République Algérienne Démocratique et Populaire

Ministère de L'Enseignement Supérieur et de

La Recherche Scientifique

Université 20 Aout 1955-Skikda

Faculté des Sciences -Département d'informatique

Mémoire de fin d'études en vue de l'obtention du diplôme de

Master

Option : *Système Informatique (SI)*

Thème : Une approche basée Clustering pour le problème d'emploi du temps des cours universitaires.

Réalisé par :

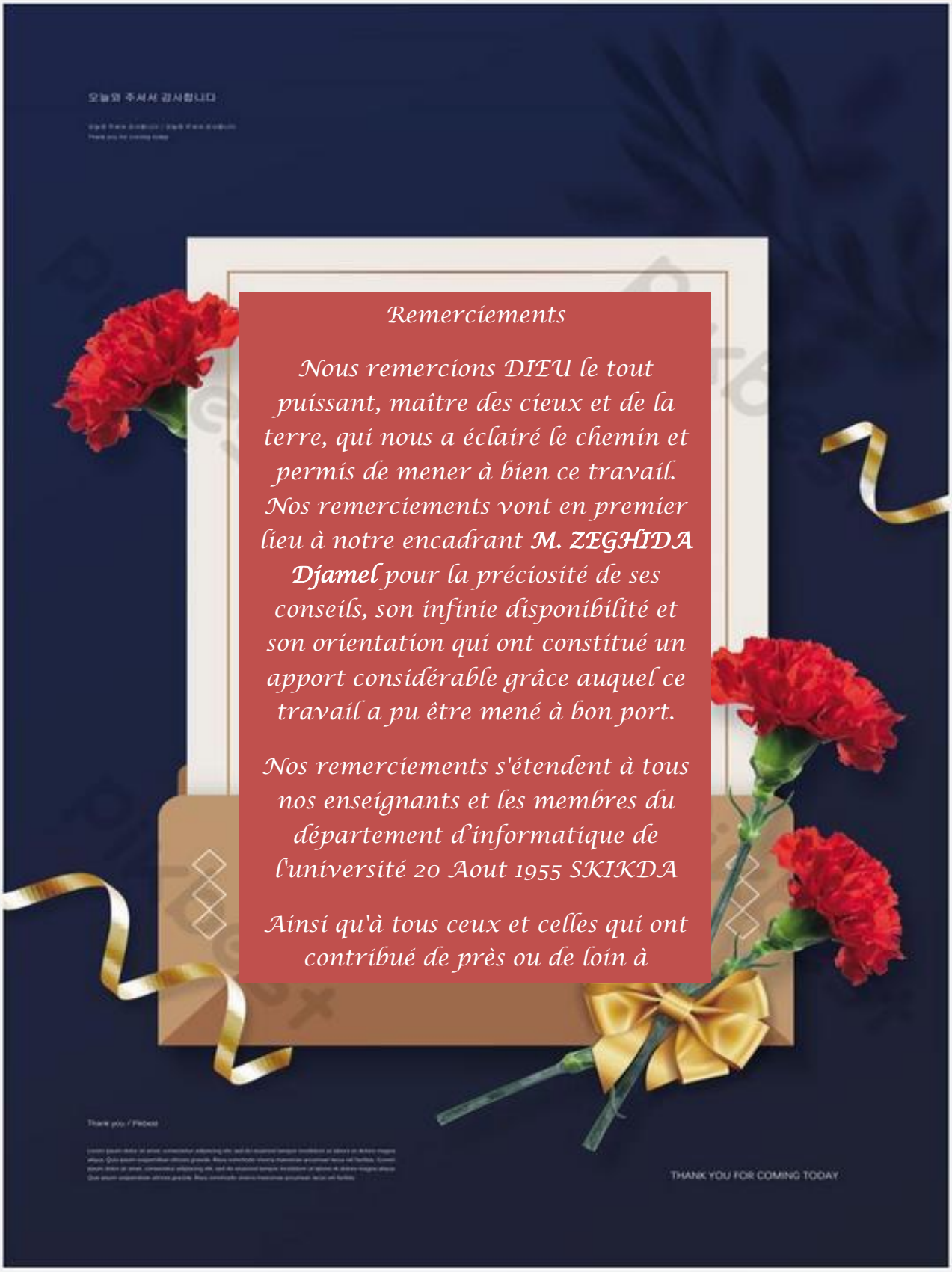
BOUTELALA Meissoune

BOUNOUARA Amina

Encadré par :

Dr Zeghida Djamel

Session : juin 2022



Remerciements

Nous remercions DIEU le tout puissant, maître des cieux et de la terre, qui nous a éclairé le chemin et permis de mener à bien ce travail. Nos remerciements vont en premier lieu à notre encadrant ***M. ZEGHIDA Djamel*** *pour la préciosité de ses conseils, son infinie disponibilité et son orientation qui ont constitué un apport considérable grâce auquel ce travail a pu être mené à bon port.*

Nos remerciements s'étendent à tous nos enseignants et les membres du département d'informatique de l'université 20 Aout 1955 SKIKDA

Ainsi qu'à tous ceux et celles qui ont contribué de près ou de loin à

Dédicace

Je dédie ce modeste travail :

A la mémoire de mon père BOUTELALA Djamel, j'espère que du monde qui est sien maintenant, il apprécie cet humble geste comme preuve de reconnaissance de la part d'une fille qui a toujours prié le salut de son âme.

A ma mère Anouche Hayet qui m'a arrosé de tendresse et d'espoir qui m'a dirigé vers la gloire.

A mes grands-parents qui m'ont toujours encouragé et soutenu au long de mon parcours universitaire.

A mes très chère frères Wail, Amine et sœurs Amira, ABir.

A tous les membre de ma famille et très chers amis.

A TOUS QUI M'AIMENT.

BOUTELALA MEISSOUNE

Dédicace

Un grand merci à l'ensemble de ma famille,

Mes chers amis pour leur amour, leur confiance,

Leur conseil tout au long de mes études.

Amina.

Résumé

Le problème d'emploi du temps peut être vu comme une instance des problèmes d'ordonnancement des tâches, et il est considéré comme un problème incontournable dans tous les domaines.

Nous parlerons des problèmes d'emploi du temps des cours universitaires, ces problèmes se produisent dans toutes les universités et sont renforcés chaque année par les praticiens.

La construction d'emploi du temps des cours universitaires consiste à organiser des rencontres (des séances de cours, de TD et de TP) entre les enseignants et les étudiants dans des salles de différentes catégories (Amphithéâtres, salles de TD et salle de TP) et durant des périodes hebdomadairement fixes.

Dans ce travail, nous nous intéressons à la résolution automatique du problème d'emploi du temps universitaire. Pour ce fait, nous proposons une approche basée clustering pour le problème d'emploi du temps des cours universitaires.

Mot clés : Problème d'emploi du temps des cours universitaires, Clustering.

Abstract

The timetabling problem can be seen as an instance of task scheduling problems, and it is seen as an unavoidable problem in all domains.

We will talk about university course timetabling problem, these problems occur in all universities and are reinforced every year by practitioners.

The construction of a timetable for university courses consists of organizing meetings (lessons, TD and TP sessions) between teachers and students in rooms of different categories (Amphitheatres, TD rooms and TP room) and during fixed weekly periods.

In this work, we are interested in the automatic resolution of the university timetabling problem. For this, we propose a clustering-based approach to the university course timetabling problem.

Keywords: University course timetabling problem, Clustering.

ملخص

يمكن النظر إلى مشكلة الجدول الزمني على أنها مثال على قضايا جدولة المهام، ويُنظر إليها على أنها عقبة في جميع المجالات.

سنتحدث عن مشكلة الجدول الزمني للدورات الجامعية، هذه المشاكل تحدث في جميع الجامعات ويتم تعزيزها كل عام من قبل الممارسين.

يتكون إنشاء جدول زمني للدورات الجامعية من تنظيم اجتماعات (دروس، وجلسات TD وTP) بين المعلمين والطلاب في غرف من فئات مختلفة (المدرجات، وغرف TD، وغرفة TP) وخلال فترات أسبوعية محددة.

في هذا العمل، نحن مهتمون بالحل الآلي لمشكلة الجدول الزمني للجامعة. لهذا، نقترح نهجًا قائمًا على التجميع لمشكلة الجدول الزمني للدروس الجامعية.

الكلمات المفتاحية: مشكلة الجدول الزمني للدروس الجامعية ، التجميع.

.

Table de Matières :

Liste des Figures :

Figure 24 : Diagramme de classe.

Figure 25: interface de l'application.

Introduction générale

La création de groupes au sein d'un ensemble d'éléments est une opération omniprésente dans notre société. Les groupes peuvent posséder plusieurs significations, comme par exemple une signification sociale quand ils d'écrivent un ensemble de personnes qui partagent des caractéristiques communes. Il est naturel de faire appel aux groupes quand il s'agit de structurer, d'organiser ou de résumer un ensemble d'éléments. Ainsi, dans la vie quotidienne, la notion de groupe est utilisée pour l'organisation et la description, comme par exemple des familles de plantes, des genres de musique, des groupes de produits, des groupes sanguins, etc.

Un groupe peut être d'défini de manière informelle comme un ensemble d'éléments qui sont rassemblés en raison d'une relation particulière entre ces éléments. La problématique consistante à former de tels groupes de manière automatique se pose dans de nombreux domaines. En marketing, on va chercher à regrouper des personnes ayant des comportements de consommation similaires pour cibler par exemple une campagne publicitaire.

Dans l'étude des réseaux sociaux, on cherche à regrouper les différents membres du réseau pour y faire émerger des communautés. En biologie, on cherche à identifier des groupes de gènes ayant le même comportement pour mettre en place des thérapies géniques.

