**17**	**17**
KFU-S-93	**19**	21	**19**	**19**
LSE-F-91	**17**	18	**17**	**17**
MEL-F-01	**23**	28	—	—
MEL-S-01	**23**	31	—	—
NOT-F-94	**22**	23	—	—
RYE-F-92	**21**	22	**21**	**21**
STA-F-83	**13**	**13**	**13**	**13**
TRE-S-92	**20**	21	**20**	**20**
UTA-S-92	34	32	32	**30**
UTE-S-92	**10**	11	**10**	**10**
YOR-F-83	**19**	23	**19**	**19**

Tableau IX

Résultats de l'algorithme sur le problème P1-P2

CAR-F-92	nb. Périodes	**30**	**31**	**32**	**33**	**34**
	Moyenne	4.9	4.6	4.4	4.4	4.1
CAR-S-91	nb. Périodes	**32**	**33**	**34**	**35**	**36**
	Moyenne	6.2	5.9	5.4	5.5	5.3
EAR-F-83	nb. Périodes	**23**	**24**	**25**	**26**	**27**
	Moyenne	39.0	35.6	31.9	29.7	27.5
HEC-S-92	nb. Périodes	**17**	**18**	**19**	**20**	**21**
	Moyenne	12.1	10.5	9.3	8.2	7.5
KFU-S-93	nb. Périodes	**19**	**20**	**21**	**22**	**23**
	Moyenne	16.2	14.4	12.7	11.6	10.3
LSE-F-91	nb. Périodes	**17**	**18**	**19**	**20**	**21**
	Moyenne	12.6	11.6	10.1	9.3	8.1
MEL-F-01	nb. Périodes	**26**	**27**	**28**	**29**	**30**
	Moyenne	3.7	3.2	2.9	2.9	2.5
MEL-S-01	nb. Périodes	**29**	**30**	**31**	**32**	**33**
	Moyenne	3.0	2.7	2.5	2.3	2.1
NOT-F-94	nb. Périodes	**22**	**23**	**24**	**25**	**26**
	Moyenne	8.1	7.2	6.5	5.8	5.1
RYE-F-92	nb. Périodes	**22**	**23**	**24**	**25**	**26**
	Moyenne	10.1	9.1	8.1	7.2	7.3
STA-F-83	nb. Périodes	**13**	**14**	**15**	**16**	**17**
	Moyenne	157.1	140.4	126.5	113.4	101.6
TRE-S-92	nb. Périodes	**21**	**22**	**23**	**24**	**25**
	Moyenne	10.5	9.4	8.8	8.1	7.3
UTA-S-92	nb. Périodes	**34**	**35**	**36**	**37**	**38**
	Moyenne	3.9	3.6	3.4	3.2	3.2
UTE-S-92	nb. Périodes	**10**	**11**	**12**	**13**	**14**
	Moyenne	25.4	21.2	17.1	14.2	11.6
YOR-F-83	nb. Périodes	**19**	**20**	**21**	**22**	**23**
	Moyenne	45.6	41.0	37.7	34.4	31.9

Tableau X

Résultats de la fonction d'objectif pour les bases de données publiques selon le modèle P2 (mise à jour)

Données		AEMH	Cart[29]	DiG[11]	Cara[8]	Bur[28]	Mer[5]	Paq[7]
CAR-F-92	Meilleure	4.4	10.5	5.2	6.0	**4.0**	4.3	—
32 périodes	Moyenne	4.2	15.0	5.6	—	**4.1**	4.4	—
CAR-S-91	Meilleure	5.4	7.1	6.2	6.6	**4.6**	5.1	—
35 périodes	Moyenne	5.5	8.4	6.5	—	**4.7**	5.2	—
EAR-F-83	Meilleure	34.2	36.4	45.7	**29.3**	35.1	35.1	40.5
24 périodes	Moyenne	35.6	40.9	46.7	—	**35.4**	**35.4**	45.8
HEC-S-92	Meilleure	10.4	10.6	12.4	**9.2**	11.3	10.6	10.8
18 périodes	Moyenne	**10.5**	15.0	12.6	—	11.5	**10.7**	12.0
KFU-S-93	Meilleure	14.3	14.0	18.0	13.8	13.7	**13.5**	16.5
20 périodes	Moyenne	14.4	18.8	19.5	—	**13.9**	14.0	18.3
LSE-F-91	Meilleure	11.3	10.5	15.5	**9.6**	10.6	10.5	13.2
18 périodes	Moyenne	11.5	12.4	15.9	—	**10.8**	11.0	15.5
MEL-F-01	Meilleure	**2.8**	—	—	—	—	**2.9**	—
28 périodes	Moyenne	**2.9**	—	—	—	—	**3.0**	—
MEL-S-01	Meilleure	**2.4**	—	—	—	—	**2.5**	—
31 périodes	Moyenne	**2.5**	—	—	—	—	**2.5**	—
NOT-F-94	Meilleure	**6.9**	—	—	—	—	**7.0**	—
23 périodes	Moyenne	7.2	—	—	—	—	**7.1**	—
RYE-F-92	Meilleure	8.8	7.3	—	**6.8**	—	8.4	—
23 périodes	Moyenne	9.0	**8.7**	—	—	—	**8.7**	—
STA-F-83	Meilleure	**157.0**	161.5	160.8	158.2	168.3	**157.3**	158.1
13 périodes	Moyenne	**157.1**	167.1	166.8	—	168.7	**157.4**	159.3
TRE-S-92	Meilleure	8.6	9.6	10.0	9.4	**8.2**	8.4	9.3
23 périodes	Moyenne	8.8	10.8	10.5	—	**8.4**	8.6	10.2
UTA-S-92	Meilleure	3.5	3.5	4.2	3.5	**3.2**	3.5	—
35 périodes	Moyenne	3.6	4.8	4.5	—	**3.2**	3.6	—
UTE-S-92	Meilleure	25.3	25.8	29.0	**24.4**	25.1	25.1	27.8
10 périodes	Moyenne	25.5	30.8	31.3	—	**25.2**	**25.2**	29.4
YOR-F-83	Meilleure	36.4	41.7	41.0	**36.2**	36.8	37.4	38.9
21 périodes	Moyenne	37.5	45.6	42.1	—	**37.3**	37.9	41.7

4.4.2 Progression de l'archive

Pour présenter l'évolution de l'archive, trois types de graphiques ont été utilisés. Il est important de noter que toutes les solutions imprimées sur les graphiques sont les solutions réalisables et non dominées de l'archive. Le premier type de graphique présente l'évolution de l'archive pour toutes les itérations de l'algorithme. Afin de permettre de mieux visualiser la convergence de l'archive, le deuxième type de graphique procure quatre agrandissements selon les quatre valeurs les plus intéressantes de la fonction f_1. Les solutions seront montrées par tranche de 25 itérations. Pour terminer, le troisième type de graphique présente, encore par tranche de 25 itérations, l'archive complète. Cette représentation permet de bien évaluer la présence ou l'absence de solution pour une valeur du domaine de la fonction f_1. Les quatre bases de données les plus intéressantes seront présentées chacune par ces trois types de graphique. Le tableau XI fait le bilan des bases de données retenues.

Tableau XI

Résumé des différents graphiques présentés pour les résultats du modèle P1-P2

Bases de Données	**Nombre d'examens**	**Nombre d'élèves**	**Graphiques associés**
hec-s-92	81	2823	figures : 17, 18 et 19
yor-f-83	181	941	figures : 23, 24 et 25 Annexe
kfu-s-93	461	5349	figures : 26, 27 et 28 Annexe
car-s-91	682	16925	figures : 29, 30 et 31 Annexe

Il est intéressant de voir que pour certaines bases de données (*hec-s-92* fig.17 et *kfu-s-93* fig. 26) l'archive finale ne contient pas de solution non dominée pour certaines valeurs du domaine de f_1. Cela provient du fait que lors de l'évolution de l'algorithme une solution ayant moins de périodes possède un niveau de proximités plus bas qu'une solution ayant une ou plusieurs périodes supplémentaires. Dans ce cas, la solution possédant une période de plus avec un niveau supérieur de proximités possèdera une valeur d'adaption moindre.

Comme le montre la figure 19, il arrive même que ce phénomène se présente sur plusieurs valeurs du domaine de f_1. Sur cette même figure on constate qu'aucune solution non dominante n'est présente à l'itération 25 pour un horaire ayant 23 et 25 périodes. Par contre, dès l'itération 50, une solution non dominée ayant 23 périodes s'ajoute à l'archive. Pour les horaires ayant 25 périodes aucune autre solution non dominée ne sera présente. Le fait de perdre des solutions non dominées est une faille dans l'algorithme. Une amélioration future pourra être faite pour palier à ce problème. Il est important de noter que l'absence de solution non dominée dès le début de l'algorithme est due à la gestion des conflits d'horaire faite par la fouille Tabou. Cette dernière n'arrive pas à respecter cette contrainte dès la première itération. Comme le montre la figure 28, pour la base de données *kfu-s-93* il a fallu attendre 25 itérations avant d'avoir une solution réalisable pour chaque valeur de f_1, alors que pour la base de données *car-s-91* ce n'est qu'à la 100^{ieme} itération qu'une solution sans conflit ayant 33 périodes est apparue.

La dernière observation se situe au niveau des graphiques de l'évolution des archives par agrandissement de périodes. En prenant l'exemple de la base de données *yor-f-83* (figure 24) on constate que selon la valeur de f_1 (le nombre de périodes) l'algorithme a convergé ou n'a pas encore convergé. Pour les solutions ayant 21 périodes l'algorithme a convergé depuis l'itération 75, alors que pour les solutions ayant 24 périodes l'algorithme, après 150 itérations, n'a pas encore convergé. En se penchant plus précisément sur la nature du problème, il est possible d'expliquer ce phénomème. En modifiant la fonction f_1, le domaine de décision change aussi. Par exemple, si on augmente le nombre de périodes à l'horaire on augmente du même coup l'espace de décision et le nombre de combinaisons possibles. Ceci implique que le nombre d'itérations de la fouille Tabou des proximités devrait en principe augmenter à mesure que le nombre de périodes augmente. Cet aspect pourrait faire l'objet d'améliorations futures.

