Université de Technology de
Belfort-Montbéliard

Projet AI52

Problème d'optimisation combinatoire

Réalisé par :

CHAUSSON Thibault

GUINOT Jossua

VIGUIER Léo

Dirigé par : DRIDI Mahjoub

13 janvier 2024

Résumé

Au travers de ce dossier cinq méthodes différentes de résolution d'emplois du temps ont été réalisées par des métaheuristiques tel que :

— les algorithmes génétiques,
— les colonies de fourmis,
— la recherche taboue,
— le recuit simulé,
— les essaims particulaires.

Nous constaterons que certaines de ces méthodes sont bien plus adaptées pour la résolution de ce problème.

Mots clés Recherche opérationnelle, métaheuristique, algorithme génétique, colonie de fourmis, recherche taboue, recuit simulé, essaim particulaire, emploi du temps

Table des matières

1 Présentation

Dans le cadre de nos études d'ingénieur en informatique, nous réalisons un projet sur la réalisation automatisée et optimale d'un emploi du temps pour l'UTBM[3]. Pour ce faire, nous avons mis en place une stratégie de résolution basée sur les algorithmes génétiques[1], voulant explorer d'autres horizons, nous avons décidé de continuer cette étude en utilisant d'autres métaheuristiques, tels que le recuit simulé, la recherche taboue, ...

Ainsi, nous réaliserons :

— Modéliser mathématiquement le problème (critère à optimiser, contraintes, variables,..)

— Définir les paramètres de la métaheuristique (exemple codage, croisement et mutation pour AG : Température, facteur de décroissance,... pour RS : Voisinage, taille de liste taboue, mouvements tabous,... pour RT : nombre de fourmis, facteur de visibilité, de trace, coefficient d'évaporation,... pour ACO : inertie de particule, facteurs d'attraction, positions, vitesses,... pour PSO)

— Exécuter quelques itérations de la méthode sur une instance de petite taille. Il ne sagit pas forcément de faire le programme pour chaque méthode, mais il sagit plutôt de montrer.

2 Définition du problème

2.1 Planning

L'objectif de ce projet est de présenter les métaheuristiques suivantes sur l'optimisation d'emploi du temps :

1. Algorithme génétique
2. Optimisation par colonies de fournies
3. Optimisation par essaim particulaire
4. Recuit simulé
5. Recherche tabou

Pour ce faire, nous allons définir le problème de la manière suivante. Soient, $n \in \mathbb{N}^{*+}$, $s \in [\![1, 7]\!]$, dans notre cas $s = 5$ qui sera le nombre de jours de la semaine, $c \in [\![1, n]\!]$, le nombre de cours disponible, avec $n = 10$. Prenons la matrice creuse $M \in \mathbb{M}_{s,c}$, ou les 1 représenter les cours dispensés ce jour :

$$M = \begin{array}{c|c} \text{Cours} \downarrow & \text{Jours de la semaine} \rightarrow \\ & \begin{array}{ccccc} Lundi & Mardi & Mercredi & Jeudi & Vendredi \end{array} \\ \begin{array}{l} 1 : GE41 \\ 2 : CC01 \\ 3 : AI50 \\ 4 : AP4A \\ 5 : EI03 \\ 6 : LJ00 \\ 7 : SY41 \\ 8 : SI40 \\ 9 : IT42 \\ 10 : WE4A \end{array} & \begin{pmatrix} 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \end{pmatrix} \end{array}$$

Pour simplifier les différentes exécutions de ce problème, nous considérons qu'un cours dure une journée entière. De plus, nous avons suffisamment de salle de cours et de professeurs pour dispenser l'ensemble de ces cours.

2.2 Classes

Nous considérons un ensemble $\mathbb{C}^d$ de $d \in \mathbb{N}^{*+}$ classe constitué de $e \in \mathbb{N}^{*+}$ vœux de cours, comme cela peu se faire à l'UTBM. Donc la liste des vœux est de la forme $(\mathbb{C}^d)^e$. Ici $d = 5$ et $e = 3$. Représenté de la manière suivante :

Classe	Cours
Classe 1	1 : GE41, 3 : AI50, 6 : LJ00
Classe 2	...
Classe 3	...
Classe 4	...
Classe 5	...

Nous considérons que deux classes peuvent avoir le même cours, ainsi la direction aux formations pédagogiques affectera assez de professeurs.

2.3 Contraintes

Les contraintes fortes :
— Tous les cours voulus par les classes dans l'emploi du temps
— Au moins $e = 3$ jours avec des cours dispensés, car les étudiants doivent participer à $e = 3$ cours
La contrainte faible :
— Satisfaire les vœux de tous les étudiants (classes)

2.4 Fitness

Nous définissions la $fitness$ comme suit : p_i le score d'une classe, avec $p \in [\![0, 2]\!]$ et $i \in [\![1, d]\!]$. Si :
— Aucun cours en même temps $p_i = 2$
— Deux cours en même temps $p_i = 1$
— Trois cours en même temps $p_i = 0$
Donc, nous avons :

$$fitness = \sum_{i=1}^{d} p_i \tag{1}$$

Ainsi, nous avons $\max(fitness) = 2 \times d$, donc $\max(fitness) = 10$

2.5 Une solution

Prenons les vœux des classes suivantes :

Classe	Cours
Classe 1	1 : GE41, 3 : AI50, 6 : LJ00
Classe 2	4 : AP4A, 5 : EI03, 9 : IT42
Classe 3	3 : AI 50, 4 : AP4A, 8 : SI40
Classe 4	2 : CC01, 4 : AP4A, 5 : EI03
Classe 5	2 : CC01, 3 : AI50, 10 : WE4A

Le planning suivant :

$$M = \begin{array}{c} \\ \text{Cours} \downarrow \end{array} \begin{array}{c} \\ 1 : GE41 \\ 2 : CC01 \\ 3 : AI50 \\ 4 : AP4A \\ 5 : EI03 \\ 6 : LJ00 \\ 7 : SY41 \\ 8 : SI40 \\ 9 : IT42 \\ 10 : WE4A \end{array} \begin{array}{c} \text{Jours de la semaine} \rightarrow \\ \begin{array}{ccccc} Lundi & Mardi & Mercredi & Jeudi & Vendredi \end{array} \\ \begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array}$$

Ainsi, nous avons la fitness suivante :

$$fitness = \sum_{i=1}^{d} p_i = 2 + 2 + 2 + 2 + 2 = 10 = \max(fitness) \tag{2}$$

Donc, ce planning optimise les vœux des étudiants.

2.6 Généralisation du problème

Voici une représentation sous forme de diagramme UML de toutes les données que nous pouvons utiliser pour générer un planning.