L'objectif principal de l'informatique est d'automatiser la résolution des problèmes du monde réel d'une façon fiable et efficiente. Comme exemple de ces problèmes, le problème d'emploi du temps, et plus précisément, le problème d'emploi du temps de cours universitaires.

Ce dernier se focalise autour de l'affectation des enseignants, des groupes d'étudiants et des salles a des créneaux horaires sans engendrer des conflits : des séances qui se déroulent au même temps comprenant des enseignants et / ou des étudiants communs, ou exploitant les mêmes salles.

Dans le cadre du projet de fin d'études, notre travail vise à traiter le problème d'emploi du temps des cours universitaires comme un problème de clustering, en faisons passer les séances de cours pour des objets, au sens du clustering et leurs intervenants : enseignants, étudiants, salles, créneaux horaires etc., seront considérés comme attributs d'objet.

Outre l'introduction générale et la conclusion générale, ce mémoire sera organisé en cinq chapitres comme suit :

- ✓ Le premier chapitre sera consacré à une présentation de l'état de l'art du clustering de données.

- ✓ Dans le deuxième chapitre nous allons présenter le problème d'emploi du temps en commence par la définition, formulation, les méthodes de résolution et ainsi que les problèmes.
- ✓ Le troisième chapitre nous allons présenter Les métaheuristique, les méthodes de recherche locale, les solutions et ces algorithmes.
- ✓ Le quatrième chapitre est dédié à la présentation de la conception et la modélisation de l'approche basé clustering qui est optimiser par les loups gris et le coloriage.
- ✓ Le cinquième chapitre sera consacré à l'implémentation et les résultats de nos variantes loups gris dans une application bureau avec des captures détaillées de l'application.

1

Le Clustering

1. Introduction :

L'analyse de cluster est un nom général pour un grand ensemble de statistiques des méthodes qui visent toutes à détecter des grappes dans un échantillon objets, ces groupes sont généralement appelés clusters [43].

Le Clustering a été créé en anthropologie par *Driver* et *Kroeber* en 1932 [39] et introduite en psychologie par *Joseph Zubin* en 1938 [40] et *Robert Tryon* en 1939 [41] et utilisée par *Cattell* à partir de 1943 [42] pour la classification de la théorie des traits en psychologie de la personnalité.

En raison de l'ambiguïté concernant les problèmes de clustering, différentes approches ont étés proposées dans la littérature afin d'améliorer son utilisation sur des applications spécifiques. Ces techniques se diffèrent dans leurs principes, propriétés, paramètres et formes générales du partitionnement généré.

Dans ce chapitre, nous présentons le clustering et ses différentes méthodes. Ensuite, nous allons citer les principales approches de clustering, leurs caractéristiques et les techniques de validation d'un algorithme de clustering. En dernier nous parlerons de ses domaines d'applications.

2. Définition de Clustering :

Le Clustering est la division (regroupement ou partitionnement) des données non étiquetées en un certain nombre de groupes de sorte que les données des mêmes groupes sont plus semblables ou proches et que les données des autres groupes sont moins semblables et distantes. Il s'agit essentiellement d'une méthode d'apprentissage non supervisée.

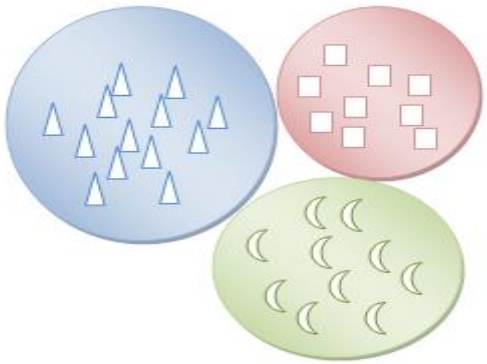


Figure 1 : Le clustering.

Dans le problème de clustering, l'ensemble des données est composé de ***m*** objets (observations) sans étiquette (ou classes prédéfinies), chacun est décrit par plusieurs variables. On note :

$X = \{x_1, x_2, x_3, \ldots, x_n\}$, l'ensemble des ***n*** variables décrivant l'ensembles des ***m*** objets [1]. Les données peuvent donc être représentées par une matrice de taille (***m*******n***) avec ***m*** lignes, chacune représente un objet, et ***n*** colonne chacune représente un attribut.

L'idée est donc de découvrir des groupes au sein des données de telle sorte que les données du même cluster aient des caractéristiques similaires à l'intérieur de chaque cluster, les données sont regroupées selon une caractéristique commune.

L'outil de classification est un algorithme qui mesure la proximité entre chaque élément à partir de critères définis. En clair, le clustering cherche à faire des classes telle que [17][18] :

— Les différences intra-classe soient minimales pour obtenir des clusters.

— Les différences inter-classe soient maximales afin d'obtenir des sous-ensembles bien différenciés.

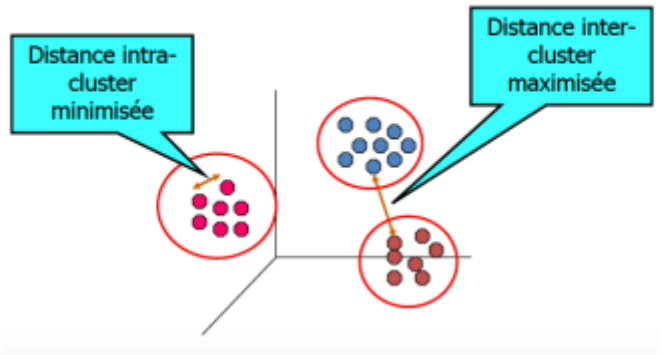


Figure 2 : Distance intra-classe et inter-classe [17].

Le but des algorithmes de clustering est de donner un sens aux données et d'extraire de l'information à partir de grandes quantités de données structurées et non structurées. Il s'agit d'une technique d'analyse statistique des données très utilisée dans de nombreux domaines, y compris l'apprentissage automatique, la reconnaissance de formes, le traitement de signal et d'images, la recherche d'information, la bio-informatique, la compression de données, l'infographie, etc. [19].

3. Les principales étapes des algorithmes de clustering :

Généralement les méthodes de clustering doivent inclure les étapes suivantes [15], [16] :

- Choix de données (optionnellement inclut l'extraction et/ou la sélection des caractéristiques).
- Définition d'un modèle (une fonction) de calcul de proximité appropriée aux données.
- Classification ou regroupement.
- Abstraction des données (s'il est nécessaire).
- Evaluation des résultats (s'il est nécessaire).

4. Mesure de distance :

Les principales mesures de distance utilisées en clustering sont :

4.1 Distance de Minkowski :

La distance de *Minkowski* est une échelle métrique utilisée lorsque les attributs des objets sont de type numérique.

Elle est définie par des données de deux éléments de dimension p,

$x_i = (x_{i^1}, \dots, x_{iP})$ et $x_j = (x_{j^1}, \dots, x_{j_P})$, la distance entre les deux éléments peut être calculée par la métrique de *Minkowski* :

$$d(x_i, x_j) = |x_{i1} - x_{j1}|^g + |x_{i2} - x_{j2}|^g + \ldots + |x_{ip} - x_{jp}|^g$$

Pour $g = 2$, on obtient la distance *Euclidienne*.

Pour $g = 1$, la mesure est appelée la distance de *Manhattan*.

Pour $g = \infty$, c'est la métrique de *Tchebychev*.

4.2 Mesures pour les données binaires :

Dans le cas des attributs binaires, la distance entre les objets peut être calculée sur la base de la table de contingence. Un attribut binaire est symétrique si ses deux états ont le même poids. Dans ce cas, on utilise le coefficient d'appariement simple pour évaluer la dissimilitude entre deux objets [5] :

$$d(x_i, x_j) = \frac{r + s}{q + r + s + t}$$

Où ***q*** est le nombre d'attributs qui sont égaux à 1 pour les deux objets, ***t*** est le nombre d'attributs qui sont égaux pour les deux objets, ***s*** et ***r*** sont les nombres d'attributs qui sont inégaux pour les deux objets.

Un attribut binaire est dit asymétrique si une valeur est moins significative que l'autre, dans ce cas la dissimilarité est calculée en utilisant le coefficient de *Jaccard* [5] :

$$d(x_i, x_j) = \frac{r + s}{q + r + t}$$

4.3 Mesures pour les données nominales :

Lorsque les attributs sont nominaux, deux approches peuvent être utilisées [5] :

1. L'appariement simple (Simple Matching) :

$$d(x_i, x_j) = \frac{p - m}{p}$$

Où $\boldsymbol{p}$ est le nombre total d'attributs, et $\boldsymbol{m}$ le nombre de correspondance entre les objets (égalité en valeur des attributs).

2. Crée un attribut binaire pour chaque état de chaque attribut nominal et calculer la dissimilarité [6].

4.4 Mesures pour les données ordinales :

Lorsque les attributs sont ordinaux, l'ordre des valeurs est significatif, dans ces cas les attributs peuvent être traités en tant qu'attributs numériques après leur remplacement par leur rang respectif sur l'intervalle [0.1], ce remplacement est effectué comme suit [5] :

$$z_{i,n} = \frac{r_{i,n} - 1}{M_n - 1}$$

Où $\boldsymbol{z_{i,n}}$ est la valeur standardisée de l'attribut $\boldsymbol{a_n}$ de l'objet i.

$\boldsymbol{r_{i,n}}$ est la derrière valeur avant standardisation et $\boldsymbol{M_n}$ la limite supérieure du domaine de l'attribut $\boldsymbol{a_n}$ (la limite inferieur est supposée égale à 1).

5. Fonction de similarité :

La fonction de similarité $\boldsymbol{s(x_i, x_j)}$ qui compare deux vecteurs $\boldsymbol{x_i}$ et $\boldsymbol{x_j}$ constitue une alternative aux mesures de distance [7].