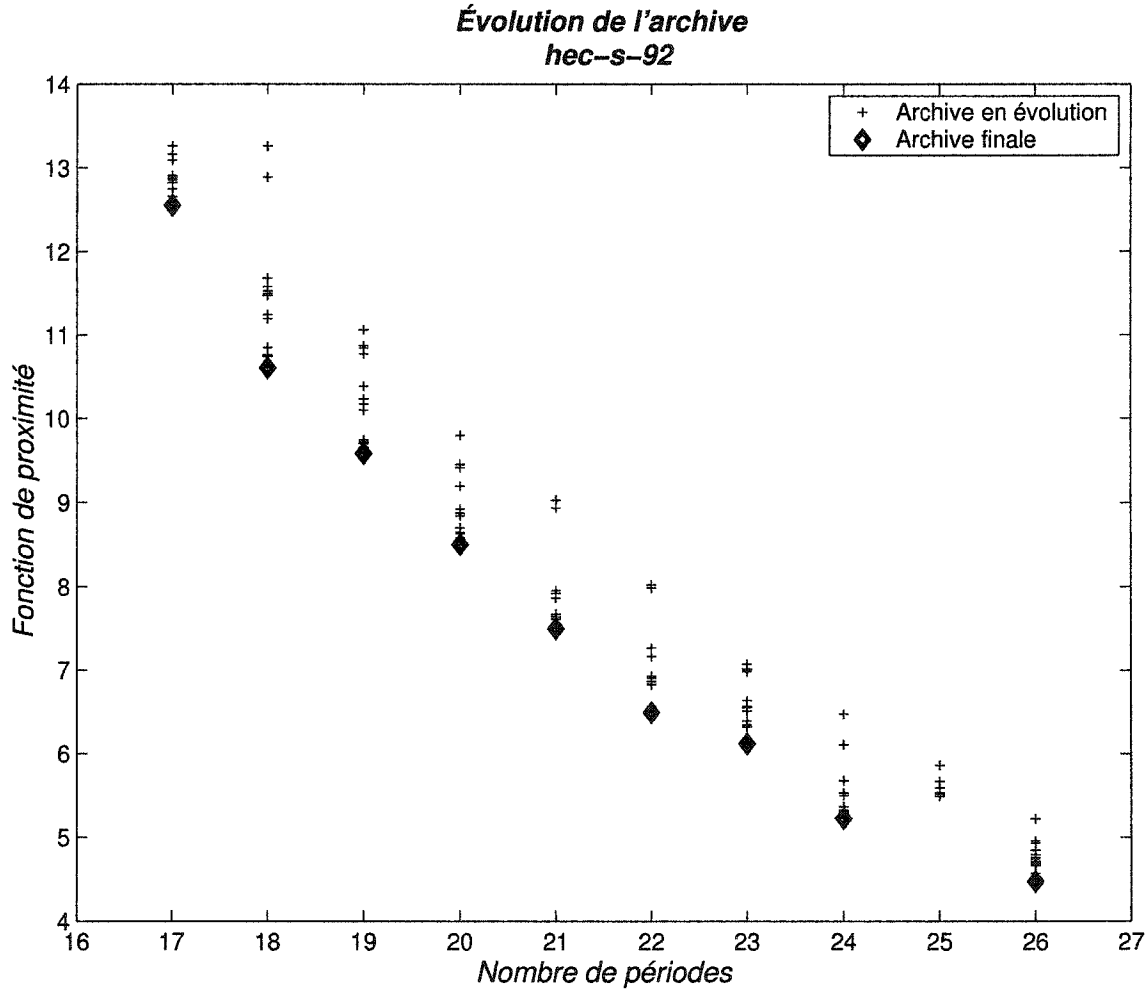

Figure 17 Évolution de l'archive *hec-s-92* (P2) : vue d'ensemble

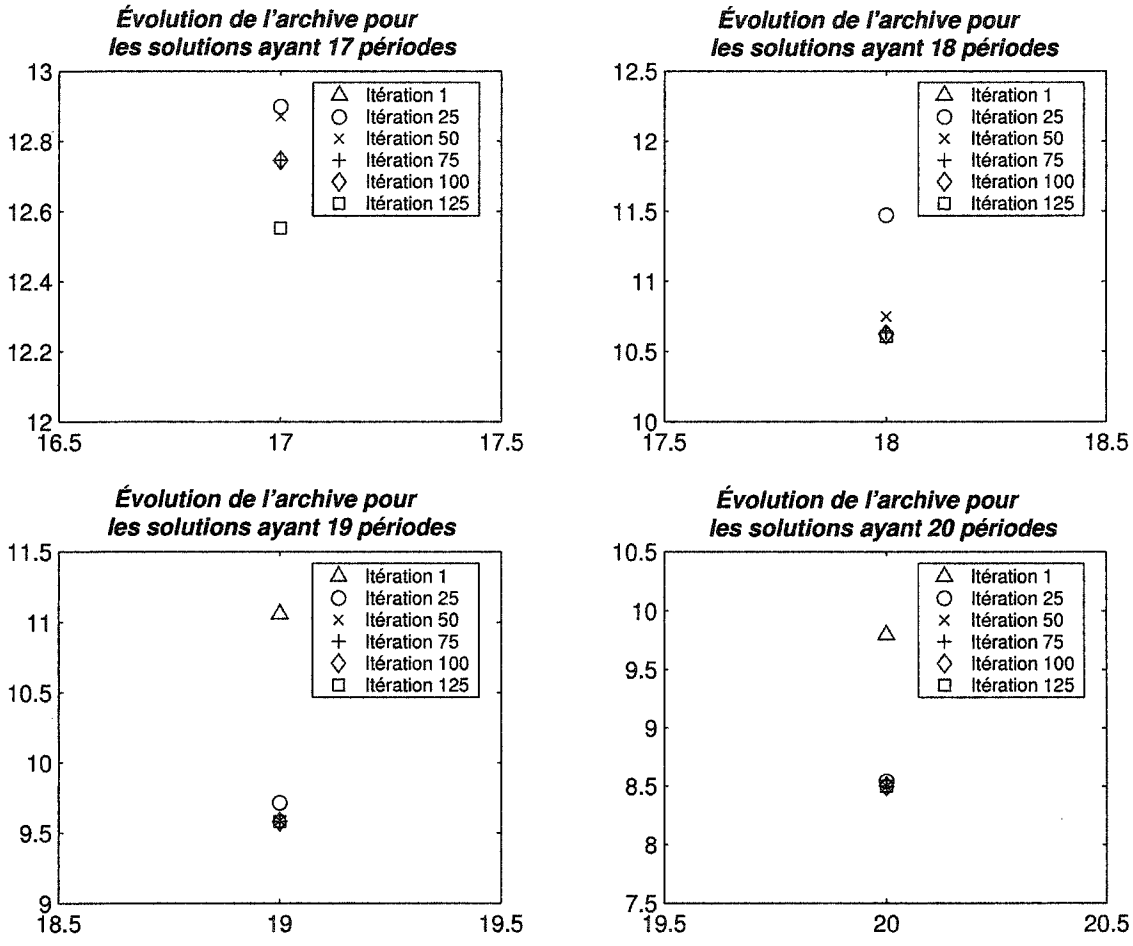

Figure 18 Évolution de l'archive *hec-s-92* (P2) : agrandissement par période

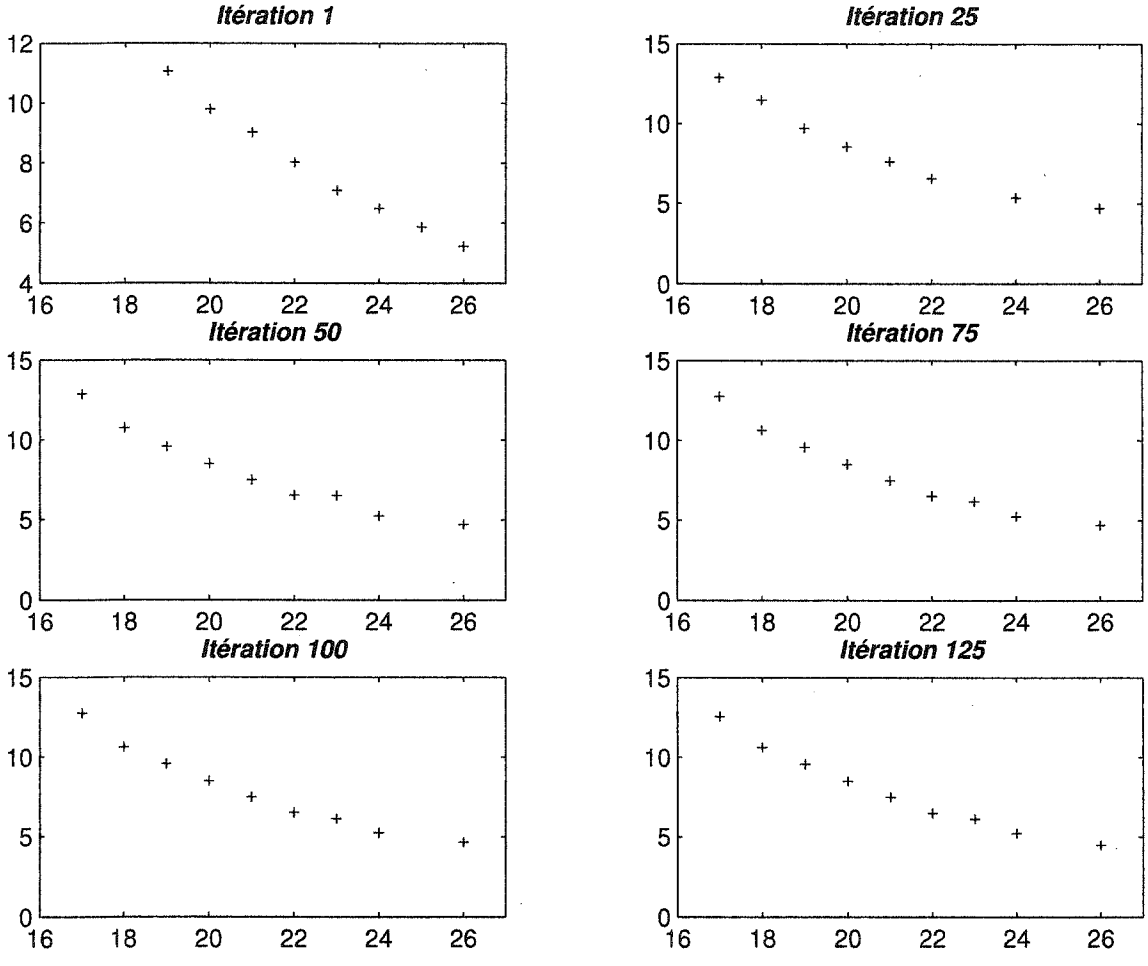

Figure 19 Évolution de l'archive *hec-s-92* (P2) : tranche de 25 itérations

4.5 Résultats pour le problème P1-P3

Le problème P1-P2 (problème 3.1, page 63) n'est pas très réaliste. Une des contraintes importantes d'une institution procédant à la création d'un horaire d'examens est la capacité des locaux disponibles. Cette contrainte peut engendrer de graves problèmes administratifs si elle n'est pas respectée. Pour ces raisons Burke et al. ont publié un autre type de problème standard soit le problème P3. Ce problème tient compte à la fois des proximités, des conflits d'horaire, du nombre de périodes et de la capacité des locaux :

Problème 4.1 *Le problème multi-critères P1-P3 est défini par :*

$$\begin{aligned}
\min \quad & f_1 = \|\mathcal{T}\| ; \\
\min \quad & f_2 = \frac{1}{2} \sum_{k=1}^{\|\mathcal{T}\|-1} \sum_{i=1}^{\|\mathcal{E}\|} \sum_{j=1}^{\|\mathcal{E}\|} \eta_{ij} \epsilon_{ik} \epsilon_{jk+1} ; \\
s.a. \quad & \sum_{k=1}^{\|\mathcal{T}\|} \sum_{i=1}^{\|\mathcal{E}\|} \sum_{j=1}^{\|\mathcal{E}\|} \eta_{ij} \epsilon_{ik} \epsilon_{jk} = 0 ; \\
& \sum_{i=1}^{\|\mathcal{S}\|} \sum_{j=1}^{\|\mathcal{E}\|} \tau_{ij} \epsilon_{jk} \leq \sum_{m=1}^{\|\mathcal{L}\|} Cp_m, \qquad \forall\, k \in \mathcal{T}.
\end{aligned}$$

où $\mathcal{T}$ représente l'ensemble des périodes de l'horaire, $\mathcal{E}$ définit l'ensemble des examens et $\mathcal{L}$ est l'ensemble des locaux disponibles. De plus, η_{ij} est le nombre d'élèves en commun entre l'examen i et j, Cp_m indique la capacité maximale du local m, $\tau_{ij} = 1$ si l'élève i est inscrit à l'examen j, sinon $\tau_{ij} = 0$ et $\epsilon_{jk} = 1$ si l'examen j est assigné à période k, sinon $\epsilon_{jk} = 0$

Ce modèle revient à résoudre le problème P1 et P3 dans une même simulation. Il est important de noter que la fonction d'objectif reliée aux proximités P3 diffère de celle utilisée dans le modèle P1-P2. Dans le modèle P1-P3 (problème 4.1), la deuxième fonction d'objectif, celle des proximités, indique seulement les élèves ayant deux examens consécutifs. Pour se rapprocher encore plus de la réalité, les périodes ne seront plus considérées comme une suite d'évènements indépendants mais bien comme une semaine complète avec ses

soirs, ses matins et ses fins de semaine. Dans ce contexte, les élèves ayant un examen le soir et un le lendemain matin ne sont pas inclus dans la fonction d'objectif. Une semaine comporte cinq jours et une période est prévue pour assigner des examens le samedi matin. Une contrainte de capacité des locaux s'ajoute à ce modèle en plus de la contrainte des conflits d'horaire. La deuxième contrainte du problème P1-P3 indique que le nombre total d'élèves qui ont leurs examens à la période k ne doit pas dépasser la capacité total des locaux. La section 4.2.8 a présenté la manière dont l'algorithme effectuera la gestion de la contrainte de capacité.

Tableau XII

Résultats de l'algorithme sur le problème P1-P3

CAR-F-92	nb. Périodes	**39**	**40**	**41**	**42**	**43**
	Moyenne	298	267	248	155	168
CAR-S-92	nb. Périodes	**51**	**52**	**53**	**54**	**55**
	Moyenne	78	94	89	93	47
KFU-S-93	nb. Périodes	**20**	**21**	**22**	**23**	**24**
	Moyenne	322	278	195	115	83
NOT-F-94	nb. Périodes	**23**	**24**	**25**	**26**	**27**
	Moyenne	182	119	103	46	34
TRE-S-92	nb. Périodes	**34**	**35**	**36**	**37**	**38**
	Moyenne	2.8	2.4	0	7.8	2.0
UTA-S-92	nb. Périodes	**37**	**38**	**39**	**40**	**41**
	Moyenne	417	338	230	200	141

4.5.1 Analyse des résultats

Tout comme le problème P2, plusieurs auteurs ont validé leurs algorithmes sur les bases de données publiques. Le tableau XII montre les résultats obtenus suite à l'application de l'algorithme évolutif proposé pour la résolution du modèle P1-P3 alors que le tableau XIII compare les résultats avec ceux déjà publiés. Pour le problème P3, en général les chercheurs n'ont pas utilisé toutes les bases de données disponibles. La comparaison sera donc effectuée sur les bases de données les plus utilisées pour le problème P3.

Contrairement aux modèles P2, les résultats de Merlot sont supérieurs, en moyenne, à tous les autres. Caramia ne réussit pas le même tour de force qu'avec le modèle P2. Pour ce qui est des résultats du AEMH ils se classent en deuxième position après ceux de Merlot. Encore une fois on note que pour certaines bases de données comme *uta-s-92* les résultats sont très bons. La configuration de l'algorithme semble donc être idéale pour cette base de données. Le même phénomène avait été observé lors de la résolution du modèle P2 pour la base de données *sta-f-83*. C'est avec la base de données *not-f-94* que les performances sont les moins intéressantes.

Tableau XIII

Résultats de la fonction d'objectif pour les bases de données publiques selon le modèle P3 (mise à jour)

Données			AEMH	DiG[11]	Cart[8]	Bur[28]	Mer[5]
CAR-F-92	Périodes 40	Meilleure	204	331	424	268	**158**
	Capacité 2000	Moyenne	267	—	443	—	**212.8**
CAR-S-91	Périodes 51	Meilleure	70	81	88	74	**31**
	Capacité 1550	Moyenne	78.6	—	98	—	**47**
KFU-S-93	Périodes 20	Meilleure	292	974	512	912	**247**
	Capacité 1995	Moyenne	309	—	597	—	**282.8**
NOT-F-94	Périodes 23	Meilleure	156	123	—	269	**83**
	Capacité 1550	Moyenne	181.8	134	—	—	**105**
NOT-F-94	Périodes 26	Meilleure	33	11	44	53	**2**
	Capacité 1550	Moyenne	45.6	**13**	—	—	15.6
TRE-S-92	Périodes 35	Meilleure	**0**	4 5	2	53 3	**0**
	Capacité 655	Moyenne	2.9	—	—	—	**0.4**
UTA-S-92	Périodes 38	Meilleure	**245**	772	554	680	334
	Capacité 2800	Moyenne	**338.5**	—	625	—	393.4

4.5.2 Progression de l'archive

Les trois types de graphiques utilisés afin d'effectuer l'analyse de l'archive représentent une vue d'ensemble de l'évolution de l'archive, un agrandissement de l'évolution de l'archive selon différentes valeurs de f_1 et une présentation de l'archive par tranche de 25 itérations. Le tableau XIV fait le bilan de l'ensemble des graphiques présentés.

Comme l'algorithme est sensiblement le même que celui utilisé pour le problème P1-P2, les mêmes caractéristiques apparaissent au niveau des graphiques présentés. On observe toujours l'absence de quelques solutions non dominées dans l'archive finale pour des valeurs de f_1. Pour ce qui est du nombre d'itérations de l'algorithme, on peut conclure qu'en moyenne une centaine d'itérations sont nécessaires pour une bonne convergence. De nombreux essais étant maintenant réalisés avec succès, il est possible de tirer des conclusions plus générales et de comparer les avantages et les inconvénients de l'algortihme évolutif proposé dans ce mémoire avec ceux qui ont déjà été publiés.