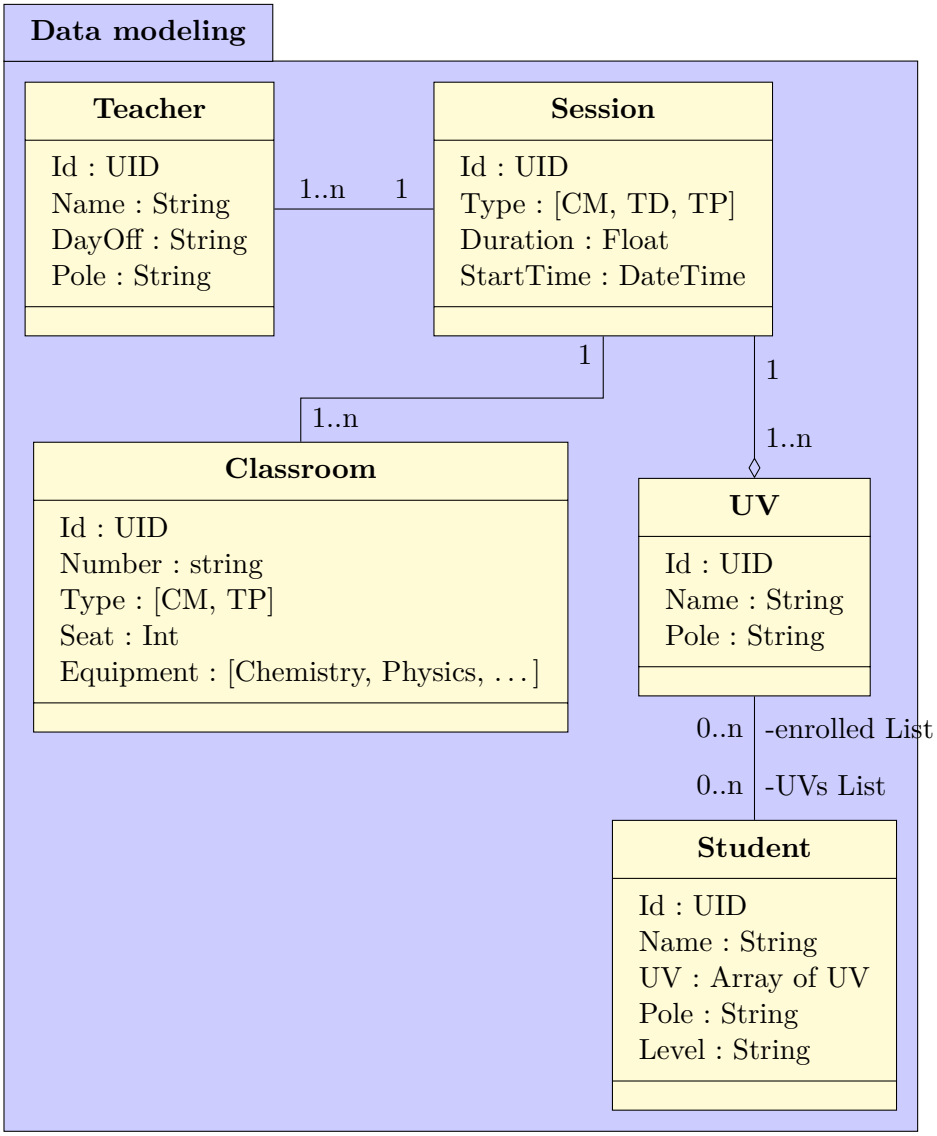


Image 1 – UML data

3 Algorithme Génétique

Pour réaliser cette métaheurisitique, nous devons fixer :
— La population initiale de 6 individus
— Le taux de mutation et la méthode de mutation
— La méthode de croisement
Comme vu plus haut la valeur maximale de la fitness est de 10.

Nous avons 10 UVs réparties sur 5 jours, chaque UV peut être placée sur 5 jours donc l'espace de solution est de dimension $nb\_jours^{nb\_UVs} = 5^{10} = 9765625$.

3.1 Vœux des étudiants

Classes	1	2	3	4	5
UVs	1, 2, 3	4, 5, 9	3, 4, 8	2, 4, 5	2, 3, 10

Tableau 1 – Voeux des étudiants

3.2 Population initiale

Tirons de manière aléatoire la répartition des UVs par jours :

Parents \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
1	Ma	L	L	V	Me	J	J	V	Me	L	5
2	V	L	V	Ma	Me	V	L	V	Ma	Me	7
3	J	L	Ma	Ma	V	Me	Ma	Me	L	Ma	8
4	Me	Ma	L	J	V	J	L	V	J	V	9
5	Ma	Me	Me	Me	V	Ma	L	Ma	V	Ma	5
6	Me	L	J	J	L	L	J	J	Ma	L	6

Tableau 2 – Population initiale (chromosomes)

Jours	L	Ma	Me	J	V
Parents 1	2, 3, 10	1	5, 9	6, 7	4, 8
Parents 2	2, 7	4, 9	5, 10	_	1, 3, 6, 8
Parents 3	2, 9	3, 4, 7, 10	6, 8	1	5
Parents 4	3, 7	2	1	4, 6, 9	5, 8, 10
Parents 5	7	1, 6, 8, 10	2, 3, 4	_	5, 9
Parents 6	2, 5, 6, 10	9	1	3, 4, 7, 8	_

Tableau 3 – Population initiale (jours)

Le parent 1 a les conflit suivant :

1. 2 cours en même temps donc $p_1 = 1$
2. 2 cours en même temps donc $p_2 = 1$
3. 2 cours en même temps donc $p_3 = 1$
4. 0 cours en même temps donc $p_4 = 2$
5. 3 cours en même temps donc $p_5 = 0$

Donc, la fitness est de 5.

Le parent 2 a les conflit suivant :

1. 2 cours en même temps donc $p_1 = 1$
2. 2 cours en même temps donc $p_2 = 1$
3. 2 cours en même temps donc $p_3 = 1$
4. 0 cours en même temps donc $p_4 = 2$
5. 0 cours en même temps donc $p_5 = 2$

Donc, la fitness est de 7.

Le parent 3 a les conflit suivant :

1. 0 cours en même temps donc $p_1 = 2$
2. 0 cours en même temps donc $p_2 = 2$
3. 2 cours en même temps donc $p_3 = 1$
4. 0 cours en même temps donc $p_4 = 2$
5. 2 cours en même temps donc $p_5 = 1$

Donc, la fitness est de 8.

Le parent 4 a les conflit suivant :

1. 0 cours en même temps donc $p_1 = 2$
2. 2 cours en même temps donc $p_2 = 2$

3. 0 cours en même temps donc $p_3 = 3$
4. 0 cours en même temps donc $p_4 = 2$
5. 0 cours en même temps donc $p_5 = 3$

Donc, la fitness est de 9.