Cette fonction doit être symétrique (i.e $s(x_i, x_j) = (x_i, x_j)$), avoir une valeur élevée lorsque $\boldsymbol{x_i}$ et $\boldsymbol{x_j}$ sont similaires, et atteindre maximum lorsque les vecteurs sont identiques[6].

Une fonction de similarité où l'intervalle cible est [0.1] est appelée fonction dichotomique de similarité.

5.1 Mesure du cosinus :

Lorsque l'angle entre deux vecteurs est une mesure significative de leur similarité, alors le produit intérieur normalisé peut être une mesure de similarité appropriée [8] :

$$s(x_i, x_j) = \frac{x_i^T x_j}{\| x_i \| \| x_j \|}$$

5.2 Mesure de corrélation de Pearson :

Elle est définie par [8] :

$$s(x_i, x_j) = \frac{(x_i - \bar{x}_i)^T (x_j - \bar{x}_j)}{\| (x_i - \bar{x}_i) \| \| (x_j - \bar{x}_j) \|}$$

Où $\bar{x}_i$ représente la moyenne de x sur toutes les dimensions.

5.3 Mesure de Jaccard étendue :

La mesure de *Jaccard* étendue a été présentée par *Strehl* et *Ghosn* en 2000 [8] et elle est définie par :

$$s(x_i, x_j) = \frac{x_i^T x_j}{\| x_i \|^2 \| (x_j \|^2 - x_i^T x_j}$$

5.4 Coefficient de mesure matriciel :

Il est similaire à la mesure de *Jaccard* étendue est défini par [8] :

$$s(x_i, x_j) = \frac{x_i^T x_j}{\| x_i \|^2 \| x_j \|^2}$$

6. Approches de clustering :

Le processus de clustering peut être formalisé de trois façons différentes [2] :

6.1 Clustering dur (*Hard* ou *crisp clustering*):

C'est le regroupement des données de sort que chaque élément de donnée appartient à un seul cluster.

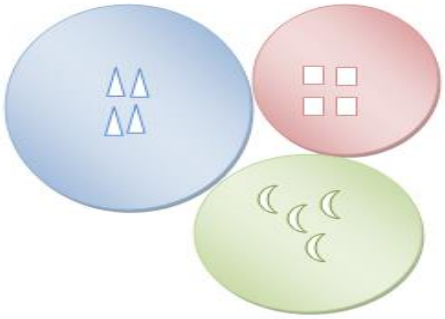


Figure 3: Le clustering dur.

6.2 Clustering flou (*Fuzzy clustering*):

C'est le regroupement des données de sort que chaque élément de donnée peut appartenir à plusieurs clusters.

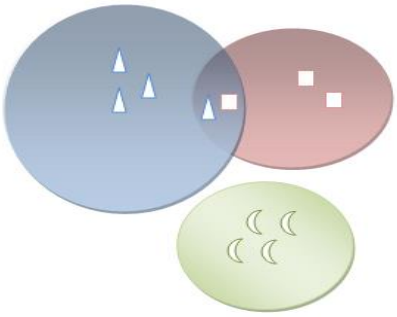


Figure 4: Le clustering flou.

6.3 clustering doux (*Soft clustering*):

C'est le regroupement des données de sort que chaque élément de données appartient à plusieurs groupes à la fois.

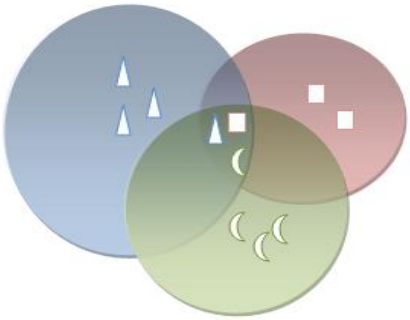


Figure 5 : Le clustering doux.

7. Les Méthodes de clustering :

Les méthodes de clustering sont souvent divisées en ces catégories :

7.1 Le clustering par partitionnement :

C'est la méthode qui effectue un partitionnement à un niveau choisi sur les données. Le partitionnement est basé sur la division des données en groupes, ces groupes sont appelés clusters, de sorte que chaque cluster comprend au moins un objet.

Le principe est alors de comparer plusieurs schémas de clustering (plusieurs partitionnements) afin de retenir le schéma qui optimise un critère de qualité [2].

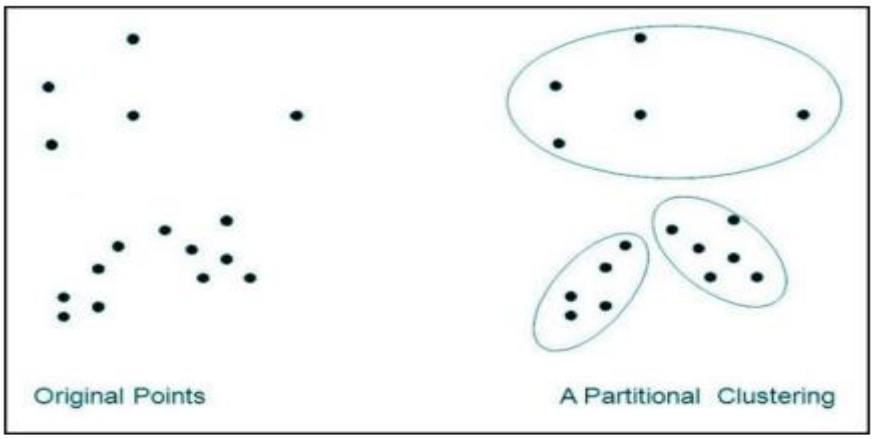


Figure 6 : Présentation de l'algorithme de clustering de partitionnement [11].

7.1.1 L'algorithme des k-moyennes (k-means) :

Le terme "*k-mean*" a été utilisé pour la première fois par *James MacQueen* en 1967 [9]. C'est l'algorithme de clustering le plus connu et le plus utilisé dans divers domaines d'application scientifique et industrielle, il permet de regrouper des objets non étiquetés.

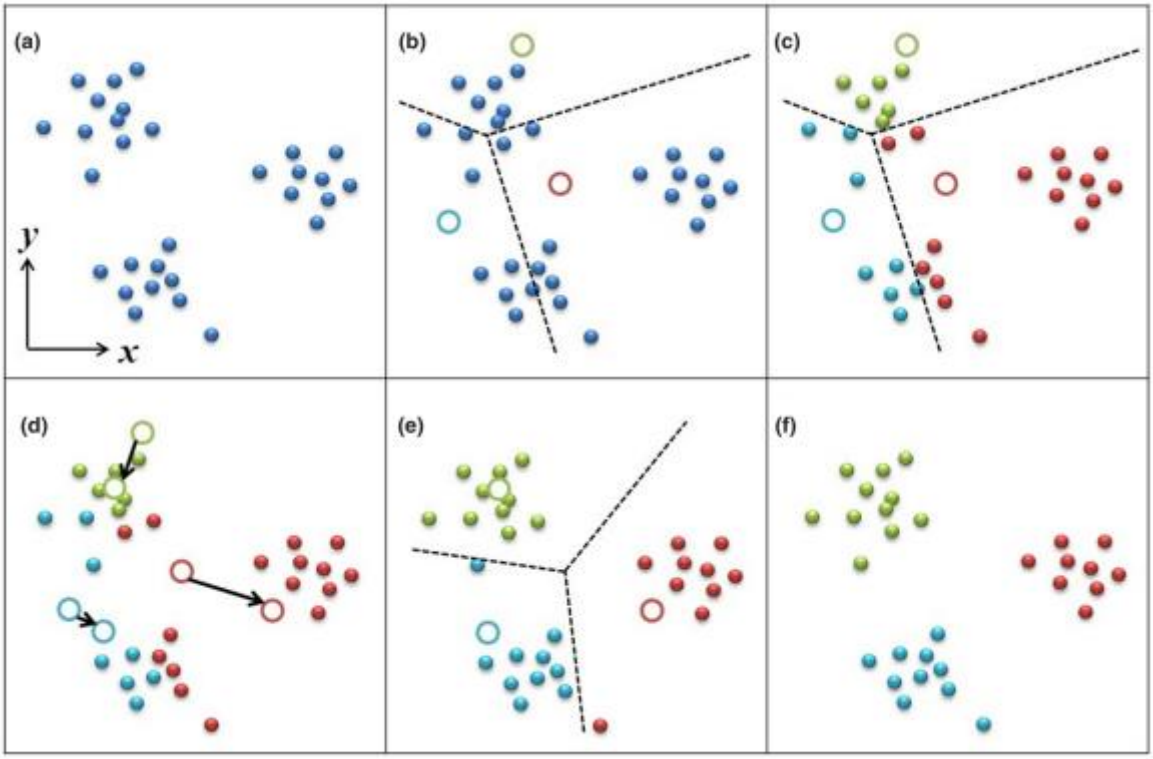


Figure 7 : Une illustration schématique de l'algorithme K-means pour le regroupement de données bidimensionnelles. (a) Les points de données (cercles bleus pleins) ont été regroupés dans un espace d'entités 2D. (b) Pour les emplacements aléatoires des centres de cluster (cercles creux aqua, verts et

rouges), chaque point de données peut être associé au centre le plus proche. (c) L'espace 2D est divisé en trois régions par trois limites de décision (lignes pointillées noires). (d) Chaque centre se déplace vers le centre de gravité des points de données qui lui sont actuellement attribués (mouvements indiqués par les flèches noires). (e) Le cluster mis à jour les attributions des points de données obtenues en fonction des nouveaux emplacements des centres. Les étapes (c) et (d) sont répétées jusqu'à ce que la convergence soit atteinte. (f) Les affectations finales aux grappes [12].