Tableau XIV

Résumé des différents graphiques présentés pour les résultats du modèle P1-P3

Bases de données	**Nombre d'examens**	**Nombre d'élèves**	**Capacité des locaux**	**Graphiques associés**
car-f-92	543	18419	2000	figure : 20, 21 et 22
kfu-s-93	461	5349	1995	Annexe
uta-s-92	800	7896	1550	Annexe

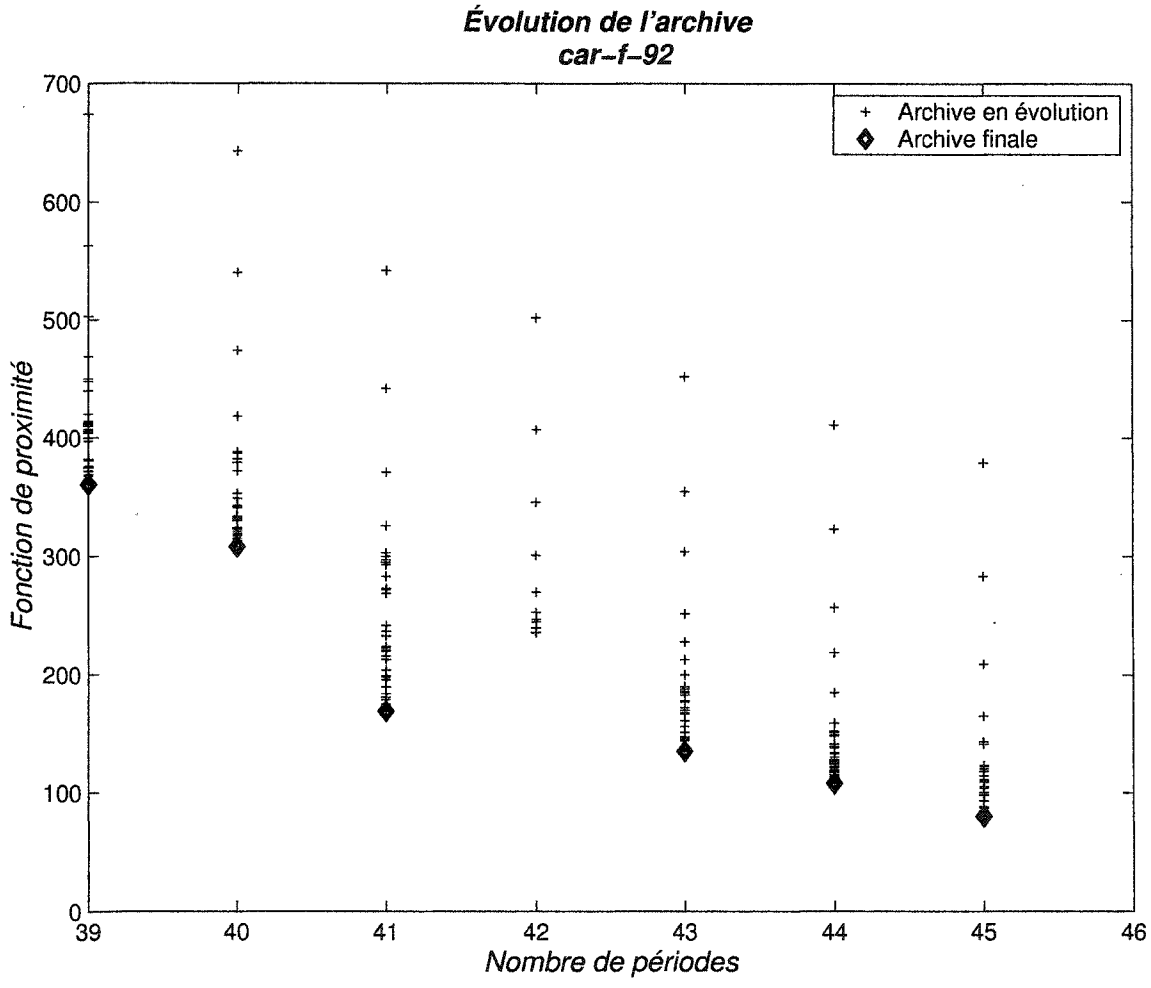

Figure 20 Évolution de l'archive *car-f-92* (P3) : vue d'ensemble

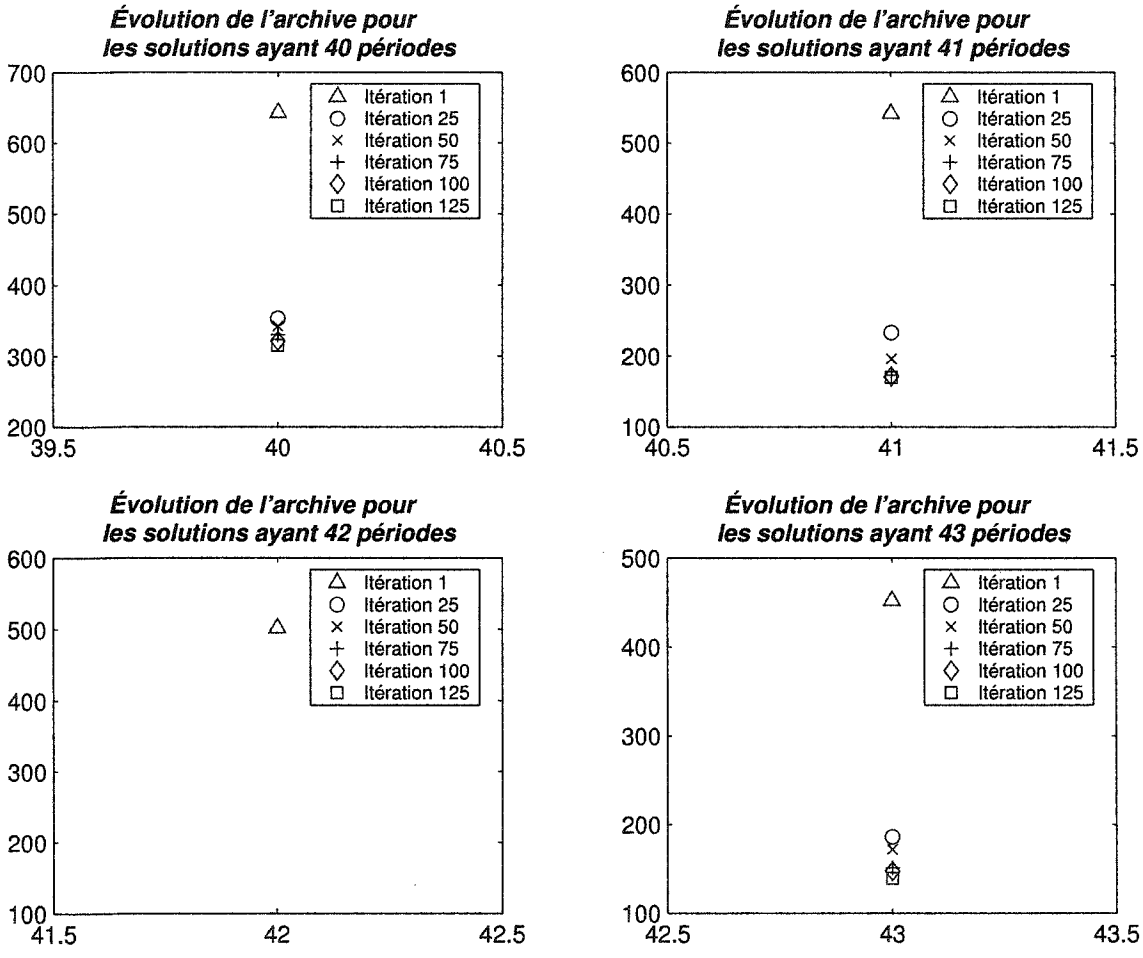

Figure 21 Évolution de l'archive *car-f-92* (P3) : agrandissement par période

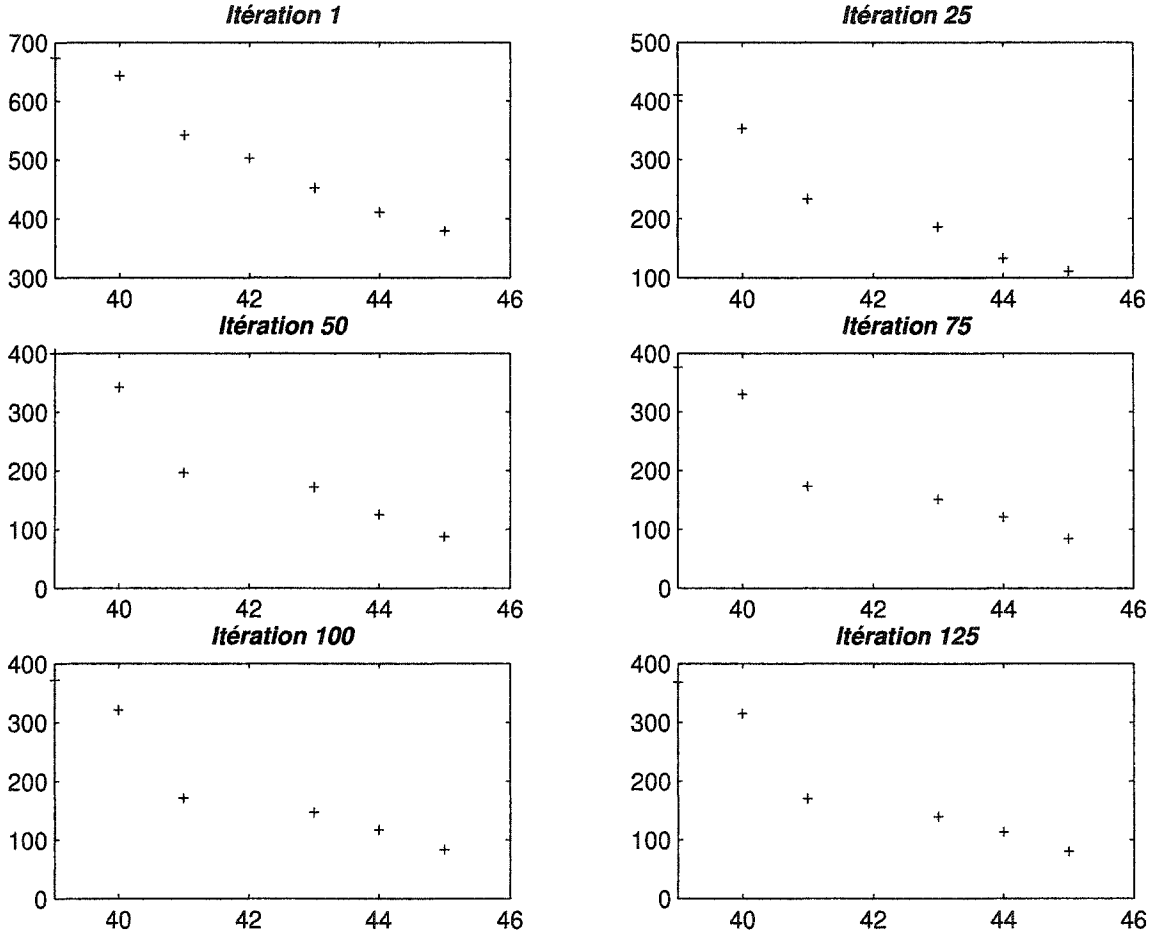

Figure 22 Évolution de l'archive *car-f-92* (P3) : tranche de 25 itérations

4.6 Conclusion

La conception et l'implantation d'un nouvel AEMH ont été réalisées avec succès. Les résultats ont montré que l'algorithme performe aussi bien sur le problème P2 que sur le problème P3. Toutes les lacunes observées lors des simulations avec les algorithmes évolutifs multi-critères NSGA-II et SPEA-II ont été corrigées. Tout d'abord la gestion des conflits d'horaire est réalisée de façon très efficace par l'ajout d'une fouille locale appliquée à chaque itération sur l'ensemble des solutions. Une deuxième fouille locale servant à l'optimisation des proximités a permis d'obtenir des résultats excellents et comparables aux taux de proximités obtenus par les meilleurs travaux publiés. Pour terminer, la diversité, qui est un objectif important dans la résolution d'un problème d'optimisation multi-critères, a été traitée avec succès. L'utilisation de l'archive permet d'enregistrer les meilleures solutions pour l'ensemble des valeurs du domaine de la fonction du nombre de périodes. Avec ces observations on peut donc conclure que l'objectif globale de ce projet est atteint.

CHAPITRE 5

CONCLUSION

5.1 Points forts du AEMH

Tout d'abord, le point le plus important à souligner est l'originalité de ce travail. L'algorithme qui a été conçu est un assemblage de différentes techniques qui forment une approche novatrice appliquée au problème de la création des horaires d'examens. De plus, l'AEMH est le premier algorithme multi-critères à avoir été implanté sur des bases de données publiques. En plus de son caractère unique, l'AEMH a donné de très bons résultats comparables à ceux publiés. Pour le modèle P1-P2 qui minimise le taux de proximités en réduisant la longueur de l'horaire, l'algorithme réussit à obtenir en moyenne le taux le plus faible de proximités et ce pour les bases de données *hec-s-92*, *mel-s-01*, *mel-f-01* et *sta-f-83*. Pour le modèle P1-P3 qui minimise la longueur d'un horaire et le taux de proximités tout en respectant la capacité des locaux, l'AEMH a réussi à avoir le taux de proximités moyen le plus faible pour les bases de données *uta-s-92* et *tre-s-92*. Pour ce qui est des autres bases de données, l'AEMH se positionne en général en deuxième place. De plus, tous les résultats ont été obtenus en utilisant les mêmes paramètres de simulation. Aucune modification spéciale n'a été faite en fonction de la base de données utilisée. Ceci montre bien que l'AEMH est très polyvalent et permet d'obtenir de très bons horaires peu importe les données utilisées.

Le fait que l'AEMH soit un algorithme multi-critères offre quelques avantages intéressants. Utilisé dans un contexte de production réelle, l'algorithme permettrait de présenter à l'usager un ensemble de solutions non dominées pour différentes longueurs d'horaire. L'usager pourrait donc effectuer un choix à postériori sans avoir à relancer l'algorithme de nouveau. Ce dernier pourrait voir directement l'impact de la réduction de la taille de l'horaire sur le taux de proximités, ce qui éliminerait toute forme de pondération de cri-

tères posée à priori. Si, comme exemple, suite à l'obtention des résultats, l'usager devait réduire la taille de l'horaire pour des raisons administratives, il pourrait directement opter pour une autre alternative en choisissant une autre solution dans l'archive. De cette façon, l'usager n'aurait pas à relancer l'algorithme en essayant de trouver des pondérations qui permettrait de trouver un bon compromis entre le temps d'étude donné aux élèves et la réduction de la taille de l'horaire.