Le parent 5 a les conflit suivant :

1. 2 cours en même temps donc $p_1 = 1$
2. 2 cours en même temps donc $p_2 = 1$
3. 2 cours en même temps donc $p_3 = 1$
4. 2 cours en même temps donc $p_4 = 1$
5. 2 cours en même temps donc $p_5 = 1$

Donc, la fitness est de 5.

Le parent 6 a les conflit suivant :

1. 0 cours en même temps donc $p_1 = 2$
2. 0 cours en même temps donc $p_2 = 2$
3. 3 cours en même temps donc $p_3 = 0$
4. 2 cours en même temps donc $p_4 = 1$
5. 2 cours en même temps donc $p_5 = 1$

Donc, la fitness est de 6.

3.3 Méthode de sélection

Pour la sélection, nous ferons le choix d'une sélection par la méthode de la roulette.

3.4 Méthode de croisement

Nous utiliserons un croisement un point en point central.

3.5 Méthode de mutation

Nous changerons le jour d'un cours par un autre aléatoire. Notons $\mu = 0.2$ le taux de mutation.

3.6 Exécution

3.6.1 Itération 1

Sélection

Calculons la somme des fitness :

$$\sum_{i=1}^{6} fitness_i = 5 + 7 + 8 + 9 + 5 + 6 = 40 \tag{3}$$

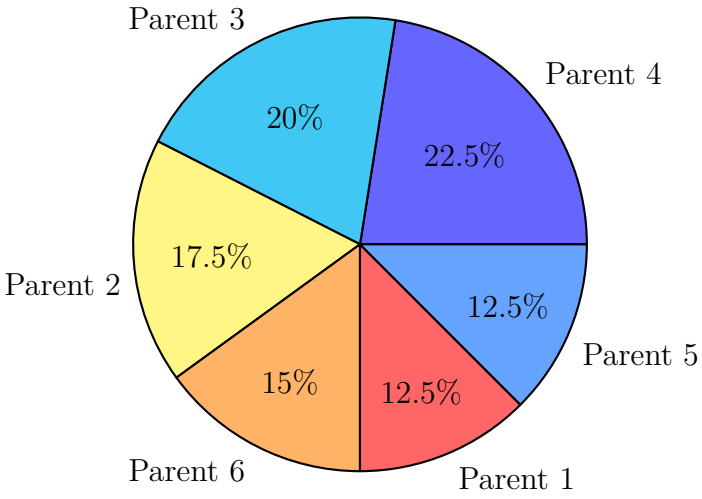


Image 2 – Roulette 1

Tirons deux nombres entre 0 et 100% pour choisir les parents :
— 91 → Parent 5
— 76 → Parent 1

Croisement

Cours / Parents	1	2	3	4	5	6	7	8	9	10	Fitness
1	Ma	L	L	V	Me	J	J	V	Me	L	5
5	Ma	Me	Me	Me	V	Ma	L	Ma	V	Ma	5
Fils 1	Ma	L	L	V	Me	Ma	L	Ma	V	Ma	7

Tableau 4 – Croisement 1 (chromosomes)

Jours	L	Ma	Me	J	V
Fils 1	2, 3, 7	1, 6, 8, 10	5	_	4, 9

Tableau 5 – Croisement 1 (jours)

Le fils 1 a les conflits suivants :

1. 2 cours en même temps donc $p_1 = 1$
2. 2 cours en même temps donc $p_2 = 1$
3. 0 cours en même temps donc $p_3 = 2$
4. 0 cours en même temps donc $p_4 = 2$
5. 2 cours en même temps donc $p_5 = 1$

Donc, la fitness est de 7.

Mutation

Tirons un nombre aléatoire pour savoir si nous devons réaliser une mutation $0.3 > \mu = 0.2$, donc pas besoin de faire de mutation.

Évaluation

Donc nous pouvons supprimer le parent 5 qui a une fitness de 5 alors que le fils 1 a une fitness de 7.

Voici la population :

Cours / Parents	1	2	3	4	5	6	7	8	9	10	Fitness
1	Ma	L	L	V	Me	J	J	V	Me	L	5
2	V	L	V	Ma	Me	V	L	V	Ma	Me	7
3	J	L	Ma	Ma	V	Me	Ma	Me	L	Ma	8
4	Me	Ma	L	J	V	J	L	V	J	V	9
Fils 1	Ma	L	L	V	Me	Ma	L	Ma	V	Ma	7
6	Me	L	J	J	L	L	J	J	Ma	L	6

Tableau 6 – Population 1 (chromosomes)

Jours	L	Ma	Me	J	V
Parents 1	2, 3, 10	1	5, 9	6, 7	4, 8
Parents 2	2, 7	4, 9	5, 10	_	1, 3, 6, 8
Parents 3	2, 9	3, 4, 7, 10	6, 8	1	5
Parents 4	3, 7	2	1	4, 6, 9	5, 8, 10
Fils 1	2, 3, 7	1, 6, 8, 10	5	_	4, 9
Parents 6	2, 5, 6, 10	9	1	3, 4, 7, 8	_

Tableau 7 – Population 1 (jours)

3.6.2 Itération 2

Sélection

Calculons la somme des fitness :

$$\sum_{i=1}^{6} fitness_i = 5 + 7 + 8 + 9 + 7 + 6 = 42 \tag{4}$$

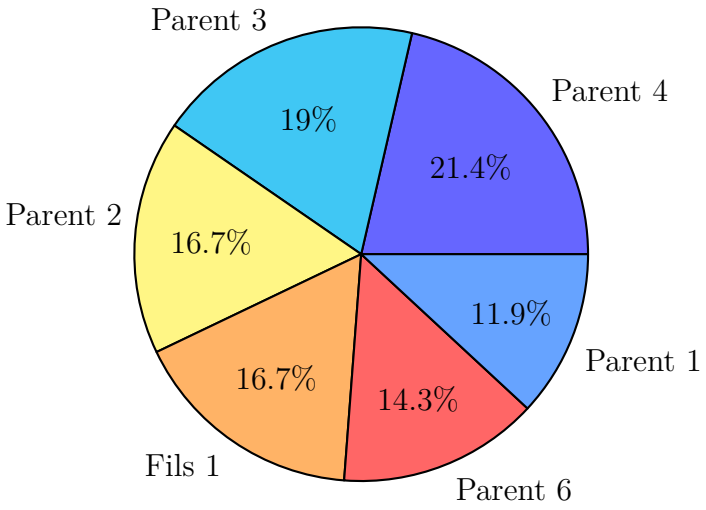


Image 3 – Roulette 2

Tirons deux nombres entre 0 et 100% pour choisir les parents :
— 26 → Parent 3
— 81 → Parent 6

Croisement

Le fils 2 a les conflits suivants :

Parents \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
3	J	L	Ma	Ma	V	Me	Ma	Me	L	Ma	8
6	Me	L	J	J	L	L	J	J	Ma	L	6
Fils 2	J	L	Ma	Ma	V	L	J	J	Ma	L	7

Tableau 8 – Croisement 2 (chromosomes)

Jours	L	Ma	Me	J	V
Fils 2	2, 6, 10	3, 4, 9	_	1, 7, 8	5

Tableau 9 – Croisement 2 (jours)

1. 0 cours en même temps donc $p_1 = 2$
2. 2 cours en même temps donc $p_2 = 1$
3. 2 cours en même temps donc $p_3 = 1$
4. 0 cours en même temps donc $p_4 = 2$
5. 2 cours en même temps donc $p_5 = 1$

Donc, la fitness est de 7.