La première sélection de centroïdes détermine le résultat final. Lorsqu'un groupe de points est accepté, *K-Means* ajuste les points de chaque groupe afin que le total ne puisse pas être réduit. Le résultat est un ensemble de groupes clairement séparés et combinés, sous réserve de la sélection de la valeur correcte ***K*** pour le nombre de groupes. Chaque cluster est représenté par son centre.

K-means Algorithme :

1. Sélectionner ***K*** points comme centroïdes initiaux ;

2. Former *K* clusters en assignant chaque point au centroïde le plus proche ;

3. Recalculer le centroïde de chaque cluster nouvellement formé ;

4. Répéter les étapes 2 et 3 jusqu'à ce qu'aucun centroïde ne change.

Avantages :

- ✓ Facile à implémenter ;
- ✓ Fonctionne avec toutes les mesures standards ;
- ✓ Insensible à l'ordre des données

Inconvénients :

- ✓ Il n'est pas applicable en présence d'attributs qui ne sont pas du type numérique
- ✓ Le résultat final dépend fortement du choix des centroïdes initiaux
- ✓ Ne peut découvrir les groupes non-convexes
- ✓ Sensible à la présence de bruits

7.1.2 L'algorithme des k-médoïdes (k-medoids) :

K-médoïdes est parfois utilisé, où des objets représentatifs appelés médoïdes sont considérés à la place des centroïdes. Parce qu'il est basé sur l'objet situé le plus au centre d'un cluster, il est moins sensible aux valeurs aberrantes par rapport au clustering *K-means* [10]. Cette approche est liée au clustering **k-**

means, c'est-à-dire à la fois d'entre eux sont destinés à diviser un ensemble d'observations en $\boldsymbol{k}$ clusters de telle manière que les clusters minimisent une erreur critère de la somme des carrés (ESS : Error Sum-of-Squares) entre une observation et un centre du cluster.

Dans le clustering *k-médoides*, le critère ESS est formulé sur la base de la métrique de proximité prédéfinie comme suit [13] :

$$ESS = \sum_{r=1}^{k} \sum_{c(i)=k} \sqrt{(x_{i\_} c_k)^T (x_{i\_} c_k)^T}$$

où $\boldsymbol{c_k}$ est le médoïde du $k$ème cluster et $\boldsymbol{c(i)}$ est le cluster contenant $\boldsymbol{x_i}$**.** Pour plus d'informations, la Figure 1.8 illustre les organigrammes du regroupement des *k-médoïdes*.

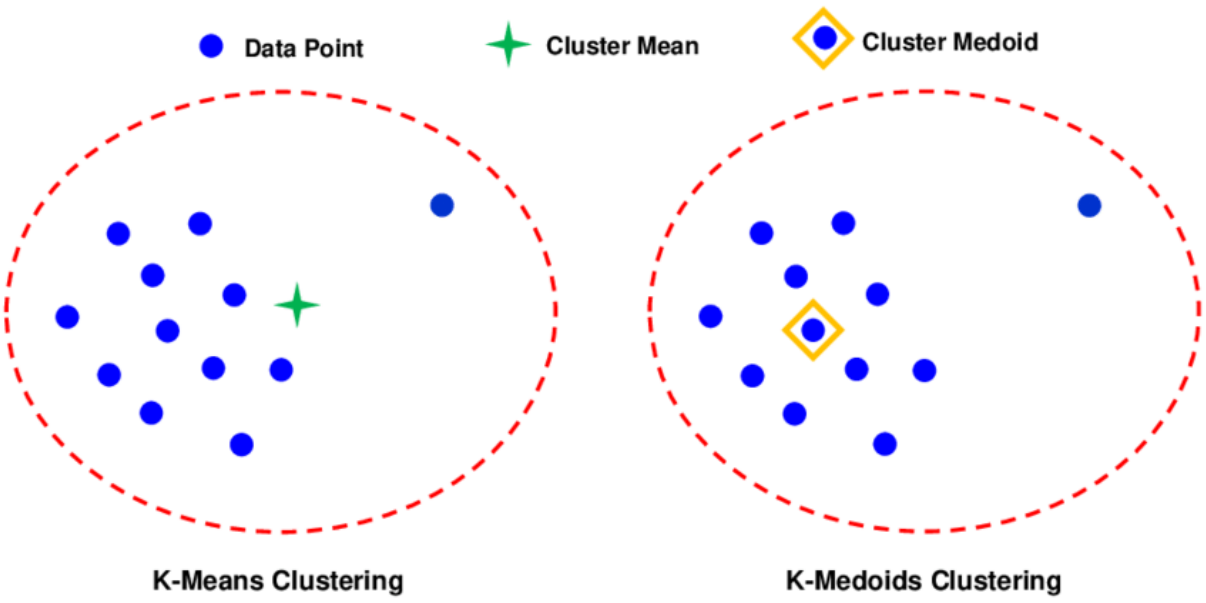


Figure 8 : La représentation graphique de la différence entre les méthodes de clustering k-means et k-médoides [13].

L'algorithme représentatif de cette famille est *PAMI* (Partitionig Around Médoids), l'avantage est moins sensible aux points bruits (points isolé) et applique n'importe quel type de variable (numérique et catégorielle), chaque cluster est représenté par un ou des objets du cluster.

K-médoides Algorithme :

1. Initialiser : sélectionnez $\boldsymbol{k}$ points aléatoires parmi les $\boldsymbol{n}$ points de données comme médoïdes.
2. Associez chaque point de données au médoïde le plus proche en utilisant n'importe quelle méthode métrique de distance commune.
3. Pendant que le coût diminue :

Pour chaque médoïde ***m***, pour chaque donnée ***o*** point qui n'est pas un médoïde :

1. Échangez ***m*** et ***o***, associez chaque point de données au médoïde le plus proche, recalculez le coût.
2. Si le coût total est supérieur à celui de l'étape précédente, annulez l'échange.

7.2 Le clustering hiérarchique :

Le clustering hiérarchique est un outil, parmi d'autres. Il est nécessaire dans la prise de décisions et il est basé sur une reconnaissance des similitudes entre les objets.

Les méthodes hiérarchiques génèrent une succession de partitions emboîtées les unes dans les autres au lieu d'une seule partition de l'espace des données [20]. Un cluster peut être divisé en sous clusters, l'ensemble des clusters étant généralement représenté par un arbre. Un objet appartient à une et une seule feuille dans la hiérarchie, mais également à son nœud père, et ainsi de suite jusqu'à la racine [21]. Les méthodes de clustering hiérarchiques se subdivisent en deux grandes familles :

7.2.1 Le clustering de division (descendant (Top down**)) :** l'ensemble d'individus est décomposé en ***K*** groupes [22]. On part d'un grand cluster que l'on divise en deux progressivement de façon à optimiser un critère donné pour obtenir au final un ensemble de singletons (des groupes qui contiennent un seul individu) [23] (Fig 9 (a)).

Algorithme :

1. Mettre tous les objets dans un seul cluster ;

2. Répéter jusqu'à atteindre la condition d'arrêt :

(a) Choisir un cluster à diviser ;

(b) Remplacer le cluster choisi par le sous cluster obtenu.

Les avantages :

- ✓ Flexibilité incluse concernant le niveau de la granularité.
- ✓ Facilite de manipuler toutes formes de similitude ou de distance.
- ✓ Applicabilité a tout type d'attribut.

Les inconvénients :

- ✓ Imprécision sur les critères d'arrêt.
- ✓ Coûteux en temps de calcul.
- ✓ La plupart des algorithmes hiérarchique ne revisitent pas les clusters une fois construits.

7.2.2 Le clustering d'agglomération (ascendant (Bottom up)**) :** l'ensemble d'individus est décomposé en une arborescence de groupes [22]. On commence avec autant de clusters qu'il y a d'objets. Ensuite,

à chaque étape, on regroupe les deux éléments (clusters) qui sont jugés les plus similaires pour terminer avec un seul grand cluster englobant toutes les données [23] (Fig 1.9 (b)).

Algorithme :

1. Initialement, mettre chaque objet dans son propre cluster ;

2. Parmi tous les clusters courants, sélectionner les deux clusters ayant la plus petite distance ;

3. Remplacer ces deux clusters par un nouveau cluster, formé par la fusion des deux clusters originaux ;

4. Répéter les étapes 2 et 3 jusqu'à atteindre la condition d'arrêt.

Les avantages :

- ✓ Pas besoin de fixé le ***k*** en avance.
- ✓ Meilleur analyse et compréhension
- ✓ On travaille à partir des dissimilarités entre les objets que l'on veut regrouper. On peut donc choisir un type de dissimilarités adapté au sujet étudié et à la nature des données.

Les Inconvénients :

- ✓ Calcul couteux
- ✓ On ne peut pas défaire les fusions.

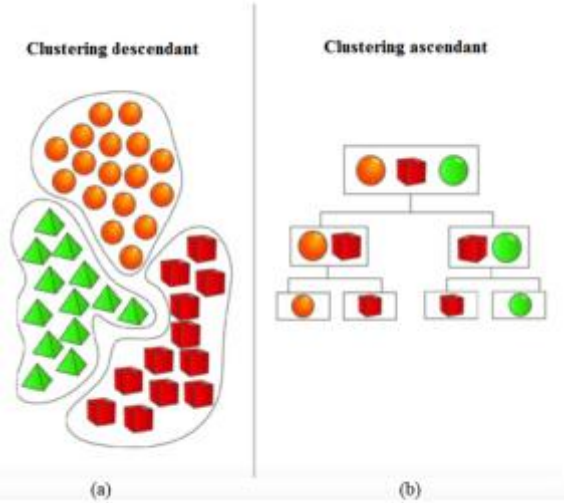


FIGURE 9 : Les types du clustering hiérarchiques [22].