L'utilisation d'une fouille locale comparativement au croisement est un avantage notable qui permet une meilleure exploitation des solutions. Nos travaux vont donc dans le même sens que ceux de Burke et *al.* [28] qui ont développé un algorithme évolutif où le croisement a été remplacé par une fouille en escalade. De plus, la fouille qui a été implantée dans l'AEMH utilise un voisinage multiple où il y a alternance entre le déplacement d'un examen et le voisiange par chaîne de Kempe. Bien peu de travaux dans le domaine de la création des horaires utilisent un voisinage multiple. Comme il a été montré dans ce chapitre, ce voisinage a grandement contribué à la puissance de la fouille locale servant à la réduction du taux de proximités. L'AEMH demeure donc un algorithme très simple à implanter.

Ce travail de recherche a été soumis à deux importantes conférences internationales soit le *Congress on Evolutionary Computation* (CEC2004) [2] ainsi qu'à la cinquième conférence *Practice and Theory of Automated Timetabling* (PATAT2004) [1]. Dans les deux cas, les articles ont été acceptés pour une présentation orale. Ces conférences sont bien différentes mais elles présentent toutes deux un intérêt majeur. Avec la CEC2004, l'algorithme a été accepté par la communauté des scientifiques travaillant sur les techniques évolutives et avec la conférence PATAT2004, les travaux ont été acceptés par la commauté des scientifiques travaillant sur l'optimisation des horaires. Ce travail a donc atteint pleinement son objectif qui consistait en l'application d'un algorithme évolutif multi-critères sur le problème de la création des horaires d'examens.

5.2 Points faibles du AEMH

Un des inconvients majeurs de la méthode développée est la flexibilité. Cette lacune est due à la puissance de la fouille locale servant à la réduction du taux de proximités. En prenant par exemple le cas d'un algorithme génétique multi-critères simple, le seul opérateur assurant l'exploitation des solutions est le croisement. Cet opérateur est direcetement lié à une valeur d'adaptation qui, dans la plupart des cas, est basée sur le principe de dominance, donc toutes les fonctions d'objectifs convergent à l'aide d'un seul opérateur. Dans la méthode utilisée, la convergence de la fonction d'objectif des proximités est assurée par un opérateur totalement indépendant de la fonction d'objectif du nombre de périodes. Dans le cas présent, ceci a peu d'importance étant donnée la nature de la fonction d'objectif du nombre de périodes. Par contre, si une troisième fonction d'objectif devait être ajoutée, l'opérateur d'assignation de la valeur d'adaptation ne serait pas assez puissant pour contrer la pression exercée par la fouille locale servant à minimiser les proximités, concrètement prenons l'exemple de la capacité des locaux. Si l'on désirait minimiser le nombre d'élèves par période en ajoutant une nouvelle fonction d'objectif, la fouille locale des proximités devra absolument tenir compte de cette nouvelle modification, sinon tout le travail fait par l'opérateur d'assignation de la valeur d'adaption et de sélection sera anéanti par l'application d'une nouvelle fouille locale. Maintenant, si le problème nécessitait l'ajout d'une quatrième et même d'une cinquième fonction d'objectif tous les opérateurs assurant la convergence devront en tenir compte. En résumé, le désavantage majeur est que la convergence des fonctions d'objectif n'est pas assurée par un même opérateur, mais bien par un ensemble d'opérateurs indépendants.

Un autre problème est l'encodage des chromosomes. La méthode choisie permet d'utiliser les opérateurs génétiques tels le croisement ou la mutation. Par contre, cette technique génère un grand nombre de conflits lors de l'application des opérateurs génétiques. Dans le cas présent, une fouille Tabou a été utilisée pour réduire ce nombre de conflits. Dans ce

contexte la fouille Tabou devient un autre algorithme qu'il faut bien utiliser et ajuster afin d'obtenir des résultats concluants.

Le dernier point négatif observé est l'initialisation des solutions pour le modèle P1-P3. La capacité des locaux a été traitée avec le voisinage des solutions dans les fouilles locales. L'acceptation des solutions se fait si la solution respecte cette contrainte. Pour arriver à respecter cette contrainte, l'opérateur de mutation ainsi que l'initialisation devaient prendre en considération cette contrainte. Pour ce qui est de la mutation, aucun problème particulier ne s'est présenté. La mutation procède au déplacement d'un examen dans une période choisie au hasard. Ce choix s'effectue parmi les périodes où le déplacement ne crée pas un débordement de la capacité des locaux. Cependant, pour l'initialisation un problème est apparu. L'initialisation se fait en respectant la capacité des locaux. Il est tout de même arrivé dans certains cas qu'un examen ne puisse être assigné sans dépassement de la capacité. Lorsque cette situation se présente, l'algorithme boucle à l'infini. Il serait donc très intéressant d'utiliser une meilleure technique d'initialisation qui permettrait à tout coup d'obtenir des solutions respectant la capacité des locaux.

5.3 Améliorations possibles du AEMH

Bien entendu, comme mentionné au paragraphe précédent, la flexibilité de l'algorithme demeure une préoccupation importante. La nature même du problème nécessite l'ajout d'une fouille locale pour l'optimisation des proximités. Présentement l'ajout de nouvelles fonctions d'objectifs et de contraintes restent une tâche très délicate. Il serait très intéressant de trouver un opérateur ou une façon de centraliser la convergence de l'ensemble des fonctions d'objectifs.

La fouille locale servant à l'optimisation des proximités a mis en valeur l'effet du voisinage multiple. En effet, avec l'utilisation de ce voisinage, le taux de proximités obtenu est de loin supérieur à celui obtenu par un voisinage unique. Il serait donc intéressant d'investiguer davantage l'utilisation d'un voisinage multiple en intégrant d'autres types de

voisinage que celui par chaîne de Kempe et déplacement d'un examen. Il serait également possible d'explorer d'autres types de fouilles locales que la fouille Tabou. Il serait même possible d'envisager l'utilisation de plusieurs types de fouilles locales dans une même implantation, donc multiples fouilles locales avec multiples voisinages.

Le modèle P3 utilisé lors des dernières simulations ressemble à une implantation réelle. Par contre, un PCHE présente d'autres objectifs et d'autres contraintes qui n'ont pas été modélisés. Notons, entre autre, la pré-assignation des examens. En optant pour la même démarche que la capacité des locaux, c'est-à-dire par la gestion au niveau du voisinage, il serait facile de traiter la pré-assignation des examens. Un PCHE peut aussi contenir une assignation des locaux, une gestion des surveillances d'examen, etc. Il serait intéressant d'obtenir un algorithme évolutif capable de traiter l'ensemble des objectifs d'un problème réel.

Bien que le temps de traitement soit une notion plutôt subjective, il n'en demeure pas moins que la méthode proposée est sans aucun doute la plus lente jamais proposée dans la littérature. Dans les résultats publiés sur le modèle P2, la plupart des auteurs arrivent dans un temps variant entre 4 et 600 secondes. Pour la méthode évolutive proposée aucune base de données n'a été executée en dèça de 300 secondes. Dans certains cas, le temps se mesure même en journée. Bien entendu aucune institution n'aurait comme exigence de produire un horaire en quelques secondes. Les sessions se déroulent la plupart du temps sur quatre mois et la production des horaires ne se fait que deux ou trois fois par année. Mais pour des fins de recherche, il pourrait être intéressant d'implanter une version distribuée pouvant améliorer le temps de traitement ou bien augmenter la taille de la population tout en gardant le temps de traitement dans la même plage de valeur ce qui augmenterait sûrement la qualité des horaires obtenus.

Finalement, l'algorithme a montré d'excellents résultats sur les bases de données publiques. Deux types de problèmes ont fait partie de l'étude soit le problème P2, mini-

misation des proximités et le problème P3, minimisation des proximités avec respect de la capacité des locaux. Pour ce qui est du problème P2, l'algorithme a réussi à établir des nouvelles marques pour les bases de données *sta-f-83*, *not-f-94* ainsi que les données de Melbourne. En général, seul le chercheur Caramia obtient dans l'ensemble des résultats supérieurs. Pour ce qui est du modèle P3, Merlot est l'auteur qui réussit à obtenir la meilleure performance moyenne. L'algorithme développé, obtenant la deuxième meilleure performance moyenne, a quand même réussi à diminuer le plus faible taux de proximités pour la base de données *uta-s-92*. Bien que l'algorithme développé dans ce travail ne soit pas aussi flexible que souhaité, il ne fait aucun doute que les méthodes évolutives multi-critères sont les plus appropriées pour l'optimisation du problème mulit-critères qu'est celui des horaires d'examens.

ANNEXE 1

RÉSULTATS DE SIMULATION POUR LE PROBLÈME P1-P2

Les graphiques présentés dans cette annexe montrent l'évolution des solutions réalisables de l'archive suite à l'application de l'AEMH sur le problème P1-P2 et P1-P3. Les bases de données utilisées pour ces graphiques sont : *yor-f-83*, *kfu-s-93*, *car-s-91* pour P1-P3 et *kfu-s-93*, *uta-s-92* pour P1-P3. Les caractéristiques de ces bases de données sont présentées dans le tableau I à la page 35. Le premier type de graphique présente l'évolution de l'archive pour toutes les itérations de l'algorithme. Le deuxième type de graphique montre quatre agrandissements selon les quatre valeurs les plus intéressantes de la fonction f_1. Les solutions sont données par tranche de 25 itérations. Pour terminer, le troisième type de graphique présente, encore par tranche de 25 itérations, l'archive complète.