Mutation

Tirons un nombre aléatoire pour savoir si nous devons réaliser une mutation $0.08 < \mu = 0.2$, donc nous devons faire de mutation.

Choisissons aléatoirement un cours entre 1 et 10 $\rightarrow$ 4 et affectons le à un jours aléatoire $\rightarrow$ Mercredi.

Donc nous avons :

Parents \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
Fils 2	J	L	Ma	Me	V	L	J	J	Ma	L	9

Tableau 10 – Mutation 2 (chromosome)

Jours	L	Ma	Me	J	V
Fils 2	2, 6, 10	3, 9	4	1, 7, 8	5

Tableau 11 – Mutation 2 (jours)

Le fils 2 a les conflits suivants : 0 0 0 0 1

1. 0 cours en même temps donc $p_1 = 2$
2. 0 cours en même temps donc $p_2 = 2$
3. 0 cours en même temps donc $p_3 = 2$
4. 0 cours en même temps donc $p_4 = 2$
5. 2 cours en même temps donc $p_5 = 1$

Donc, la fitness est de 9.

Évaluation

Nous supprimons le parent 1 car $fitness_{parent_1} < fitness_{fils_2}$ et $\min(fitness) = fitness_{parent_1}$ donc nous avons :

Cours / Parents	1	2	3	4	5	6	7	8	9	10	Fitness
Fils 2	J	L	Ma	Me	V	L	J	J	Ma	L	9
2	V	L	V	Ma	Me	V	L	V	Ma	Me	7
3	J	L	Ma	Ma	V	Me	Ma	Me	L	Ma	8
4	Me	Ma	L	J	V	J	L	V	J	V	9
Fils 1	Ma	L	L	V	Me	Ma	L	Ma	V	Ma	7
6	Me	L	J	J	L	L	J	J	Ma	L	6

Tableau 12 – Population 2 (chromosomes)

Jours	L	Ma	Me	J	V
Fils 2	2, 6, 10	3, 9	4	1, 7, 8	5
Parents 2	2, 7	4, 9	5, 10	_	1, 3, 6, 8
Parents 3	2, 9	3, 4, 7, 10	6, 8	1	5
Parents 4	3, 7	2	1	4, 6, 9	5, 8, 10
Fils 1	2, 3, 7	1, 6, 8, 10	5	_	4, 9
Parents 6	2, 5, 6, 10	9	1	3, 4, 7, 8	_

Tableau 13 – Population 2 (jours)

3.6.3 Bilan

Nous voyons étape après étape nous améliorons notre solution pour s'approcher de la solution optimale qui a une fitness de 10.

4 Recherche taboue

Pour réaliser cette métaheurisitique, nous devons fixer :
— La taille de la liste taboue notée T, que nous définissons à 3
— Le voisinage sera un ensemble de permutations de deux éléments successifs ce qui donne $n - 1 = 10 - 1 = 9$ en taille du voisinage.
Supposons que solution initiale S_0 soit

Solution \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
S_0	Ma	L	L	V	Me	J	J	V	Me	Ma	6

Tableau 14 – S_0 recherche taboue

Ainsi, nous avons :

Jours	L	Ma	Me	J	V
S_0	2, 3	1, 10	5, 9	6, 7	4, 8

Tableau 15 – S_0 initiale (jours)

Pour voir des cours bien équilibrés, nous décidons de répartir aléatoirement 2 cours par jour.

Nous reprenons la liste des vœux des étudiants du tableau : 1.

S_0 a les conflits suivants :

1. 2 cours en même temps donc $p_1 = 1$
2. 2 cours en même temps donc $p_2 = 1$
3. 2 cours en même temps donc $p_3 = 1$
4. 0 cours en même temps donc $p_4 = 2$
5. 1 cours en même temps donc $p_5 = 1$

Donc, la fitness est de 6.

La liste taboue est la suivante :

Solutions \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
S_0	Ma	L	L	V	Me	J	J	V	Me	Ma	6
_	_	_	_	_	_	_	_	_	_	_	_
_	_	_	_	_	_	_	_	_	_	_	_

Tableau 16 – T_0 liste taboue

4.1 Exécution

L'ensemble des exécutions ont été réalisées par un programme Python.

4.1.1 Itération 1

En utilisant l'échange de deux voisins pour avoir un voisinage, nous avons :

Voisins \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
$V_{1.1}$	L	Ma	L	V	Me	J	J	V	Me	Ma	6
$V_{1.2}$	Ma	L	L	V	Me	J	J	V	Me	Ma	6
$V_{1.3}$	Ma	L	V	L	Me	J	J	V	Me	Ma	7
$V_{1.4}$	Ma	L	L	Me	V	J	J	V	Me	Ma	7
$V_{1.5}$	Ma	L	L	V	J	Me	J	V	Me	Ma	7
$V_{1.6}$	Ma	L	L	V	Me	J	J	V	Me	Ma	6
$V_{1.7}$	Ma	L	L	V	Me	J	V	J	Me	Ma	7
$V_{1.8}$	Ma	L	L	V	Me	J	J	Me	V	Ma	7
$V_{1.9}$	Ma	L	L	V	Me	J	J	V	Ma	Me	7

Tableau 17 – Le voisinage de l'itération 1

La solution retenue est celle avec la plus grande fitness, si deux fitness maximales sont égales, alors nous prenons aléatoirement parmi l'ensemble de maximum, ici $V_{1.7}$.