7.2.3 Les algorithmes de clustering hiérarchique :

- SAHN (Sequential Agglomerative Hierarchical Non-overlapping) :

SAHN est une approche de clustering qui regroupe les méthodes de classification hiérarchique agglomérative (ascendant) qui sont considérés comme technique de clustering hiérarchique les plus populaires. Cette stratégie ascendante commence par placer chaque objet dans son propre cluster, puis fusionne ces clusters atomiques en clusters de plus en plus grands, jusqu'à ce que tous les objets soient dans un cluster unique en fonction de la distance entre les clusters ce qui donne naissance à quatre types importants d'algorithmes de liaison et qui sont : *Single-link* (SLINK), *Average-link* (ALINK), *Complete-link* (CLINK) et *Centoid-link* [30] [31]. Ces méthodes diffèrent par la manière dont leur matrice de similarité est initialement calculée ou dans leurs stratégies de regroupement des clusters à chaque itération.

Algorithme :

La description simplifiée de cet algorithme est la suivante [31] :

1. Chaque point de donnée est considéré comme un cluster ;

2. On calcule les distances entre les clusters ;

3. Les deux classes les plus proches sont fusionnées et remplacées par un seul ;

4. Le processus reprend en 2 jusqu'à n'avoir plus qu'une seule classe, qui contient toutes les observations.

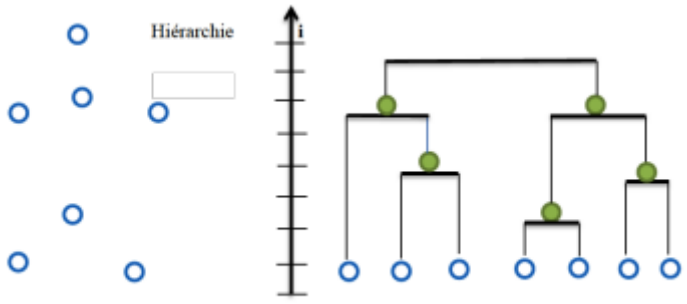


Figure 10 : La classification ascendante hiérarchique [31].

- Algorithme Single-Link :

En *SLINK* la distance entre deux clusters est représentée par la distance minimum entre toutes les paires de données entre les deux clusters (paire composée d'un élément de chaque cluster), nous parlons alors de saut minimum [31]. Le point fort de cette approche est qu'elle sait très bien détecter les classes allongées, mais son point faible est sa sensibilité à la présence de valeurs aberrantes et à la difficulté de faire face à de fortes différences dans la densité des clusters. D'un autre côté, elle affiche une insensibilité totale à la forme et à la taille des clusters [32].

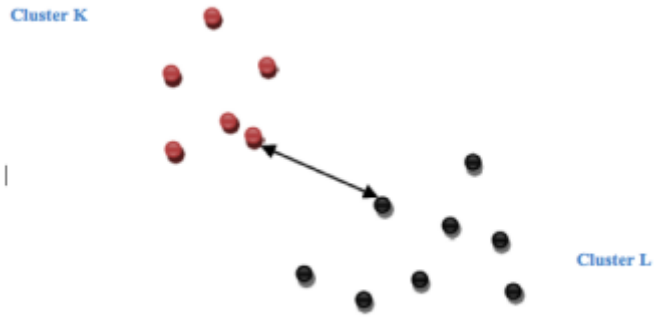


Figure 11 : Schéma de classification Single-Link [31].

- Algorithme Complete-link :

En *CLINK* la distance entre deux clusters est représentée par la distance maximum entre toutes les paires de données entre les deux clusters, nous parlons alors de saut maximum ou de critère du diamètre. Par définition cette approche est très sensible aux points aberrants donc elle est peu utilisée [31][32].

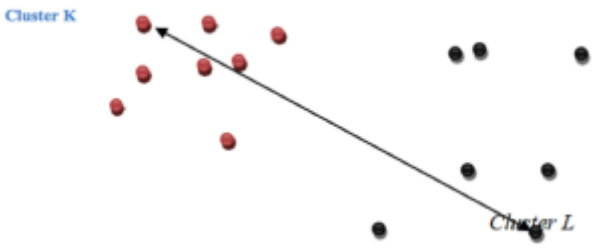


Figure 12 : Schéma de classification Complete-link [32].

- Algorithme Average-Link :

ALINK propose de calculer la distance entre deux clusters en prenant la valeur moyenne des distances entre tous les couples d'objets des deux clusters. Nous parlerons aussi de saut moyen [48]. Cette approche tend à produire des classes de même variance. La liaison moyenne est sensible à la forme et à la taille des grappes. Ainsi, il peut facilement échouer lorsque les grappes ont des formes compliquées s'écartant de la forme hyper sphérique [33].

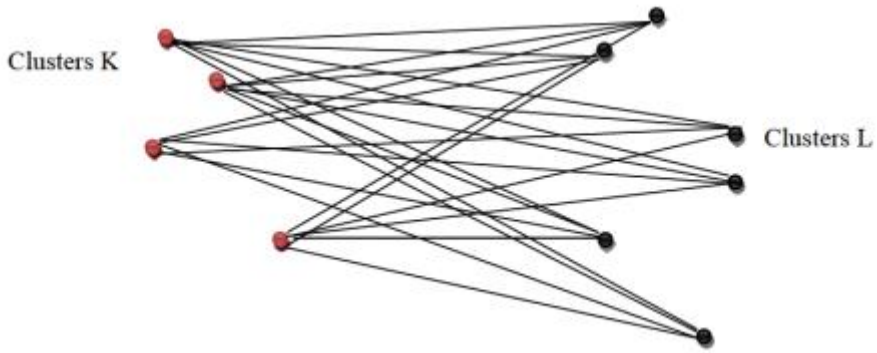


Figure 13 : Schéma de classification Average-Link [31].

- Algorithme Centoid-link :

Il définit, quant à lui, la distance entre deux clusters comme la distance entre leur centre de gravité. Une telle méthode est plus robuste aux points aberrants. Toutefois, elle est aussi limitée aux données quantitatives numériques pour lesquelles le calcul du centre de gravité est possible [31, 33].

Figure 14 : Schéma de classification Centoid-link [31].

7.3 le clustering par densités :

Les algorithmes basés sur la densité sont capables de découvrir des clusters de formes arbitraires, ce qui assure l'isolement des bruits (outliers) et la prévention contre la formation de clusters non pertinents [24].

Ces algorithmes groupent des objets selon des fonctions de densité spécifiques. La densité est habituellement définie comme nombre d'objets dans un voisinage particulier des éléments de données.

La densité des éléments dans chaque groupe est considérablement plus élevée qu'à l'extérieur du groupe et la densité dans les zones de bruit est inférieure à la densité dans l'un des groupes. Cette propriété permet facilement et sans ambiguïté de détecter des groupes et le bruit qui n'appartient à aucun de ces groupes [28].

Il existe deux types de méthodes basées sur la densité [26], [27] :

- Les méthodes connectives
- Les méthodes basées sur une fonction de densité

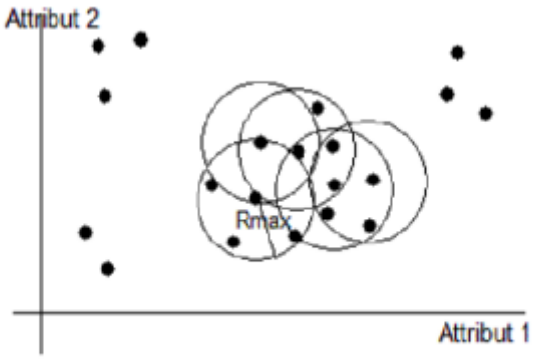


Figure 15 : Clustering basé sur la densité [25]

7.3.1 Méthodes basées sur la densité connective (Density-Based Connectivité Clustering) :

Dans cette technique de clustering, la densité et la connectivité sont mesurées en termes de distribution locale des voisins les plus proches [29]. C'est une relation d'équivalence. *DBSCAN* (Desity Based Spatial Clustering of Application with Noise) et *OPTICS* sont deux exemples de ces méthodes [34].La densité-connectivité ainsi définie est une relation symétrique et tous les points accessibles à partir des noyaux des objets peuvent être factorisés dans les composants connectés maximaux servant de clusters. Les points qui ne sont pas connectés à tout point du noyau sont considérés comme des bruits (outliers) et ils ne sont couverts par aucun cluster. Les points non fondamentaux à l'intérieur d'un cluster représentent sa borne. Finalement, les objets du noyau sont les points internes. Le processus est indépendant de l'ordre de données et n'a pas de limitation sur les dimensions ou le type d'attributs.

7.3.2 Méthodes basée sur les fonctions densité (DENSITY FUNCTIONS CLUSTERING) :

Une fonction de densité est utilisée pour le calcul de la densité. La densité globale est définie comme la somme des fonctions de densité de tous les objets. Les clusters sont déterminés par les attracteurs de densité qui sont définis comme les maxima locaux de la fonction de densité globale. [35].

8. Caractéristiques des méthodes de clustering :

Un algorithme de clustering doit répondre aux caractéristiques suivantes [36] :

- La capacité à gérer différents types d'attributs.
- La découverte de clusters avec des formes arbitraires : Les clusters peuvent avoir des formes diverses.
- Besoin minimum de connaissances du domaine pour déterminer les paramètres.
- La capacité à gérer le bruit et les exceptions.
- Non sensible à l'ordre des données en entrée.
- La capacité de gérer des données volumineuses.
- Les résultats de clustering doivent être interprétables, compréhensibles et bien évidements utilisables.