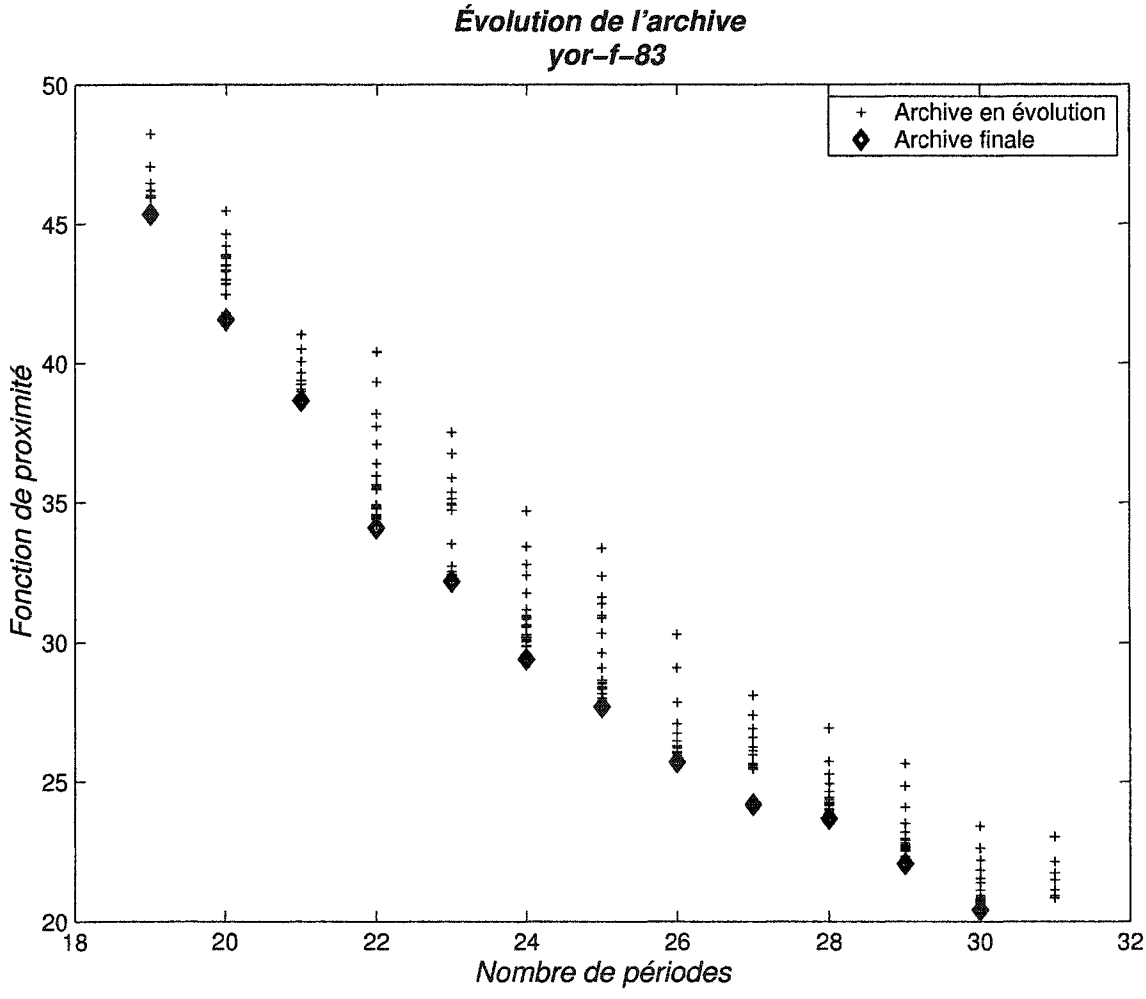

Figure 23 Évolution de l'archive *yor-f-83* (P2) : vue d'ensemble

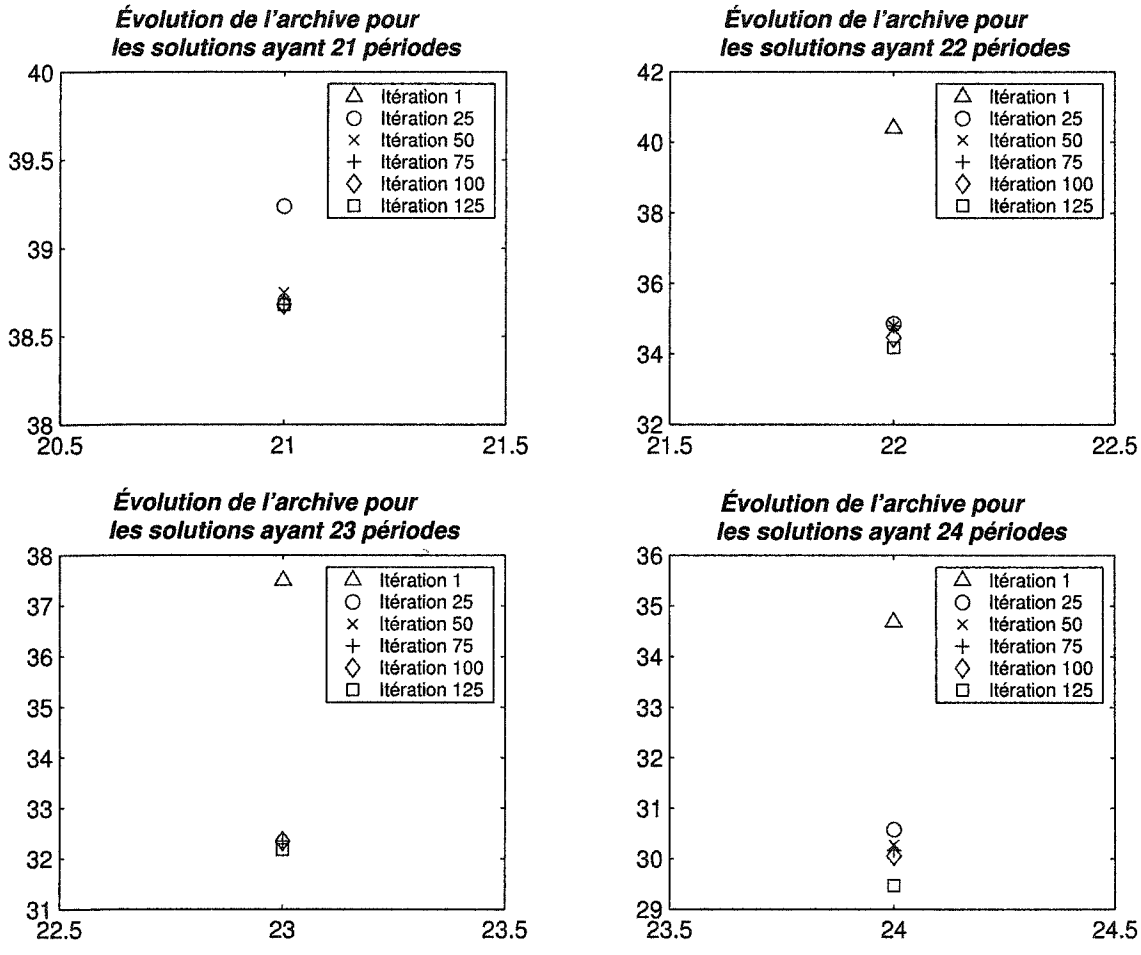

Figure 24 Évolution de l'archive *yor-f-83* (P2) : agrandissement par période

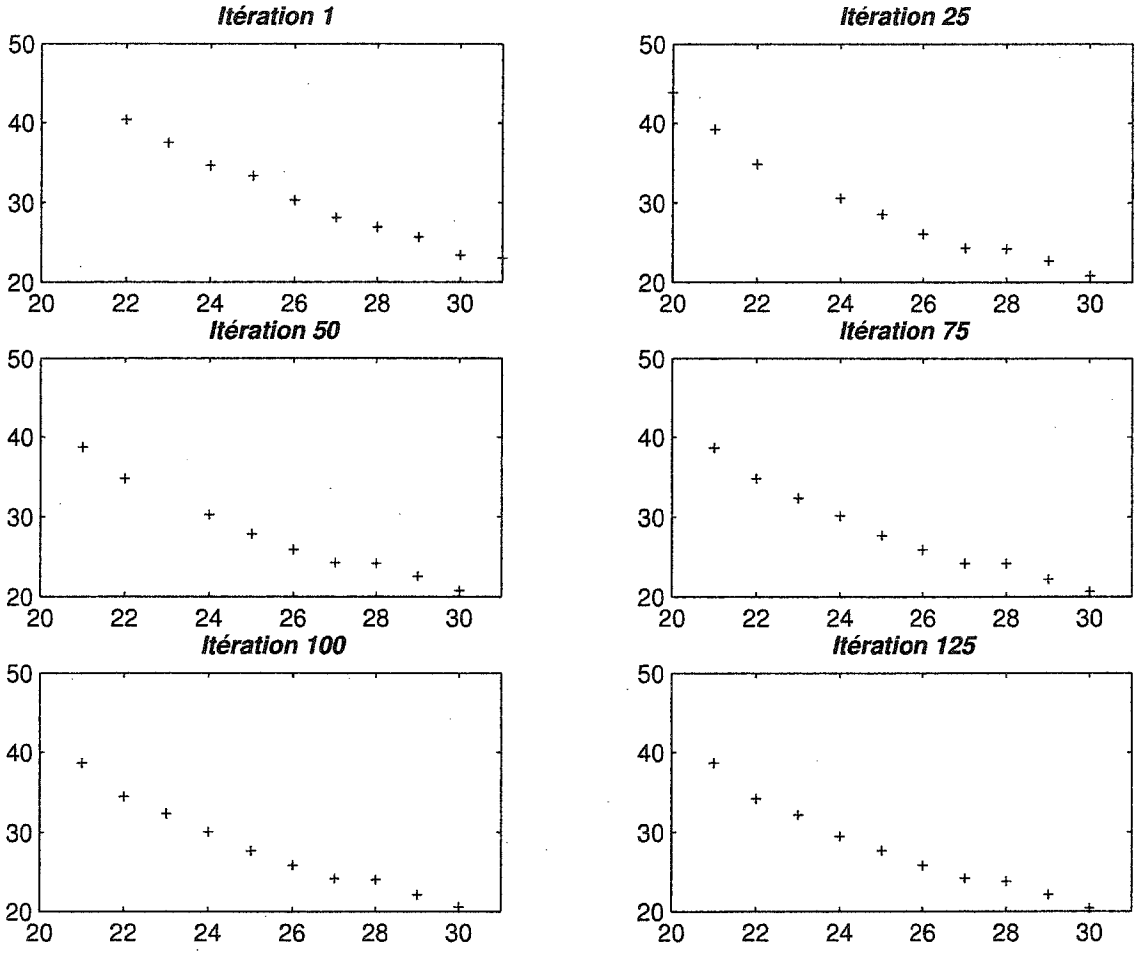

Figure 25 Évolution de l'archive *yor-f-83* (P2) : tranche de 25 itérations

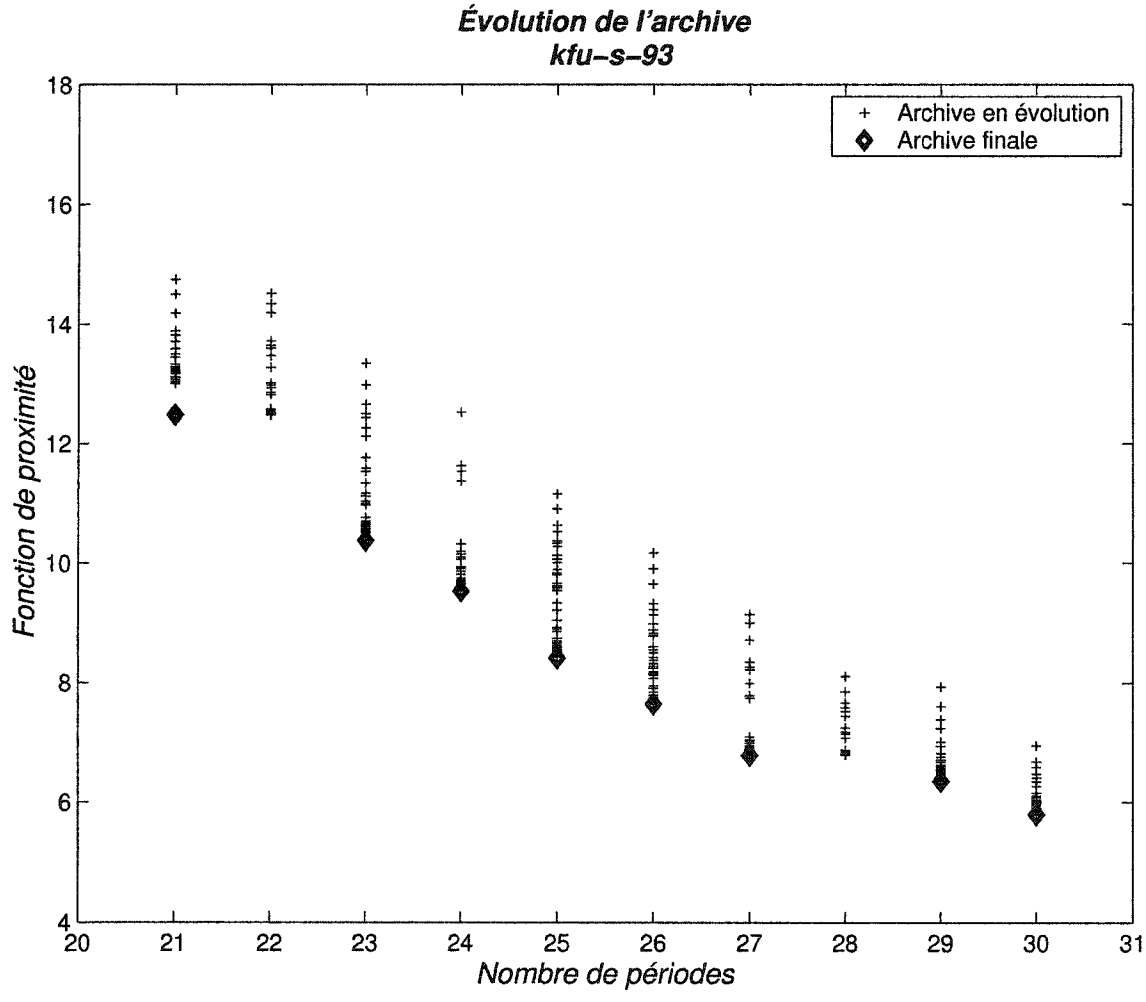

Figure 26 Évolution de l'archive *kfu-s-93* (P2) : vue d'ensemble

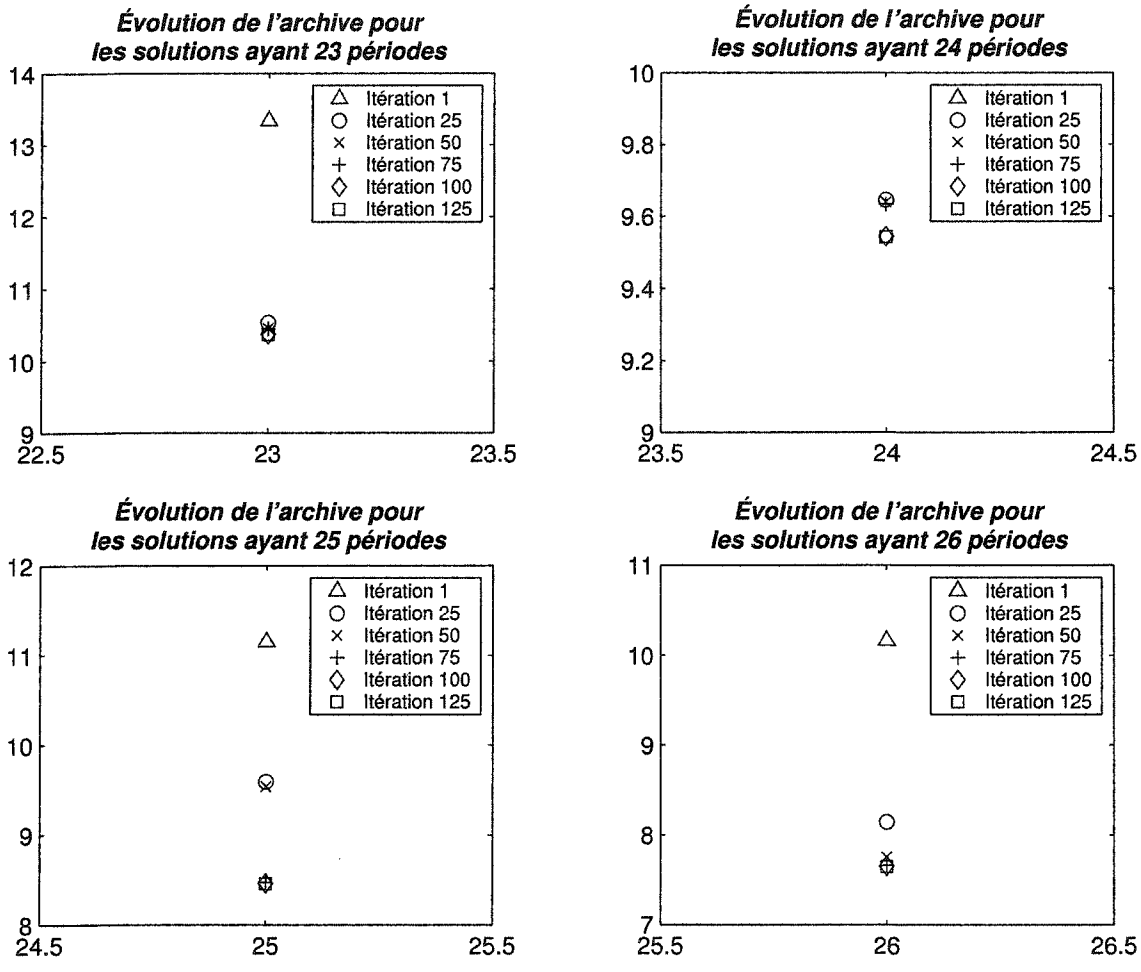

Figure 27 Évolution de l'archive *kfu-s-93* (P2) : agrandissement par période

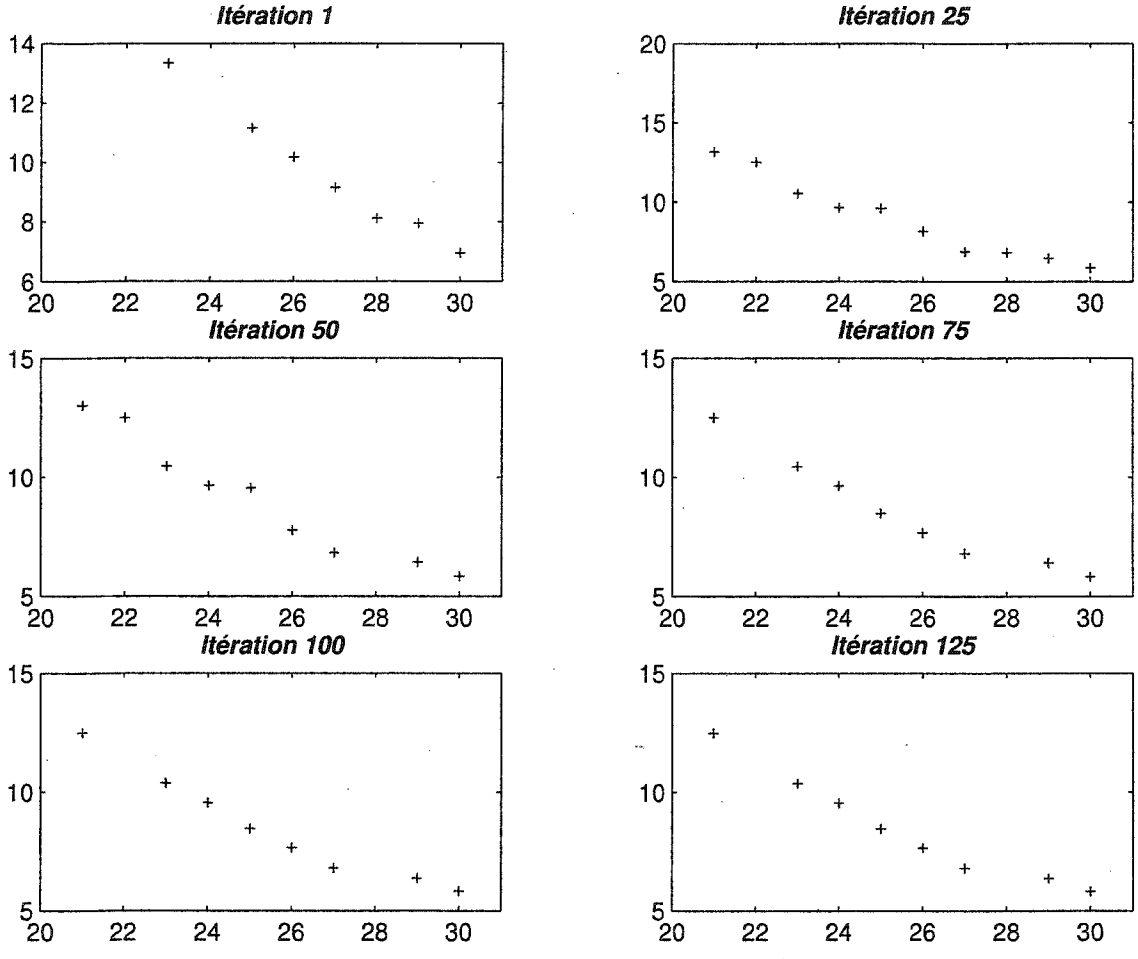

Figure 28 Évolution de l'archive *kfu-s-93* (P2) : tranche de 25 itérations

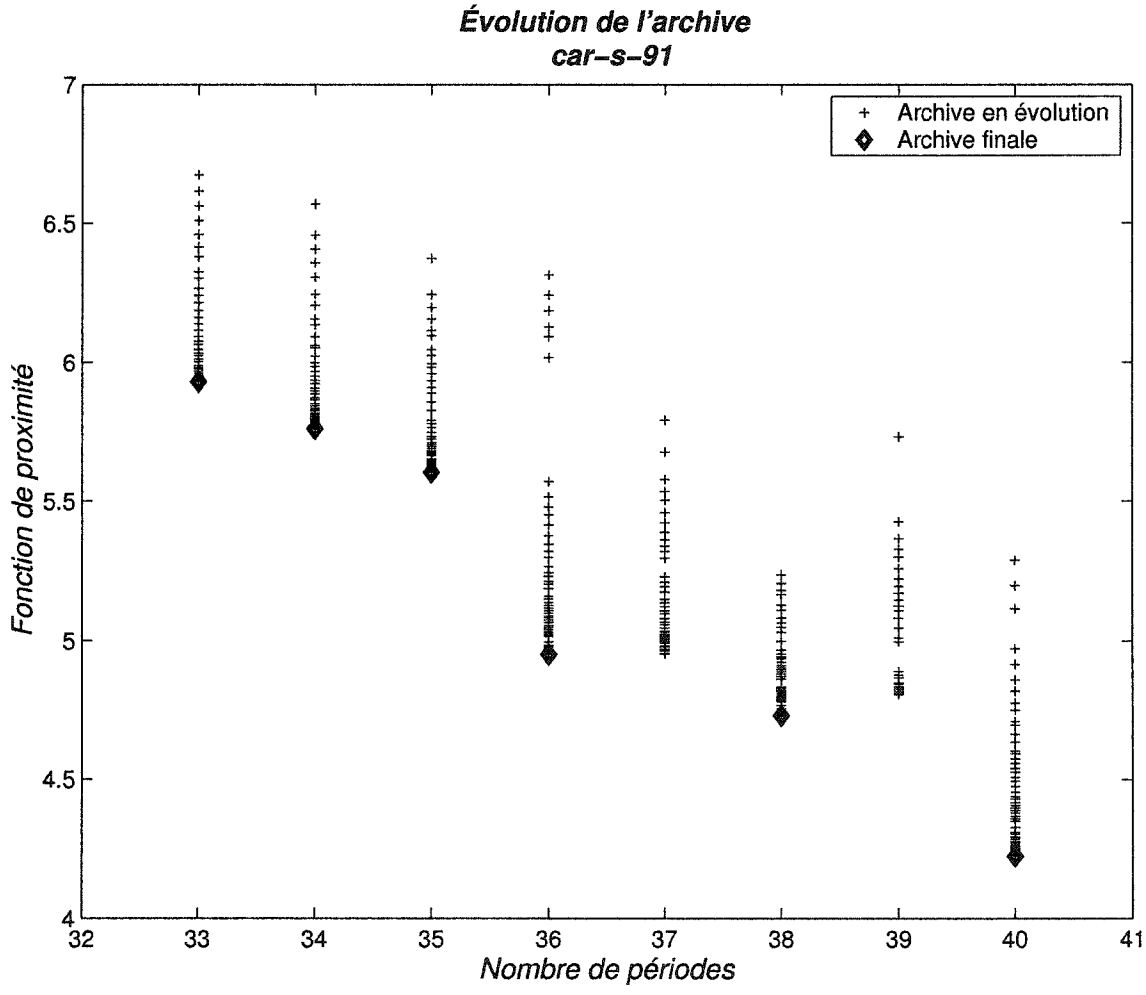

Figure 29 Évolution de l'archive *car-s-91* (P2) : vue d'ensemble

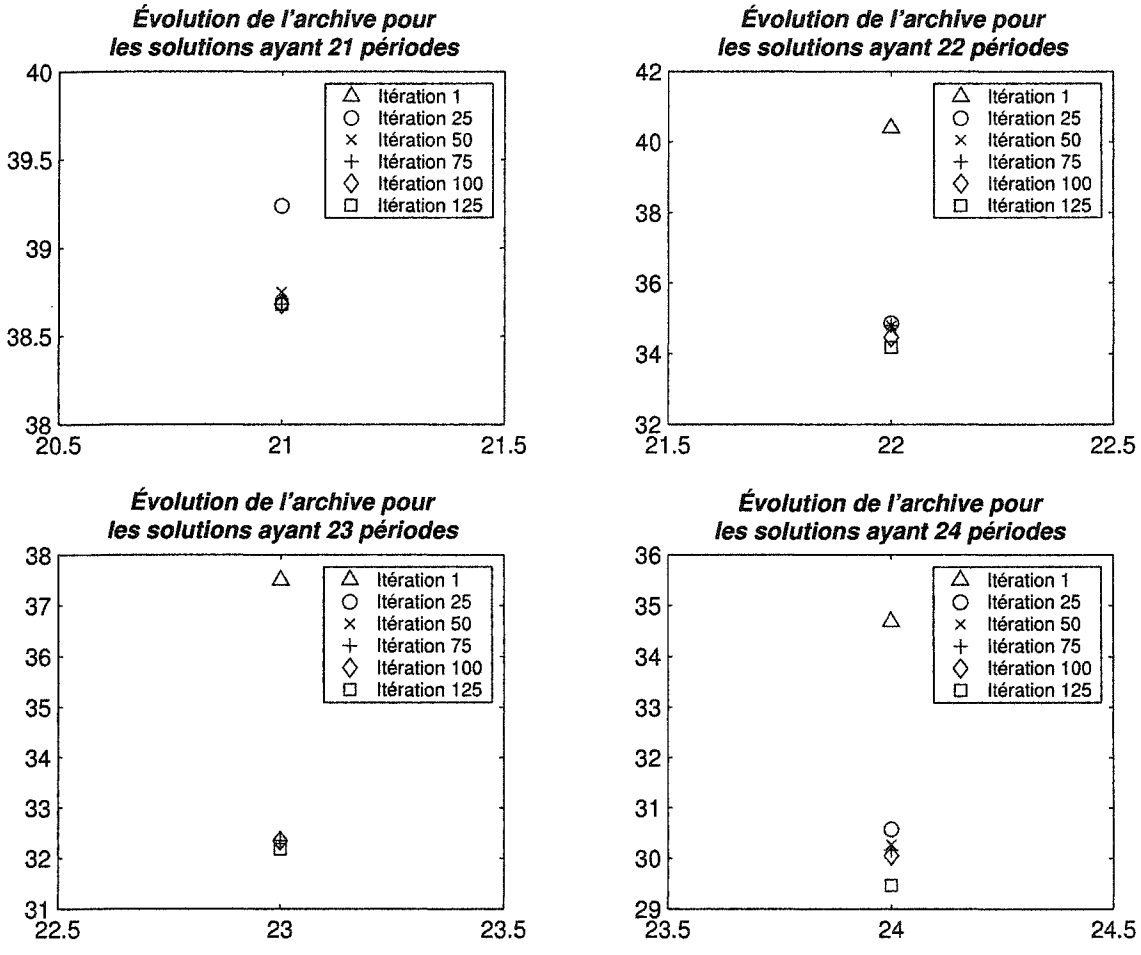

Figure 30 Évolution de l'archive *car-s-91* (P2) : agrandissement par période

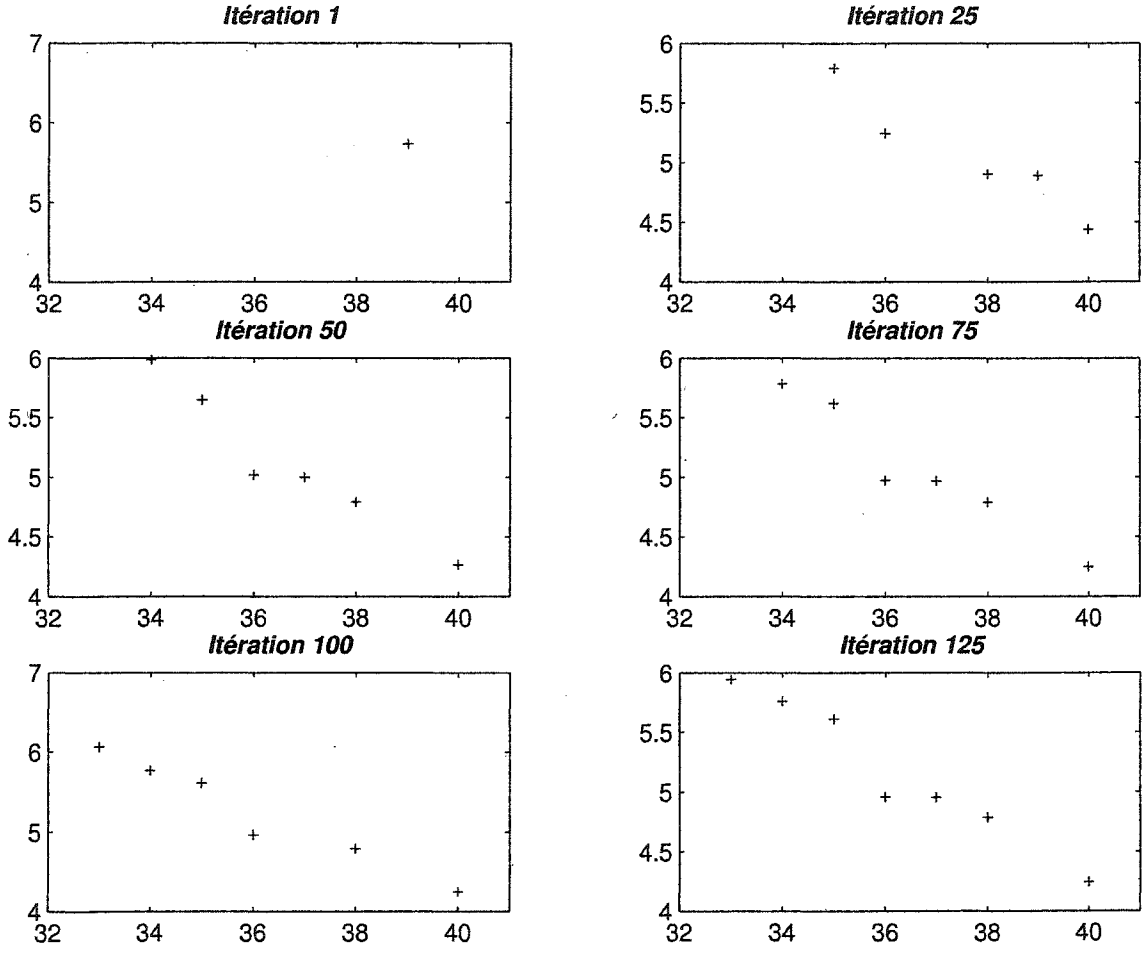

Figure 31 Évolution de l'archive *car-s-91* (P2) : tranche de 25 itérations

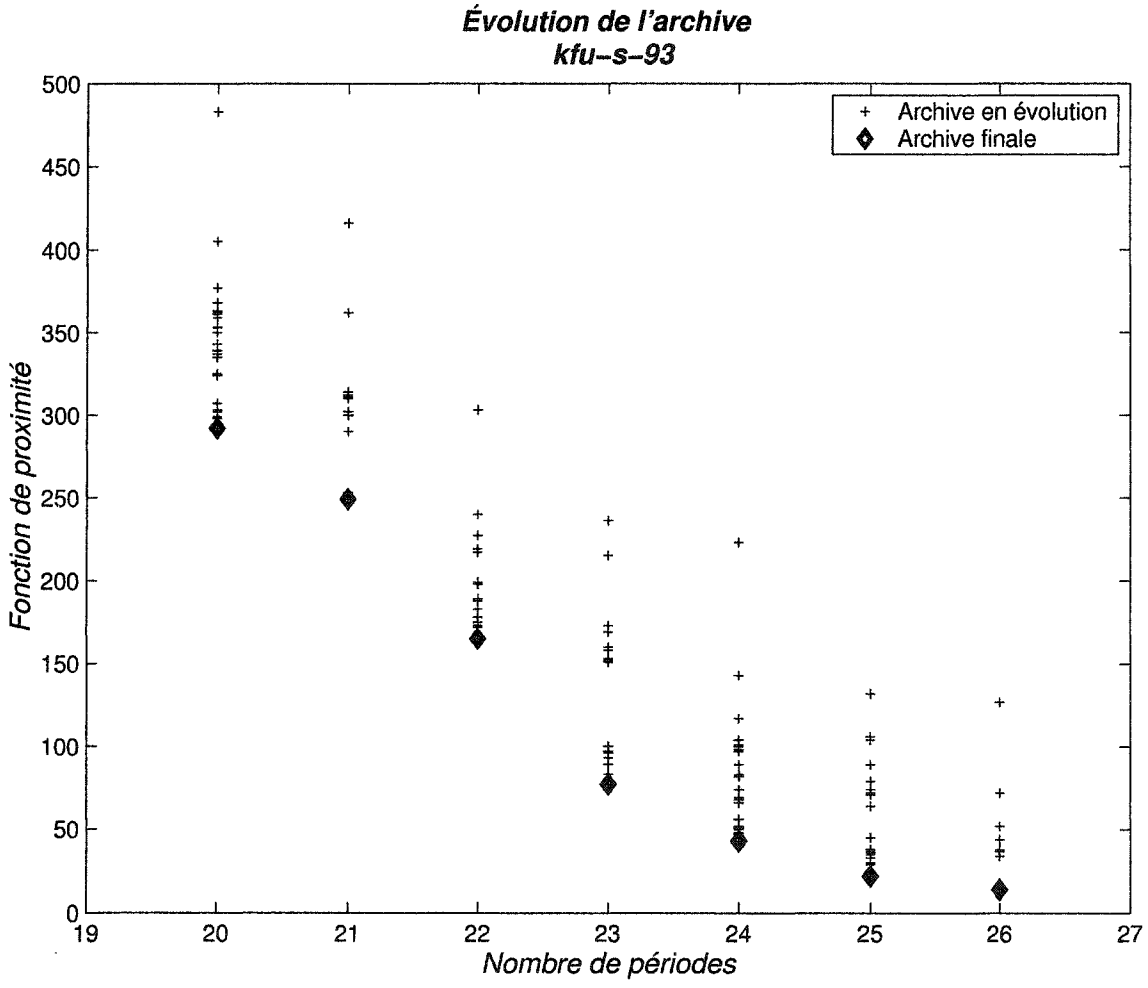

Figure 32 Évolution de l'archive *kfu-s-93* (P3) : vue d'ensemble

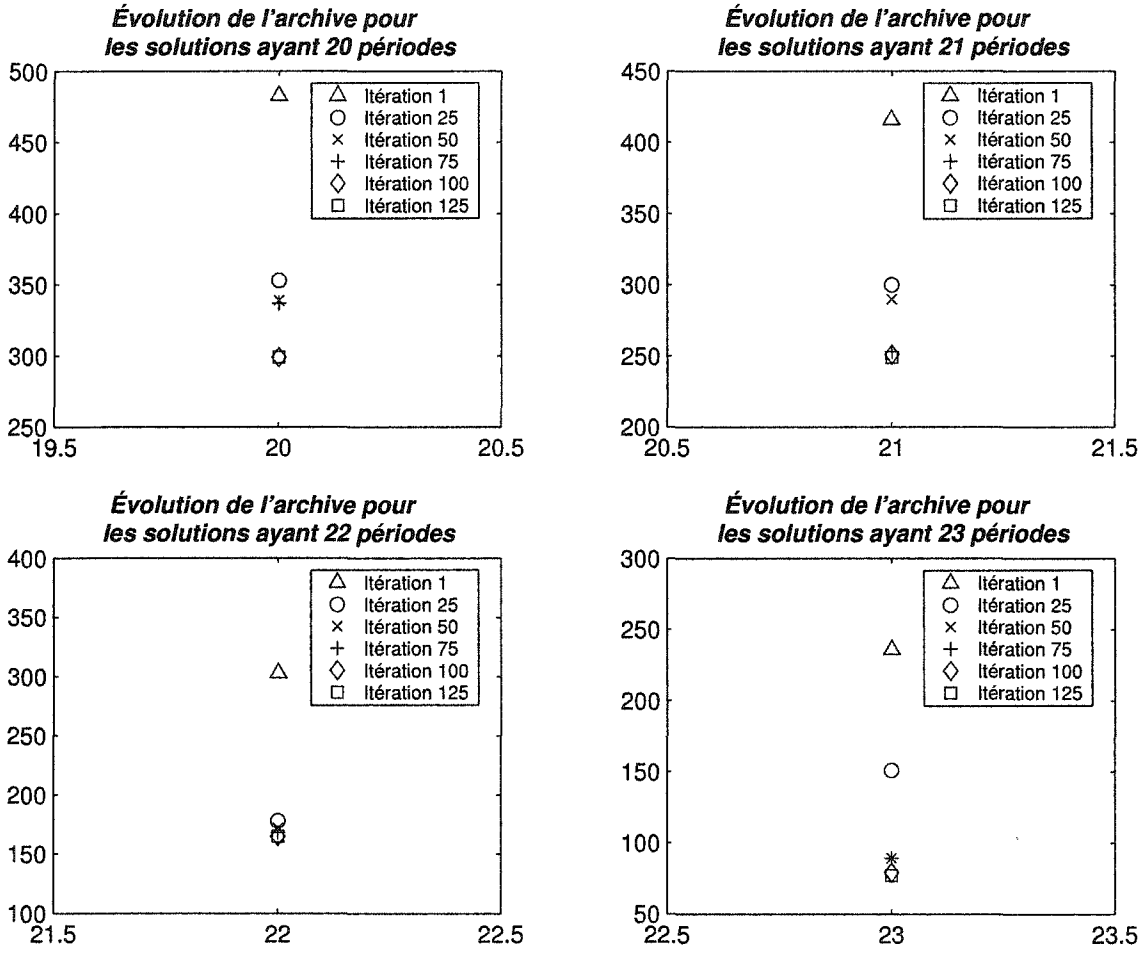

Figure 33 Évolution de l'archive *kfu-s-93* (P3) : agrandissement par période

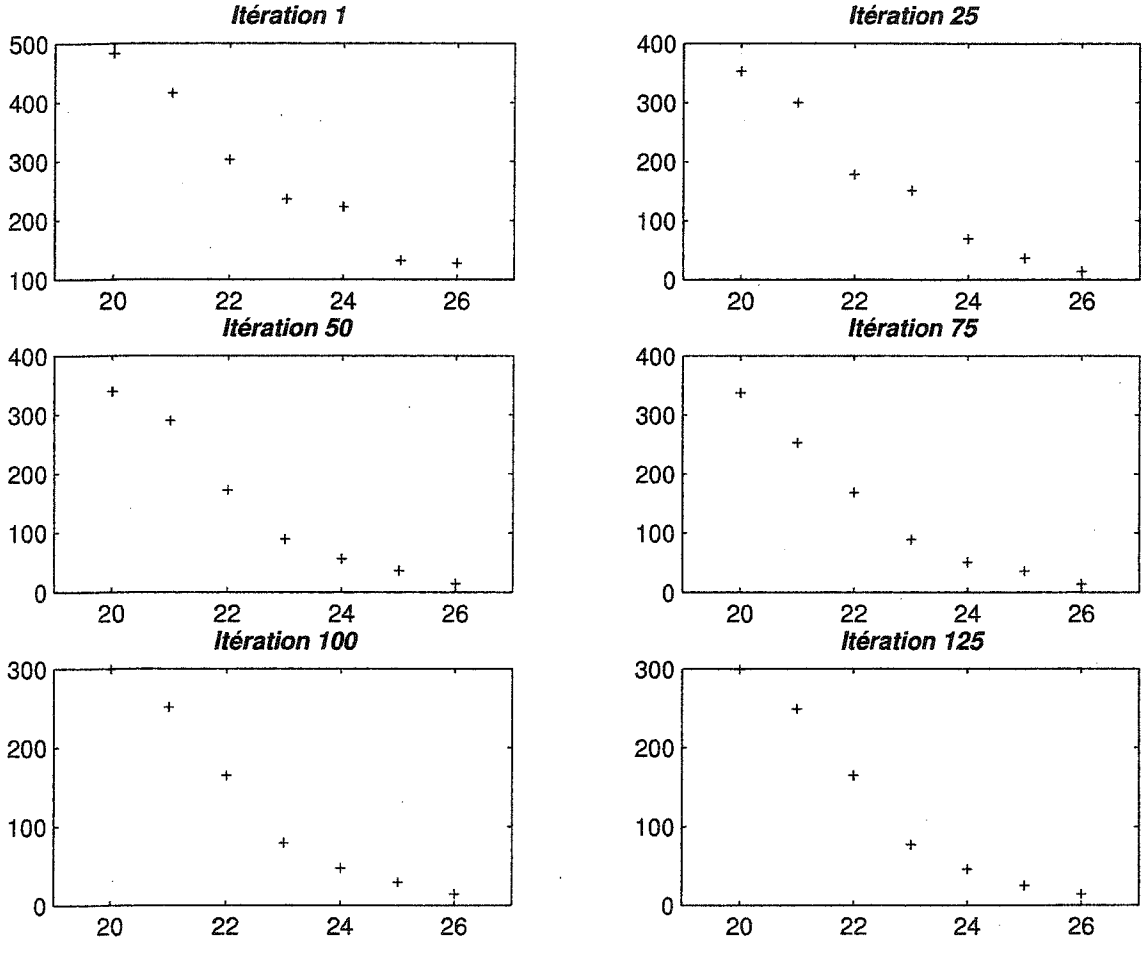

Figure 34 Évolution de l'archive *kfu-s-93* (P3) : tranche de 25 itérations

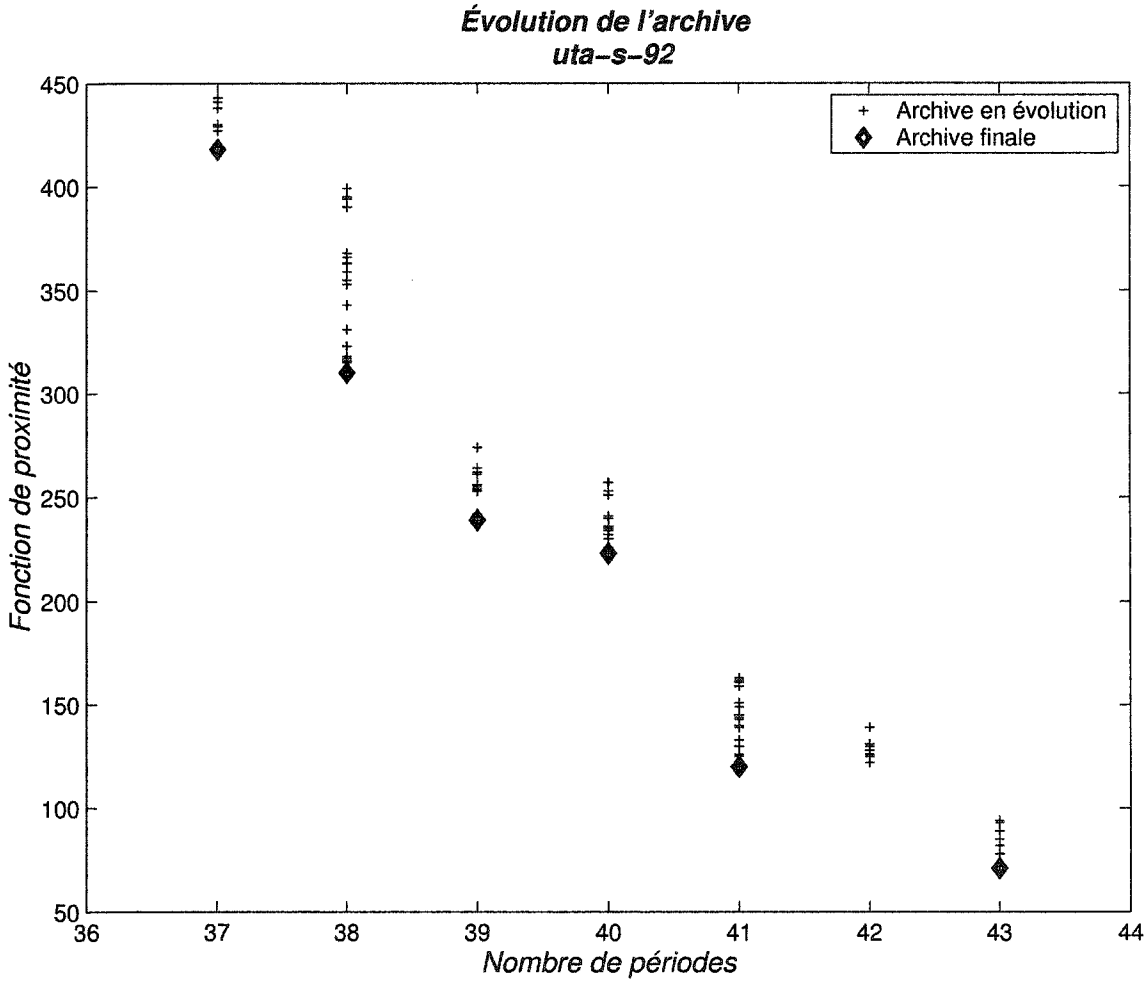

Figure 35 Évolution de l'archive *uta-s-92* (P3) : vue d'ensemble

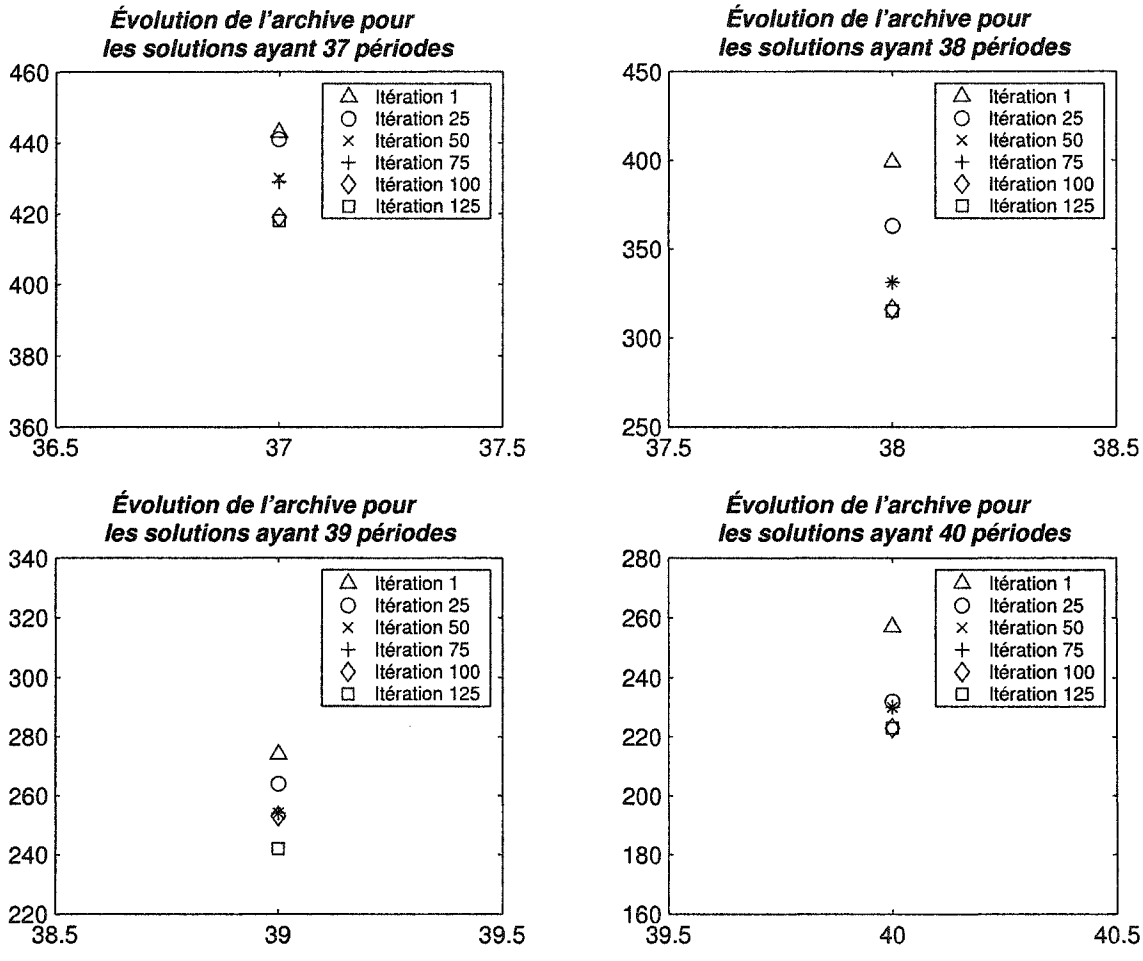

Figure 36 Évolution de l'archive *uta-s-92* (P3) : agrandissement par période

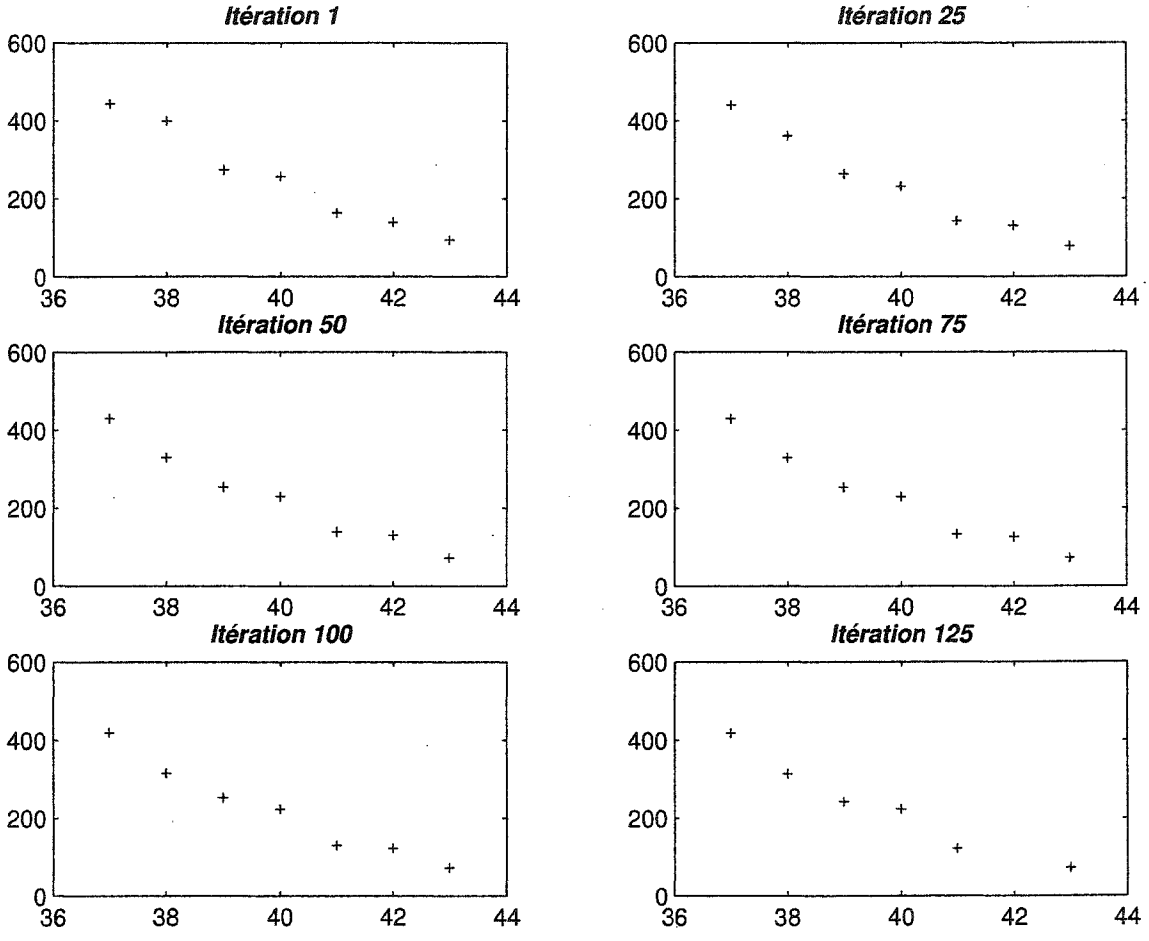

Figure 37 Évolution de l'archive *uta-s-92* (P3) : tranche de 25 itérations

BIBLIOGRAPHIE

[1] Côté, P., Wong, T., Sabourin, R. (2004) *Application of a Hybrid Multi-Objective Evolutionary Algorithm to the Uncapacited Exam proximity Problem* Practice and Theory of Automated Timetabling (PATAT2004).