Solution \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
S_1	Ma	L	L	V	Me	J	V	J	Me	Ma	7

Tableau 18 – Solution S_1

Nous ajoutons la solution S_1 à la liste taboue :

Solutions \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
S_1	Ma	L	L	V	Me	J	V	J	Me	Ma	7
S_0	Ma	L	L	V	Me	J	J	V	Me	Ma	6
—	—	—	—	—	—	—	—	—	—	—	—

Tableau 19 – T_1 liste taboue

4.1.2 Itération 2

Calculons le voisinage de S_1 :

Voisins \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
$V_{2.1}$	L	Ma	L	V	Me	J	V	J	Me	Ma	7
$V_{2.2}$	Ma	L	L	V	Me	J	V	J	Me	Ma	7
$V_{2.3}$	Ma	L	V	L	Me	J	V	J	Me	Ma	8
$V_{2.4}$	Ma	L	L	Me	V	J	V	J	Me	Ma	7
$V_{2.5}$	Ma	L	L	V	J	Me	V	J	Me	Ma	8
$V_{2.6}$	Ma	L	L	V	Me	V	J	J	Me	Ma	7
$V_{2.7}$	Ma	L	L	V	Me	J	J	V	Me	Ma	6
$V_{2.8}$	Ma	L	L	V	Me	J	V	Me	J	Ma	8
$V_{2.9}$	Ma	L	L	V	Me	J	V	J	Ma	Me	8

Tableau 20 – Le voisinage de l'itération 2

La solution retenue est $V_{2.5}$:

Cours / Solution	1	2	3	4	5	6	7	8	9	10	Fitness
S_2	Ma	L	L	V	J	Me	V	J	Me	Ma	8

Tableau 21 – Solution S_2

La liste taboue est la suivante :

Cours / Solutions	1	2	3	4	5	6	7	8	9	10	Fitness
S_2	Ma	L	L	V	J	Me	V	J	Me	Ma	8
S_1	Ma	L	L	V	Me	J	V	J	Me	Ma	7
S_0	Ma	L	L	V	Me	J	J	V	Me	Ma	6

Tableau 22 – T_2 liste taboue

4.1.3 Itération 3

Calculons le voisinage de S_2.

Cours / Voisins	1	2	3	4	5	6	7	8	9	10	Fitness
$V_{3.1}$	L	Ma	L	V	J	Me	V	J	Me	Ma	8
$V_{3.2}$	Ma	L	L	V	J	Me	V	J	Me	Ma	8
$V_{3.3}$	Ma	L	V	L	J	Me	V	J	Me	Ma	9
$V_{3.4}$	Ma	L	L	J	V	Me	V	J	Me	Ma	7
$V_{3.5}$	Ma	L	L	V	Me	J	V	J	Me	Ma	7
$V_{3.6}$	Ma	L	L	V	J	V	Me	J	Me	Ma	8
$V_{3.7}$	Ma	L	L	V	J	Me	J	V	Me	Ma	7
$V_{3.8}$	Ma	L	L	V	J	Me	V	Me	J	Ma	7
$V_{3.9}$	Ma	L	L	V	J	Me	V	J	Ma	Me	8

Tableau 23 – Le voisinage de l'itération 3

La solution retenue est $V_{3.3}$:

Solution \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
S_3	Ma	L	V	L	J	Me	V	J	Me	Ma	9

Tableau 24 – Solution S_3

La liste taboue est la suivante (comme nous sommes en FIFO, S_0 part) :

Solutions \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
S_3	Ma	L	V	L	J	Me	V	J	Me	Ma	9
S_2	Ma	L	L	V	J	Me	V	J	Me	Ma	8
S_1	Ma	L	L	V	Me	J	V	J	Me	Ma	7

Tableau 25 – T_3 liste taboue

4.1.4 Itération 4

Calculons le voisinage de S_3 :

Voisins \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
$V_{4.1}$	L	Ma	V	L	J	Me	V	J	Me	Ma	9
$V_{4.2}$	Ma	V	L	L	J	Me	V	J	Me	Ma	9
$V_{4.3}$	Ma	L	L	V	J	Me	V	J	Me	Ma	8
$V_{4.4}$	Ma	L	V	J	L	Me	V	J	Me	Ma	8
$V_{4.5}$	Ma	L	V	L	Me	J	V	J	Me	Ma	8
$V_{4.6}$	Ma	L	V	L	J	V	Me	J	Me	Ma	9
$V_{4.7}$	Ma	L	V	L	J	Me	J	V	Me	Ma	8
$V_{4.8}$	Ma	L	V	L	J	Me	V	Me	J	Ma	8
$V_{4.9}$	Ma	L	V	L	J	Me	V	J	Ma	Me	9

Tableau 26 – Le voisinage de l'itération 4

La solution retenue est $V_{4.9}$:

Solution \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
S_4	Ma	L	V	L	J	Me	V	J	Ma	Me	9

Tableau 27 – Solution S_4

La liste taboue est la suivante (S_1 part S_4 rentre) :

Solutions \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
S_4	Ma	L	V	L	J	Me	V	J	Ma	Me	9
S_3	Ma	L	V	L	J	Me	V	J	Me	Ma	9
S_2	Ma	L	L	V	J	Me	V	J	Me	Ma	8

Tableau 28 – T_4 liste taboue

4.1.5 Itération 5

Calculons le voisinage de S_4 :

Voisins \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
$V_{5.1}$	L	Ma	V	L	J	Me	V	J	Ma	Me	10
$V_{5.2}$	Ma	V	L	L	J	Me	V	J	Ma	Me	9
$V_{5.3}$	Ma	L	L	V	J	Me	V	J	Ma	Me	8
$V_{5.4}$	Ma	L	V	J	L	Me	V	J	Ma	Me	8
$V_{5.5}$	Ma	L	V	L	Me	J	V	J	Ma	Me	9
$V_{5.6}$	Ma	L	V	L	J	V	Me	J	Ma	Me	9
$V_{5.7}$	Ma	L	V	L	J	Me	J	V	Ma	Me	8
$V_{5.8}$	Ma	L	V	L	J	Me	V	Ma	J	Me	8
$V_{5.9}$	Ma	L	V	L	J	Me	V	J	Me	Ma	9

Tableau 29 – Le voisinage de l'itération 5

La solution retenue est $V_{5.1}$:

Solution \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
S_5	L	Ma	V	L	J	Me	V	J	Ma	Me	10

Tableau 30 – Solution S_5

La liste taboue est la suivante (S_2 part S_5 rentre) :

Solutions \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
S_5	L	Ma	V	L	J	Me	V	J	Ma	Me	10
S_4	Ma	L	V	L	J	Me	V	J	Ma	Me	9
S_3	Ma	L	V	L	J	Me	V	J	Me	Ma	9

Tableau 31 – T_5 liste taboue

4.1.6 Bilan

En 5 itérations, nous arrivons à la valeur maximale de la fitness.

Voici le planning satisfaisant toutes les classes :

Solution \ Cours	1	2	3	4	5	6	7	8	9	10	Fitness
S_5	L	Ma	V	L	J	Me	V	J	Ma	Me	10

Tableau 32 – Solution S_5 finale

Nous pouvons visualiser l'évolution de la fitness, en quelques itérations, nous arrivons sur une très bonne solution.