9. Technique de validation de clustering :

L'objectif principal de la validation de clusters est d'évaluer le résultat de clustering afin de trouver le meilleur partitionnement des données [37]. Chaque algorithme peut diviser les données, mais différents algorithmes ou paramètres d'entrée produisent différents résultats [38]. Ils existent en général trois approches de validation des algorithmes de clustering [38] :

• L'évaluation externe : il s'agit de confronter un schéma avec une classification prédéfinie. L'évaluation porte donc sur l'adéquation entre le schéma obtenu et une connaissance externe sur les données.

• L'évaluation interne : ce type d'évaluation utilise les données d'entrées comme référence. Ainsi, par exemple, parmi plusieurs schémas, le meilleur sera celui qui conserve le maximum d'informations.

• L'évaluation manuelle : elle se fait en faisant appel à un expert qui nous dira si un algorithme de clustering est bon ou non. Mais cela reste applicable sur des données de petite dimension.

10. Domaine d'application :

Le clustering a de nombreuses applications différentes dans le monde de machine Learning et data mining.

Ce type d'analyse peut traiter tous les grands ensembles d'informations, permettant d'obtenir d'excellents résultats avec de nombreux types de données. La technique de clustering est largement utilisée dans :

- Le Marketing : segmentation du marché en découvrant des groupes de clients distincts à partir de bases de données d'achats.
- L'environnement : identification des zones terrestres similaires dans une base de données contenant des informations (en termes d'utilisation) de la terre.
- Les assurances : identification de groupes d'assurés distincts associés à un nombre important de déclarations.
- La planification des villes : identification de groupes d'habitations suivant leur type, valeur, localisation géographique.
- La médecine : Localisation de tumeurs dans le corps humain. Par exemple, dans un nuage de points fournis par le scan du cerveau, on identifie les points définissant une tumeur.
- La segmentation d'images : Détection des zones homogènes dans une image.
- Web log analysis : Identification de profils d'utilisateurs à travers leur flux de clics (Clickstream).
- Text mining : Classification des textes selon leur similitude dans des dossiers automatiques.
- Bio-informatiques…etc.

11. Conclusion :

Dans ce chapitre, nous avons fait un tour d'horizon des principaux concepts, définitions et approche relatifs au problème du clustering. Nous avons également vue les méthodes et les approches et les algorithmes.

Nous avons focalisé notre attention sur les méthodes de la classification automatique (clustering) et sur les techniques d'évaluation de leurs algorithmes.

Le chapitre suivant sera consacré au Problème des emplois du temps des cours universitaires.

2

Problème des emplois du temps des cours universitaires

1. Introduction :

L'un des exemples les plus courants d'un problème d'optimisation dans une institution ou un établissement est l'emploi du temps. Il peut prendre de nombreuses formes différentes et spécifiques selon l'environnement ou l'institution où on a été exposé. Des méthodes classiques ou modernes peuvent être utilisées pour résoudre de tels problèmes. Cela a incité les chercheurs à proposer des méthodes et des techniques aider à gérer au maximum les heures de travail ou d'études. C'est pourquoi nous définissons les différents types de tables dans différents domaines et plus spécifiquement dans le domaine pédagogique universitaire.

Dans ce chapitre, nous allons présenter l'emploi du temps et ces formulations. Ensuite, nous allons citer les méthodes de génération d'un emploi du temps. En dernier nous parlerons de ses contraintes.

2. Définition de l'emploi du temps :

L'emploi du temps est un plan qui permet de présenter des évènements principaux et spécifiques en séquence sur une période de créneaux horaires. Il consiste à gérer les charges de travail dans le temps tout en prenant compte des ressources humaines et matérielles disponibles.

3. La formulation d'un emploi du temps pédagogique :

Il existe un grand nombre de variantes du problème d'emploi du temps qui diffèrent les unes des autres selon le type d'établissement concerné (université, école, etc.) et le type de contraintes. Parmi les nombreuses classifications possibles, nous en avons retenu trois principales dans le domaine pédagogique :

- ✓ *Emploi du temps des écoles* : un programme journalier pour toutes les classes de l'école, évitant que l'enseignant soit placé dans deux classes différentes en même temps.

- ✓ *Emploi du temps des cours* : le programme hebdomadaire pour tous les cours d'un ensemble d'université en réduisant au minimum possible les chevauchements des cours ayant les étudiants communs.

- ✓ *Emploi de temps des examens* : un programme pour les examens d'un groupe d'étudiants. L'emploi du temps d'un examen est un peu différant que l'emploi du temps des cours car il répartit autant que possible les examens entre les étudiants et évite les chevauchements d'examens de cours avec les étudiants participants, il y a un examen pour chaque matière, avec un ou deux examens pour chaque étudiant.

4. Méthodes de résolution :

L'objectif principal des méthodes proposées est de générer un emploi du temps valide qui, à la fois, satisfait toutes les contraintes dures et le maximum des contraintes souples qui concernent les enseignants et les étudiants.

Les méthodologies les plus récentes : l'approche centralisée, l'approche distribuée

4.1 Approches centralisées :

Les approches utilisées pour la résolution du problème d'EDT de façon centralisée sont :

La recherche opérationnelle **(RO),** l'intelligence artificielle **(IA),** théorie des graphes **(TG),** Méta-heuristique**,** L'hybridation de méthode.

4.1.1 Approches basées sur la recherche opérationnelle (RO) :

En général, la recherche opérationnelle fait référence à un ensemble de techniques et de méthodes permettant de déterminer la meilleure solution à des problèmes dont la résolution dépend d'une décision parmi un ensemble de solutions possibles.

Plusieurs approches pour résoudre le problème EDT ont été proposées, et elles sont classées en deux méthodologies : Méthodes de voisinage et de construction. Il existe trois approches de la méthode de voisinage : la recherche locale (RL), la recherche taboue et le recuit simulé (RS). Parmi les méthodologies de construction figurent les méthodes gloutonnes et le SEP (Separation and Progressive Evaluation)

4.1.2 Approches basées sur l'intelligence artificielle (IA) :

Dans l'IA, plusieurs approches sont utilisées pour la résolution de l'EDT. Elles sont classées sous trois catégories : les méthodes de construction, les méthodes de réparation et les méthodes d'évolution.

- Les méthodes de construction comprennent : les problèmes de satisfaction des contraintes (CSP) et l'algorithme A étoile (A*).
- Les méthodes de réparation comprennent : les heuristiques mini-conflit (MC) et les heuristiques random walk (RW).
- Les méthodes d'évolution contiennent : les algorithmes génétiques, les stratégies d'évolution et la programmation évolutive.

4.1.3 Approches basées sur théorie des graphes (TG) :

La théorie des graphes contenant des modèles et des applications pour résoudre plusieurs types de problèmes ; parmi les applications populaires : le Problème de coloration de graphe (Graph Coloring Problem).

4.1.4 Approches basées Méta-heuristique :

Les méta-heuristiques sont un type d'algorithme d'optimisation utilisé pour résoudre des problèmes d'optimisation courants dans la recherche opérationnelle, l'ingénierie et d'autres domaines. Il est plus efficace et plus intelligent de ne pas connaître la méthode exacte. Une méta-heuristique est formellement définie comme un processus génératif itératif qui guide une heuristique pour combiner intelligemment différents types de concepts afin d'améliorer les performances.

4.1.5 Approches basées sur L'hybridation de méthode :

Outre l'usage classique des méthodes complètes ou incomplètes, certains auteurs ont proposé d'autres techniques de résolution, telles que l'hybridation qui est une combinaison de deux familles de méthodes.

Les travaux de Teddy et Ruli [TR09] tentent de combiner des algorithmes génétiques avec une méthode de résolution C.S.P pour le problème d'emploi du temps universitaire. Ils sont basés sur quatre types de chromosomes (R1, R2, R3, R4), dont :

- R1 est un phénotype de type C.L.R.S (Course, Lecture, Room, Slot),

- R2 est le génotype de type L.R.S (Lecture, Room, Slot),

- R3 est le génotype de type L.R.S.R.S.F (Lecture, Room, Slot, Room, Slot, Function),

- R4 est le génotype L.R.S.R.S (Lecture, Room, Slot, Room, Slot).

4.2 Approches distribuées :

Les approches utilisées pour la résolution du problème d'EDT de façon distribuée sont : les systèmes multi-agents **SMA** et les problèmes de satisfaction distribuée des contraintes **DCSP**

4.2.1 Les systèmes multi-agents (SMA) :

Un S.M.A est basé sur une décomposition en quatre parties :

- ✓ *Agents A,* qui concernent les modèles ou architectures utilisées pour la partie active de l'agent, depuis un simple automate jusqu'à un système complexe à base de connaissances.
- ✓ *Environnements E,* qui sont des milieux dans lesquels évoluent les agents. Ils sont généralement spatiaux.
- ✓ *Interactions I*, qui englobent les infrastructures, les langages et les protocoles d'interactions entre agents, depuis les interactions physiques jusqu'aux interactions par actes de langage.
- ✓ *Organisations O*, qui structurent les agents en groupes, hiérarchies, relations.

4.2.1.1 Types des systèmes multi-agents :

Les SMA se diffèrent selon différents critères. Selon l'interaction on trouve :

- ✓ SMA Ouverts : Dans ce SMA, aucune barrière n'est imposée à l'agent (concernant leur migration). En effet, les agents peuvent entrer, sortir et changer alentour. Ce type de SMA est généralement utilisé dans les systèmes qui nécessitent une ressource (physique ou virtuelle) qui ne peut être que dans un autre système.
- ✓ SMA Fermés : L'échange n'est pas autorisé, les agents ne peuvent interagir qu'au sein de leur environnement et doivent se contenter de ce qui y existe.

Si par contre, nous considérons les SMA selon le type d'agents les constituant :

- SMA Hétérogènes : Différents types d'agents peuvent se trouver au sein du même SMA. Les agents sont créés sur des modèles différents, mais arrivent à coexister ensemble dans le même environnement grâce aux standards de communication.
- SMA Homogènes **:** Le système multi-agents est composé de plusieurs agents du même type.