[2] Wong, T., Côté, P., Sabourin, R. (2004) *A Hybrid MOEA for the Capacited Exam Proximity Problem* Congress on Evolutionary Computation (CEC2004).

[3] Burke, E.K., Petrovic, S., (2002) *Recent research directions in automated timetabling* European Journal of Operation Research , Volume : 140, 266-280.

[4] Calaor,A., Hermosilla,A.Y., Corpus, Jr,B.O. (2002) *Parallel Hybrid Adventures with Simulated Annealing and Genetic Algorithms*, International Symposium on Parallel Architectures, Algorithms and Networks (ISPAN '02).

[5] L.T.G. Merlot, N. Boland, B.D. Hughes and P.J. Stuckey. (2002) *A Hybrid Algorithm for the Examination Timetabling Problem.* Dans : Burke, E. ; De Causmaecker, P. (eds.) : Proceedings of the Fourth International Conference on the Practice and Theory of Automated Timetabling, Gent, Belgium, 348-371.

[6] Université de Melbourne. *Operation Research Group*, [En ligne]. www.or.ms.unimelb.edu.au/timetabling/ttframe.html ?ttexp4.html (Consulté le 16 juillet 2004).

[7] Paquete, L., Stutzle, T., (2002) (Empirical Analysis of Tabu Search for the Lexicographic Optimization of the Examination Timetabling Problem). Dans : Burke, E., De Causmaecker, P., (eds.) : (Proceedings of the Fourth International Conference on the Pratice and Theory of Automated Timetabling), Gent, Belique, 413-420.

[8] Caramia, M., Dell'olmo, P., Italiano, G.F. (2001) *New Algorithms for Examination Timetabling*, Lecture Notes in Computer Science. v. 1982, 230-243.

[9] Deb, K., (2001) *Multi-Objective Optimization using Evolutionary Algorithms*, Chichester : John Wiley & Sons.

[10] Deb, K. and Goel, T. (2001) *Controlled elitist non-dominated sorting genetic algorithms for better convergence*, Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization (EMO-2001), Zurich, Switzerland, 67-81.

[11] Gaspero, L.D., Schaerf, A.(2001) *Tabu Search Techniques for Examination Timetabling*, Lecture Notes in Computer Science. v.2079, 104-128.

[12] Venkararaman, P., (2001) *Applied Optimization with Matlab Programming*, John Wiley & Sons.