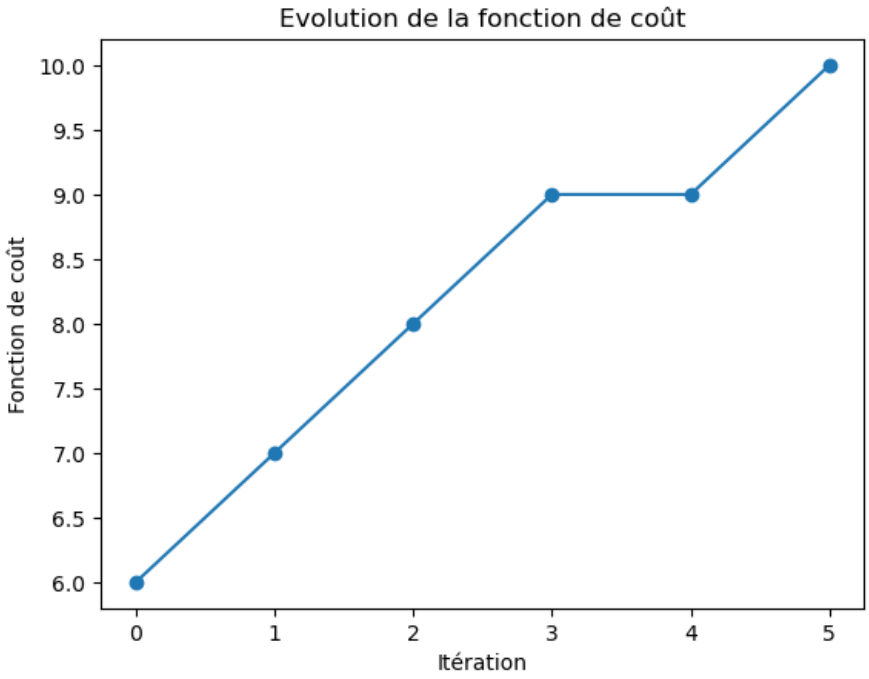


Image 4 – Évolution de la fitness

5 Recuit simulé

5.1 Paramètres du problème

Pour appliquer la méthode du recuit simulé, il nous faut premièrement en définir les paramètres.

Tout d'abord, la solution initiale est obtenue en générant aléatoirement une répartition des cours, dans les limites des contraintes fortes :

Cours / Solution	1	2	3	4	5	6	7	8	9	10
S_0	J	Me	Ma	V	V	J	L	V	V	Me

Tableau 33 – S_0 recuit simulé

Jours	L	Ma	Me	J	V
S_0	7	3	2, 10	1,6	4, 5, 8, 9

Tableau 34 – S_0 initiale (jours)

Nous pouvons maintenant étudier la fitness de cette solution :

— Classe 1 : 1 conflit, fitness égale à 1
— Classe 2 : 2 conflits, fitness égale à 0
— Classe 3 : 1 conflit, fitness égale à 1
— Classe 4 : 1 conflit, fitness égale à 1
— Classe 5 : 1 conflit, fitness égale à 1

Pour une fitness totale de 4.

Pour des raisons pratiques, on limitera le recuit simulé à 5 itérations par palier, à un facteur de refroidissement $\lambda = 0,5$, à une température initiale $T_0 = 2$, et à un seuil de $0,3$, afin de n'effectuer que 15 opérations.

Nous définissons une solution voisine comme toute solution respectant les contraintes fortes obtenues en déplaçant un cours à un jour adjacent.

On choisit la transformation à effectuer en choisissant un entier c non nul dans l'intervalle $[-10; 10]$.

Un nombre négatif correspond à un déplacement du cours numéroté $|c|$ au jour précédent (nous considérons que le jour précédant le lundi est le vendredi), et inversement pour un nombre positif.

5.2 Application

On commence l'algorithme avec $T = 2$

On tire au hasard et on obtient $c = -1$

On recule donc le cours numéro 1 au jour précédent, qui est le Mercredi.

On obtient donc la solution suivante :

Jours	L	Ma	Me	J	V
S_0	7	3	1, 2, 10	6	4, 5, 8, 9

Tableau 35 – S_1 recuit simulé

Cette solution ayant une fitness de 5, le delta de fitness est de $\Delta f = 1$, et la nouvelle solution est immédiatement adoptée.

Itération suivante :

$c = 2$

Nouvelle solution :

Jours	L	Ma	Me	J	V
S_0	7	3	1, 10	2, 6	4, 5, 8, 9

Tableau 36 – S_2 recuit simulé

$\Delta f = 1$

Adoption immédiate

Itération suivante :

$c = 9$

Nouvelle solution :

Jours	L	Ma	Me	J	V
S_0	7, 9	3	1, 10	2, 6	4, 5, 8

Tableau 37 – S_3 recuit simulé

$\Delta f = 1$

Adoption immédiate

Itération suivante :

$c = -1$

Jours	L	Ma	Me	J	V
S_0	7, 9	1, 3	10	2, 6	4, 5, 8

Tableau 38 – S_4 recuit simulé

Nouvelle solution :

$\Delta f = -1$

Probabilité d'acceptation : $p = e^{\frac{-\Delta_f}{T}} = 0.6065306597126334$

Tirage d'un nombre x entre 0 et 1 : $x = 0.280299954665289$

$x < p$

Solution acceptée

Itération suivante :

$c = -6$

Nouvelle solution :

Jours	L	Ma	Me	J	V
S_0	7 ,9	1 ,3	6, 10	2	4, 5, 8

Tableau 39 – S_5 recuit simulé

$\Delta f = 0$

Adoption immédiate

Après un cycle, on constate que la fitness est passée de 4 à 6.

On réduit la température d'un facteur de 0.5, et on obtient $T_1 = 1$, puis on recommmence un cycle, à la fin duquel on atteint la solution suivante :

Jours	L	Ma	Me	J	V
S_0	8, 9	3, 7, 3	1, 6	2, 5	4

Tableau 40 – S_{10} recuit simulé

Pour une fitness de 7.

Arpès un dernier cycle à une température de $T_2 = 0.5$, on finit sur la solution suivante :

Jours	L	Ma	Me	J	V
S_0	8, 9	3, 7, 3	6	1, 2, 5	4

TABLEAU 41 – S_{15} recuit simulé

Avec une fitness de 8. La température suivante, $T_3 = 0.25$, étant inférieure au seuil, l'algorithme s'arrête ici.

L'algorithme est donc efficace. D'autres tests nous ont permis de constater qu'abaisser le seuil à 0.2, amenant le nombre de cycles à 4, suffit quasi-systématiquement à atteindre la fitness maximale de 10.

6 Colonies de fournies

6.1 Paramètres du problème

La méthode de la colonie de fourmis étant spécialisée dans la résolution de problèmes analogues à celui du marchand de commerce, nous allons tout d'abord devoir la redéfinir pour être compatible avec notre sujet.

Il faut d'abord définir ce que représenteront les nœuds et les chemins du problème.

Une première approche serait de considérer chaque nœud comme une configuration possible de l'emploi du temps, et chaque chemin comme le passage d'une configuration à une autre.