4.2.1.2 Domaines d'application des SMA :

L'utilisation des systèmes multi-agents couvre aujourd'hui différents domaines tels que l'industrie, la santé, l'information, …

Quelques cas réels d'application où ils ont énormément réussi :

- Apprentissage électronique (E-learning).
- Commerce électronique (E-commerce).
- Recherche d'information.

4.2.2 Les problèmes de satisfaction des contraintes distribuées (DCSP) :

Un DCSP est un CSP dans lequel les variables et les contraintes sont distribuées entre des agents. Ces Contraintes se divisent en deux ensembles disjoints : les contraintes intra-agents et les contraintes inter-agents. Une contrainte intra-agent n'est connue que par un agent. Habituellement, on considère qu'une contrainte inter-agent est connue par tous les agents ayant une variable dans cette contrainte. Comme dans le cas centralisé, résoudre un DCSP consiste à trouver une assignation de valeur aux variables en ne violant aucune contrainte (bien que la littérature des DCSP se concentre principalement à la résolution des contraintes inter-agents). Trouver une assignation de valeur aux variables inter-agents peut être vu comme réaliser la cohérence ou la consistance d'un système multi-agent. Les heuristiques de résolution d'un DCSP se classent en deux catégories. D'une part, la programmation orientée-marché repose sur les mécanismes d'enchères. D'autre part, des agents peuvent être en compétition pour utiliser des ressources, ces ressources pouvant calculer leur demande. En outre, les algorithmes de résolution des CSP semblent similaires à des méthodes de traitement parallèle/distribué pour résoudre des CSP, quoique les motivations de recherche soient fondamentalement différentes.

5. Problèmes d’un emploi du temps :

5.1 Présentation de Problème d’emploi du temps :

Le Problème d’emploi du temps consiste à affecter un ensemble de tâches et/ou d'activités à un groupe de ressources selon des intervalles de temps bien définis, soumis à un ensemble de contraintes qui doivent être respectées.

Une tâche est un travail élémentaire dont la réalisation nécessite un certain nombre d'unités de temps (sa durée) et d'unités de chaque ressource. Une ressource est un moyen technique ou humain, dont la disponibilité limitée ou non connue a priori [46].

Le problème d'emploi du temps consiste à définir un certain nombre d'affectations qui permet d'assigner plusieurs ressources sur une période fixe de temps. Il s'adresse habituellement aux organisations où un ensemble de tâches doit être accompli par un ensemble d'employés ayant leurs propres qualifications, contraintes et préférences. Les contraintes peuvent différer d'un problème à un autre suivant la spécificité ainsi que les caractéristiques attendues de l'emploi du temps recherché. Ces contraintes sont souvent classées en deux catégories, la première regroupe les contraintes dures (un emploi du temps qui ne satisfait pas ce genre de contraintes est infaisable, le groupe de deuxième catégorie des contraintes appelés souvent contraintes molles, souples ou de préférence dont la satisfaction a différents degrés d'importance mais dont le non-respect n'empêche pas une application plus au moins acceptable de l'emploi du temps trouvé. Typiquement ces contraintes de préférence sont utilisées pour exprimer ce que doit être un « bon » emploi du temps. Ces contraintes sont plus difficiles à formaliser que les contraintes dures et leurs traitements sont plus délicats. En ce qui nous concerne, nous nous sommes concentrés plus précisément sur le problème d'emploi du temps universitaire [47].

5.2 Génération automatique d'un emploi du temps [44] :

La génération automatique d'un emploi du temps est une activité qui consiste à créer, gérer et maintenir un EDP à l'aide d'un ordinateur et avec le moins d'intervention humaine possible tout en satisfaisant les ressources humaines, matérielles et temporelles.

La génération automatique d'un horaire nécessite le développement d'un système capable de gérer les ressources temporelles et matérielles, les contraintes imposées, et résoudre les conflits de ressources. La création d'un tel système est une tâche difficile et chronophage pour les humains. Un emploi du temps est également très compliqué et coûteux à produire manuellement.

5.3 Définition formelle du problème de l'emploi du temps :

Un PET se définit comme la gestion d'un ensemble de ressources (matérielles, humaines et temporelles) ainsi que la satisfaction d'un ensemble de contraintes.

Le PET consiste à équilibrer les contraintes de temps et de ressources afin de proposer un emploi du temps à plusieurs acteurs (professeurs, étudiants, cours, salles, etc.) sur une période de temps précise. Dans ce cas, les objectifs sont prédéterminés en fonction d'un projet pédagogique et de contraintes telles que :

- Disponibilité temporelle ;
- Compétences ;
- Le besoin de ressources spécifiques (matériels pédagogiques).

Formellement on peut représenter le problème de l'emploi du temps comme suit :

- un ensemble de classes : C_1 ..., C_m ; avec ***m*** est le nombre de classes,
- un ensemble de professeurs t_1 ..., t_n ; avec ***n*** est le nombre de professeurs,
- un ensemble de périodes de 1..., h avec ***h*** est le nombre de périodes.

Une matrice (x_{ijk}) indique les contraintes entre les éléments (salle, prof, cours). La formulation mathématique est comme suit :

EPT : consiste à Trouver $x_{ijk}(\mathrm{i} = 1 \ldots \mathrm{m}; \mathrm{j} = 1 \ldots \mathrm{n}; \mathrm{k} = 1 \ldots \mathrm{h})$ où

$$x_{ijk} = \begin{cases} 1 \; si\ la\ classe\ Ci\ \ et\ le\ professeur\ tj\ se\ réunissent\ \text{à}\ la\ période \\ 0 \qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad Dans\ le\ cas\ contraire \end{cases}$$

5.4 Présentation des données :

Les modèles mathématiques proposés dans la littérature sont souvent trop restrictifs pour prendre en compte toutes les contraintes qui interviennent dans le processus de construction d'un emploi du temps. L'algorithme d'optimisation, qu'il s'agit de concevoir, doit être suffisamment flexible pour tenir compte des particularités propres de l'établissement et des différents changements qui peuvent surgir par la suite.

Les éléments d'information ci-dessous sont pris en considération lors de l'établissement d'un emploi du temps de cours semestriel :

A. Un ensemble de périodes P= {p_1, ..., p_{np}} pendant lesquelles différentes leçons peuvent être placées.

B. Un ensemble d'intervenants PR= {pr_1, …, pr_{npr}}

C. Un ensemble de Groupes GR= {gr_1, …, gr_{ngr}}. Chaque groupe regroupe un ensemble d'étudiants qui ne possède pas forcement le même programme d'études.

D. Un ensemble de filières F= {f_1, …, f_{nf}}

E. Un ensemble de salles S= {s_1, …, s_{ns}}. Il y a plusieurs types de salle, salle de cours, de Travaux dirigés (TD), Travaux pratiques (TP) et salles de Sport (gymnase).

F. Un ensemble de sites ST= {st_1, …, st_{nst}} distant les uns par rapport aux autres. Deux sites sont distants si les intervenants et les étudiants n'ont pas suffisamment de temps pour se déplacer de l'un à l'autre durant une pause.

G. Un ensemble d'unités de Valeur UV= {uv_1, …, uv_{nuv}}, chaque cours uvi est caractérisé par les informations suivantes :

- Un ensemble d'intervenants,
- Un ensemble de groupes,
- Un ensemble de filières,
- Le nombre de séance de cours durant un semestre,
- Le nombre de séance de TD durant un semestre,
- Le nombre de séance de TP durant un semestre,
- Le site sur lequel le cours doit être donné,
- L'état d'espacement (nombre de séance de cours à effectuer avant un TD ou un TP, …).

On note UP= {up_1, …, up_{nup}} la liste des séances qui résulte de la donnée de l'ensemble d'UV = {uv_1, …, uv_{nuv}}.

5.5 Formulation des contraintes :

Un horaire peut être considéré comme satisfaisant s'il convient aux professeurs, aux étudiants et s'il respecte la disponibilité des salles. Le souci d'assurer un certain confort aux professeurs et aux étudiants rend particulièrement difficile la confection d'un emploi du temps. Il est bien difficile de recenser toutes les exigences susceptibles, d'intervenir dans un problème d'emplois du temps. Certaines découlent directement des données décrites dans la section précédente alors que d'autres viennent se superposer selon le cas. Nous mentionnons ci-dessous les contraintes les plus fréquentes et celle dont nous allons tenir compte par la suite.

En clair, le problème de la confection d'un emploi du temps universitaire consiste à donner une heure, à chaque leçon upi{up_1, …, up_{nup}} tout en satisfaisant les contraintes suivantes :

1. *Non chevauchement des Intervenants :* un intervenant ne peut pas enseigner plus d'une leçon à la fois.
2. *Contraintes de site pour Intervenants :* L'enseignant ne change pas de site au cours d'une demi-journée.
3. *Contraintes de pause pour Intervenants :* L'enseignant a au moins une pause de 45 minutes aux heures de midi.
4. *Disponibilité des Intervenants :* L'emploi du temps essaie de respecter la journée de disponibilité des enseignants. Aucune différence n'est faite entre les intervenants extérieurs et le personnel de l'université.
5. *Non chevauchements des étudiants :* Un étudiant ne peut suivre qu'une seule unité pédagogique à la fois.
6. *Contraintes de pause Étudiants :* L'étudiant a au moins une pause de 45 minutes aux heures de midi.
7. *Contraintes de site Étudiants :* L'étudiant ne change pas de site au cours d'une demi-journée.
8. *Contraintes d'affectation d'étudiants :* Un étudiant est affecté à un seul groupe de cours, TD ou TP par UV.
9. *Contraintes sur les horaires des UPs :* On ne peut commencer une unité pédagogique si on ne peut la finir dans les horaires prédéfinis. Par exemple, on ne peut commencer un cours de deux heures à 19h.
10. *Contraintes de positionnement des UPs :* Toutes les unités pédagogiques doivent être positionnées.
11. *Contraintes d'affectation de salles :* Dans une salle, il ne peut y avoir qu'une seule unité pédagogique programmée par créneau horaire
12. *Contraintes de choix de salles :* les affectations des salles doivent respecter les demandes des responsables d'UV.
13. *Contraintes de capacité de salle :* Une unité pédagogique ne peut être planifiée dans une salle ne permettant pas de recevoir l'ensemble des étudiants lié à cette unité.
14. *Contraintes jours non ouvrés :* aucun enseignement ne peut avoir lieu durant une journée non ouvré.