[13] Zitzler, E., Laumanns, M., Thiele, L.(2001) *SPEA2 : Improving the Performance of the Strength Pareto Evolutionary Algorithm*, Technical Report 103, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich.

[14] Deb, K., Agrawal, S., Pratap, A., Meyarivan, T. (2000) *A Fast Elitist Non-dominated sorting genetic algorithm for multi-objective optimization : NSGA-II*, Proceedings of the Parallel Problem Solving from Nature VI Conference, Paris, France, 849-858.

[15] Gottlieb, J., (2000) *Evolutionary algorithms for constrained optimization problems*, Allemagne : Springer-Verlag Berlin.

[16] Herrera, F., Lozano, M. (2000) *Gradual Distributed real-Coded Genetic Algorithms*,IEEE Transactions on Evolutionary Computation, volume : 4, no.1, 43-63.

[17] Pham, D.T., Karaboga, D. (2000) *Intelligent optimisation techniques : genetic algorithms, Tabu search, simulated anneling and neural networks* Éditeur : New York, N.Y. : Springer-Verlag.

[18] G.M. White and B.S. Xie. *Examination timetables and tabu search with longer-term memory.* Dans : Burke E. ; Erben W. (eds.) : PATAT 2000, Konstanz, Germany, August 16-18, 2000. Selected Papers. Lecture Notes in Computer Science 2079 Springer-Verlag, Berlin Heidelberg 85-103.

[19] Burke, E.K., Newall, J.P. (1999) *A multistage evolutionary algorithm for the timetable problem*, Evolutionary Computation, IEEE Transactions on , Vol.3 Issue : 1, 63 -74.

[20] Chu, S.C., Fang, H.L. (1999) *Genetic algorithms vs. Tabu search in timetable scheduling*, Knowledge-Based Intelligent Information Engineering Systems, 1999. Third International Conference, 492-495.

[21] Eiben, A.E., Hinterding, R., Michalewicz, Z. (1999) *Parameter Control in Evolutionary Algorithms*, IEEE Trans on Evolutionary computation, Vol 3, No 2, 124-141.

[22] Zitzler, E.(1999) *Evolutionary Algorithms for Multiobjective Optimization : Methods and Applications*, Swiss Federal Institute of Technology (ETH) Zurich. TIK-Schriftenreihe Nr. 30, Diss ETH No. 13398, Germany : Shaker Verlag.