Cependant, cette approche s'avère rapidement irréalisable. Chaque cours peut se trouver dans 6 configurations : assigné à un des 5 jours de la semaine, ou non-assigné.

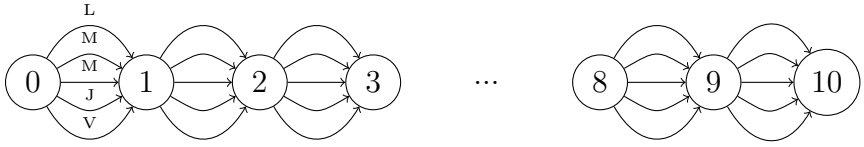
Avec 10 cours, on atteint $6^{10} = 6E7$ nœuds, ce qui, au delà des problèmes de mémoire, demanderait aux fourmis un nombre d'itérations astronomique pour faire le moindre progrès.

Il s'avère en réalité que le problème de l'affectation d'emploi du temps est assez peu compatible avec la méthode de la colonie de fourmis. Il est néanmoins possible d'en utiliser une version modifiée.

Pour limiter le nombre de nœuds, on considérera 11 nœuds, le premier représentant l'emploi du temps initial vide, et chaque nœud suivant représentant un des 10 cours à affecter.

Chaque paire de nœuds consécutifs sera séparée par 5 chemins, représentant les 5 jours de la semaine auxquels peut être affecté le prochain cours.

Les fourmis parcourront donc tous les nœuds dans le même ordre, mais en empruntant des chemins différents entre chaque nœud, comme représenté sur ce graphe :



Chaque chemin, de haut en bas, représente un jour de lundi à vendredi.

Au lieu de pondérer le choix du prochain nœud par sa distance, les fourmis calculeront le nombre de conflits engendrés par le choix de chaque chemin.

Ainsi, la probabilité qu'un chemin c soit choisi pour passer du nœud $n-1$ au nœud n est décrite par la formule suivante :

$$P_{n,c} = \frac{\tau_{n,c}^{\alpha} \eta_{n,c}^{\beta}}{\sum_{d=0}^{5} \tau_{n,d}^{\alpha} \eta_{n,d}^{\beta}}$$

Avec α et β deux réels positifs, $\tau_{n,c}$ la trace de phéromones présente sur le chemin c entre les nœuds $n-1$ et n, et $\eta_{n,c} = \frac{1}{1+conflits_{n,c}}$, avec $conflits_{n,c}$ le nombre de classes devant assister à la fois au cours n, et à au moins un autre cours déjà assigné au jour c.

A la fin de leur parcours, les fourmis déposeront sur leur chemin une trace proportionnelle au score de fitness de l'emploi du temps obtenu, et on met à jour la trace via la formule :

$$\tau_{n,c} = (1-\rho)\tau_{n,c} + Q \sum_{k \in K} \Delta_{n,c}^{k}$$

Avec K l'ensemble des fourmis, $\rho \in]0;1[$ le facteur d'évaporation, Q une constante, et $\Delta_{n,c}^{k}$ égal à la fitness de la fourmi k si elle a emprunté le chemin (n,c), et 0 sinon.

Une fois le nombre d'itérations voulues passer, on retourne la fourmi de la population finale ayant la meilleure fitness.

6.2 Application numérique

On initialise l'algorithme avec les constantes suivantes :

— $\alpha = 0.5$
— $\beta = 0.5$
— $Q = 1$
— $\rho = 0.7$
— $\tau_0 = 0.001$
— Population de fourmis : 10
— Nombre d'iterations : 3

La première fourmi se trouve au nœud 0. Les probabilités de choix pour le prochain chemin sont :

Chemin	1	2	3	4	5
Probabilité	0.2	0.2	0.2	0.2	0.2

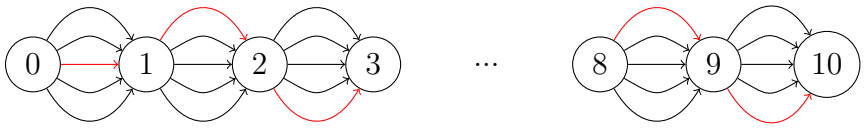
Tableau 42 – Probabilités de choix de la fourmi 1 pour le nœud 1 à la génération 1

Elle choisit le nœud 3. Les probabilités de choix pour le nœud 2 sont ainsi :

Chemin	1	2	3	4	5
Probabilité	0.22	0.22	0.12	0.22	0.22

Tableau 43 – Probabilités de choix de la fourmi 1 pour le nœud 2 à la génération 1

La fourmi choisit le nœud 1. On continue jusqu'au nœud 10, et on obtient le chemin suivant :



Ce qui correspond à la répartition suivante :

Jour	1	2	3	4	5	Fitness
Cours	2, 4, 9		1, 5, 6, 8		3, 7, 10	6

Tableau 44 – Répartition de l'emploi du temps de la fourmi 1 à la génération 1

Après avoir répété l'opération pour les 10 fourmis, on obtient comme élite la fourmi 6 :

Jour	1	2	3	4	5	Fitness
Cours	2	1, 5, 7, 9, 10	3		4, 6, 8	8

Tableau 45 – Répartition de l'emploi du temps de la fourmi 6 à la génération 1

La trace de phéromones est la suivante :

Noeud \ Chemin	1	2	3	4	5
1	1.0e-03	2.9e+01	1.2e+01	1.5e+01	1.4e+01
2	2.3e+01	1.0e+01	1.7e+01	1.4e+01	6.0e+00
3	1.0e-03	1.9e+01	8.0e+00	8.0e+00	3.5e+01
4	1.0e+01	2.2e+01	1.5e+01	7.0e+00	1.6e+01
5	9.0e+00	2.0e+01	1.8e+01	1.7e+01	6.0e+00
6	7.0e+00	1.8e+01	1.6e+01	1.0e-03	2.9e+01
7	1.2e+01	1.8e+01	7.0e+00	2.3e+01	1.0e+01
8	6.0e+00	1.2e+01	2.8e+01	7.0e+00	1.7e+01
9	2.0e+01	3.6e+01	1.0e-03	1.0e-03	1.4e+01
10	1.0e+01	3.9e+01	1.0e-03	1.0e-03	2.1e+01

Tableau 46 – Trace laissée par la génération 1

On passe maintenant à la deuxième génération de fourmis :

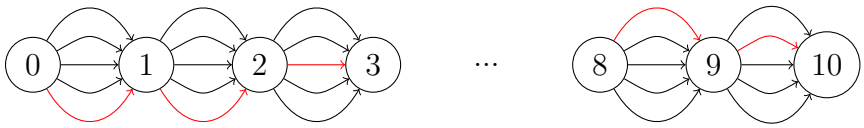
La première fourmi se trouve au nœud 0. Les probabilités de choix pour le prochain chemin sont :

Chemin	1	2	3	4	5
Probabilité	0.001	0.326	0.210	0.235	0.228

Tableau 47 – Probabilités de choix de la fourmi 1 pour le nœud 1 à la génération 2

Elle choisit le nœud 4.

On continue ainsi jusqu'à la dernière génération, et on obtient cette élite pour la génération 3 :



Qui donne donc cette répartition :

Jour	1	2	3	4	5	Fitness
Cours	8, 9	5, 10	4, 6, 7	3	1, 2	10

TABLEAU 48 – Répartition de l'emploi du temps de la meilleure fourmi à la génération 2

On a réussi à atteindre une solution pour laquelle la fitness est maximale.

6.3 Retour critique

Comme vu plus haut, la méthode de la colonie de fourmis ne se prête pas particulièrement à la résolution de ce genre de problème. Bien que nous finissions par trouver une solution dont la fitness est maximale, il s'agit vraisemblablement plus de hasard que d'une réelle efficacité de la méthode.

Le principal obstacle à la conception d'un algorithme de colonie de fourmis efficace pour résoudre ce problème est due à la nature de problème d'affectation de notre problème.

Étant donné qu'aucun « chemin » de l'arbre n'est intrinsèquement meilleur qu'un autre, la qualité de chaque chemin dépendant des chemins empruntés précédemment, les phéromones se révèlent peu utiles dans la découverte d'une solution optimale.

7 Essaim particulaire

La résolution de notre problème grâce à un essaim particulaire n'est pas une chose aisée. En effet, cet algorithme ne semble pas très bien se prêter aux problèmes d'optimisation d'emplois du temps, à cause de l'encodage d'une solution. Dans un algorithme PSO, le calcul pour mettre à jour la vélocité des particules fait appel aux solutions elles-mêmes, mais il n'est, en l'état, pas possible de calculer la différence entre une solution et une autre, en faisant en sorte que le calcul ait un sens au niveau de l'algorithme. On pourrait bien entendu traduire tous les jours en chiffres, mais cela n'arrangerait pas le problème du fait que la descente en profondeur aurait extrêmement de mal à se faire, car cette différence n'a pas réellement de sens.

$$v_i(t+1) = \omega v_i(t) + c_1 r_1[pbest_i(t) - x_i(t)] + c_2 r_2[g_best(t) - x_i(t)]$$

Une solution qui nous est venue à l'esprit pour palier à ce problème est de ne pas travailler sur 1 solution = 1 particule, mais plutôt sur 1 UV = 1 particule. On aurait alors un tableau d'UVs avec leur jour correspondant sur lequel il serait plus simple de travailler. Malheureusement, il se dresse à nous un nouveau problème, celui du calcul de la fitness.

Chaque UV représentant une particule, il faut mettre à jour leur vélocité individuellement, ce qui demande à recalculer la fitness pour chacune d'entre elles. Malheureusement, pour certaines classes, un placement d'UV peut tout à fait être compatible avec leurs autres vœux, mais dans une autre, il peut y avoir un conflit.

UV	1	2	3	4	5	6	7	8	9	10
Jour	Ma	L	L	V	Me	J	J	V	Me	L

Tableau 49 – Explosion de notre solution initiale en traitant une UV comme une particule

Pour les vœux des étudiants suivants :

Classes	1	2	3	4	5
UVs	1, 2, 3	4, 5, 9	3, 4, 8	2, 4, 5	2, 3, 10

Tableau 50 – Voeux des étudiants

Il y aurait par exemple le problème suivant avec les calculs de fitness : Pour la classe 1, la fitness de l'UV 2 serait à 1, car il y a 1 conflit, mais pour la classe 4, elle serait à 2, car il n'y a pas de conflits.

Cela signifie donc que la fitness ne peut pas être calculée sur chaque particule, rendant impossible la résolution du problème avec cette méthode.

Ensuite, nous avons pensé qu'il pourrait être plus intéressant de travailler avec 1 classe = 1 particule, puisque cela règlerait le problème de calcul de fitness. Cela créerait néanmoins une nouvelle difficulté à surmonter, celle de devoir travailler avec des matrices, au lieu d'un simple chiffre réel, comme dans la méthode précédente. Aussi, un doute persiste sur le fait que la possibilité pour mettre à jour la vélocité ait un réel sens.

Un papier (Particle swarm optimization algorithm applied to scheduling problems[2]) sur l'optimisation d'emplois du temps grâce à un algorithme PSO semble réussir à régler ce problème en encodant chaque solution, afin de pouvoir effectuer des opérations sur celles-ci, tout en s'assurant qu'elles gardent leur sens. Cela étant dit, les chercheurs ayant écrit cet article ne vont pas dans les détails du calcul de l'encodage et du décodage, et, plutôt que d'utiliser la version classique de l'algorithme, ils utilisent des variantes qui ne paraissent pas applicable à notre problème.

8 Conclusion

En analysant les performances de ces cinq métaheuristiques, il est notable que les algorithmes à population (comme l'algorithme génétique), la recherche taboue et le recuit simulé sont particulièrement efficaces pour ce type de problème. Cette efficacité s'explique en grande partie par la facilité à modéliser le problème de façon similaire aux emplois du temps courants, mais aussi une exploration efficace de l'espace des solutions et d'obtenir rapidement de bons résultats.

Concernant l'approche par colonie de fourmies, bien que nous ayons adapté le problème des emplois du temps sous forme de graphe pour le rapprocher au problème du voyageur de commerce (où cet algorithme excelle), les améliorations apportées à la solution restent limitées, ne faisant pas de cette méthode la plus efficace pour notre cas. Quant à l'essaim particulaire, nous n'avons pas réussi à trouver une modélisation adaptée à ce problème.

En résumé, pour la résolution de problèmes d'emploi du temps, les méthodes les plus efficaces s'avèrent être l'algorithme génétique, la recherche taboue et le recuit simulé. Pour améliorer encore cette solution, nous pouvons créer avec l'algorithme génétique une très bonne solution et l'améliorer avec une recherche taboue ou un recuit simulé.

Annexes

Glossaire

UTBM University of Technologie of Belfort-Montbéliard. 5, 6

Table des figures

Liste des tableaux

Références

[1] Edmund K Burke, David Elliman, and Rupert Weare. A genetic algorithm based university timetabling system. In *Proceedings of the 2nd east-west international conference on computer technologies in education*, volume 1, pages 35–40. Citeseer, 1994.

[2] Pisut Pongchairerks. Particle swarm optimization algorithm applied to scheduling problems, 2009.

[3] Chausson Thibault, El Haj Hissein Olivier, Guinot Jossua, Metallaoui Nassim, and Viguier Léo. *AI50 - Time table scheduling*. GitHub, https://github.com/thibault-chausson/AI50-TimeTableScheduling consulté le 13 janvier 2024, le 7 janvier 2024.