[23] Herrera, F., Lozano, M., Verdegay, J.L. (1998) *Tackling Real-Coded Genetic Algorithms : Operators and Tools for Behavioural Analysis*, Artificial Intelligence Rev., vol. 12, no.4, 265-319.

[24] Nurmi, K., (1998) *Genetic Algorithms for Timetabling and Traveling Salesman Problems*, University of Turku : , Ph.D. thesis, Faculty of Mathematics and Natural Sciences of the University of Turku.

[25] Zitzler, E., Thiele, L. (1998) *An Evolutionary Algorithm for Multiobjective Optimization : The Strength Pareto Approach*, Technical Report 43, Computer Engineering and Communication Networks Lab (TIK), Swiss Federal Institute of Technology (ETH) Zurich, Gloriastrasse 35, CH-8092 Zurich.

[26] Glover, F.W.,(1997) *Tabu Search*, Manuel Laguna, Kluwer Academic Publishers, Boston Mass.

[27] de Werra,D. (1997) *The combinatorics of timetabling*, European Journal of Operational Research, Vol : 96, Issue : 3, February 1, 504-513.

[28] Burke, E. K., Newall, J.P., Weare, R., (1996) *A Memetic Algorithm for University Exam Timetabling*,1st International Conference on the Practice and Theory of Automated Timetabling (ICPTAT'95), Napier University, Edinburgh, UK.

[29] Carter, G., Laporte, G., Lee, S.t., (1996) *Examination timetabling : algorithmic strategies and applications*, Journal of the Operational Research Society. 373-383.

[30] Srinivas, N. and Deb, K. (1995) *Multiobjective function optimization using non-dominated sorting genetic algorithms*, Evolutionary Computation Journal, 221-248.

[31] Burke, E. K., Elliman, D.G., Weare, R., (1994) *A Genetic Algorithm for University Timetabling*, AISB Workshopon Evolutionary Computing, University of Leeds, UK, Society for the Study of Artificial Intelligence and Simulation of Behaviour (SSAISB).

[32] Gendreau, M., Hertz, A., Laporte, G. (1994) *A tabu Heuristic for véhicle Routing Problem*, Manage. Sci. 40, 1276-1290.

[33] Morgenstern, C., Shapiro, H. (1990) *Coloration neighborhood structures for general graph coloring*, In Proceedings of the First Annual ACM-SIAM Symposium on Discrete Algorithms, San Francisco, California, 226-235.