L'existence d'un horaire respectant à la lettre toutes les contraintes n'est pas garantie dans le cas général. En réalité, il s'agit de trouver un horaire aussi bon que possible dans un temps de calcul

raisonnable tout en tolérant la présence de conflits ; le nombre de ces conflits doit être minimisé pour que l'horaire soit jugé satisfaisant par les différents parties concernées (direction de l'université, Intervenant, Etudiants, etc.)

L'emploi du temps est un problème difficile à élaborer et plus demandé à tous les niveaux dans divers domaines. La difficulté principale vient de la complexité de gérer les contraintes temporelles qui le caractérisent. Donc, la génération d'un emploi du temps :

- Est difficile car c'est typiquement un problème de résolution de contraintes dont la solution n'est pas à priori connue dans le cas général ;
- Nécessite de fournir un programme capable de s'adapter pour réagir aux changements dynamiques de l'environnement pour fournir une solution [45].

5.6 Le PET comme problème de satisfaction des contraintes :

Les acteurs mis en jeu pour l'élaboration d'un emploi du temps au niveau scolaire sont :

- Des enseignants.
- Des groupes d'étudiants.
- Des salles.
- Des modules.

Il y a plusieurs contraintes. On peut les classer en deux catégories : les contraintes fortes et les contraintes faibles. Les contraintes fortes doivent être satisfaites mais les contraintes faibles peuvent être violées. Chacun de ces acteurs possède des contraintes fortes et faibles.

Les contraintes des enseignants :

- *Fortes*
  - Sa disponibilité (jour de la semaine, tranche horaire, ...)
  - Ses propres qualifications,
  - Ses préférences
  - Sa possibilité d'enseigner un ou plusieurs modules
- *Faibles*
  - Ses compétences,
  - Son niveau d'études
  - Son type de contrat (permanent ou contractuel)

Les contraintes des ressources matérielles :

- *Fortes :*

- Disponibilité des salles et appareils (data show...).

- *Faibles :*
  - Les équipements spéciaux (chauffage, ventilateur, climatiseur ...)

Les contraintes des modules :

- *Fortes :*
  - Chaque module doit être enseigné au moins par un professeur.

Les contraintes des groupes d'étudiants :

- *Faibles* :
  - Chaque groupe a besoin d'une pause.

6. Problèmes d'emploi du temps universitaire :

Le problème d'emploi du temps universitaire est un exemple des problèmes d'ordonnancement cyclique les plus connus dans la littérature, il s'agit d'ordonnancer les tâches (qui ne possède pas le caractère cyclique) d'un ensemble d'enseignants en leur allouant un ensemble de salles et en leur fixant leurs dates de début et de fin. De nos jours, la construction d'un calendrier qui satisfait tous les besoins d'un établissement universitaire est une tâche très importante, mais elle est extrêmement difficile. Avec le progrès réalisé dans les technologies matérielles et logicielles, la communauté scientifique continue à travailler sur ce problème, afin d'élaborer des procédures formelles et automatisés pour confectionner des calendriers efficaces et souhaitables. [47]

Burke et ses collègues [48] notent à cet impératif que ce type de problème se di-étau en deux catégories principales : les cours et les examens. Déférents aspects séparent ces deux catégories. Par exemple, on cherche à regrouper les cours, alors qu'on préfère éloigner les examens les uns des autres le plus possible. Ou encore, un cours peut se tenir à un instant donné dans une salle, alors que plusieurs examens peuvent se tenir en même temps dans la même salle ou un même examen peut être dispatché dans plusieurs salles [49][47].

6.1 Problèmes d'emploi du temps des cours universitaires :

Dans notre problématique, nous choisissons de traiter le problème d'emploi du temps universitaires des cours, qui se définie comme un ensemble de cours universitaire qui se déroulent tout au long des périodes spécifiques pendant cinq ou six jours par semaine, dirigés par un nombre limité de

professeurs et de salles de classe nécessitant une meilleure gestion pour bien contenir le nombre important d'étudiants inscrits [47].

6.1.1 Différent type de contrainte :

Une contrainte ne revêt pas nécessairement un aspect absolu (soit elle est vérifiée ou violée) mais peut être formulée sous un objectif qui doit être approché autant que possible, selon ses critères. Ces contraints sont souvent classées en deux catégories :

6.1.1.1 Contrainte dure (hard) :

Ce sont des contraintes qui doivent être satisfaites dans n'importe quel environnement, car briser ces Contraintes peuvent entraîner la génération d'une solution insatisfaisante.

-Deux lectures ne peuvent pas être programmées dans la même pièce et dans la même période de temps.

-Les lectures à donner par le même professeur ne peuvent pas être programmées dans la même période de temps.

-Une pièce ne peut être affectée qu'à une seule lecture dans une période de temps.

-Une séance de lecture collective ne peut avoir lieu dans la même période avec une autre qui n'est pas dans un groupe pour le même cours ayant le même niveau d'études.

-Le nombre d'élèves doit être inférieur ou égal à la capacité de la salle à attribuer.

6.1.1.2 Contrainte faible (soft) :

La violation de ces contraintes n'a aucun effet sur la génération d'une solution satisfaisante.

- L'affectation des salles et des plages horaires doit permettre de satisfaire les Préférences des enseignants.

- L'affectation des salles aux différentes lectures doit permettre de satisfaire au mieux certaines préférences.

Les domaines d'application :

L'emploi du temps est utilisé en général dans tous les domaines :

- ✓ Économiques,
- ✓ Industriels,
- ✓ Administratifs,
- ✓ Les établissements scolaires et les universités,
- ✓ Les instituts,

- ✓ Les entreprises…

7. Conclusion :

Dans ce chapitre, nous venons de voir ce que c'est qu'un emploi du temps, sa définition, nous avons mentionné la formulation d'un emploi du temps pédagogique et les méthodes de résolution, défini une génération automatique d'un EDT.

Nous avons conclu par les contraintes d'un problème d'emploi du temps et ces domaines d'application. Le chapitre suivant sera consacré pour les Métaheuristiques.

3

Les Métaheuristiques

1. Introduction :

Les métaheuristiques forment un ensemble de méthodes utilisées en recherche opérationnelle et en intelligence artificielle pour résoudre des problèmes d'optimisation réputés difficiles. Résoudre un problème d'optimisation combinatoire, c'est trouver l'optimum d'une fonction, parmi un nombre fini de choix, souvent très grand. Les applications concrètes sont nombreuses, que ce soit dans le domaine de la production industrielle, des transports ou de l'économie – partout où se fait sentir le besoin de minimiser des fonctions numériques, dans des systèmes où interviennent simultanément un grand nombre de paramètres.
Dans ce chapitre, nous présentons d'abord les Heuristique et Métaheuristique, ensuite nous donnerons Les méthodes de recherche locale de Métaheuristique et Les solutions, et nous terminerons par quelques exemples de domaines d'applications.

2. Heuristique et Métaheuristique :

Pour saisir la signification de l'heuristique et de la métaheuristique, il faut d'abord saisir le concept de complexité. Ce dernier désigne les ressources nécessaires à l'exécution d'un algorithme ; généralement, le temps de calcul est utilisé pour le déterminer. La complexité est divisée en deux types de problèmes : P et NP. La classe P rassemble les problèmes qui peuvent être résolus en un temps polynomial. Les problèmes de classe P sont considérés comme efficacement solubles ; sinon, le problème est considéré comme difficile. Les problèmes de la classe NP sont ceux dont il est possible de vérifier que la solution peut être trouvée en temps polynomial. Si P est contenu dans NP, la réciproque est incertaine, et le problème P = NP est resté irrésolu à ce moment.

En conséquence, nous comprenons le désir de développer des méthodes efficaces pour résoudre les problèmes difficiles le plus rapidement possible. Les méthodes d'optimisation sont classées en plusieurs familles : mono-objectif vs multi-objectif, continu vs discret, parallélisé vs non parallélisé, etc. L'heuristique et la métaheuristique sont des méthodes stochastiques, ce qui signifie que l'algorithme ne renverra pas la même solution après deux exécutions indépendantes en raison du caractère aléatoire de la méthode.

Le terme heuristique vient du mot grec eurisko, qui signifie « trouver », et il fait référence à tout ce qui est utile pour la découverte, l'invention ou la recherche. Les heuristiques sont des méthodes d'optimisation pour résoudre des problèmes difficiles qui sont relativement simples et rapides (avec une complexité polynomiale en temps). Contrairement aux métaheuristiques, ces méthodes sont généralement spécifiques à un type de problème et donc dépendantes de celui-ci. Le terme métaheuristique est dérivé du mot grec meta, qui signifie "au-delà" et peut être traduit par "à un niveau supérieur", et de l'heuristique ou euristique (du grec ancien εὑρίσκω, heuriskô, « je trouve ») est « l'art d'inventer, de faire des découvertes » en résolvant des problèmes à partir de connaissances incomplètes [51].

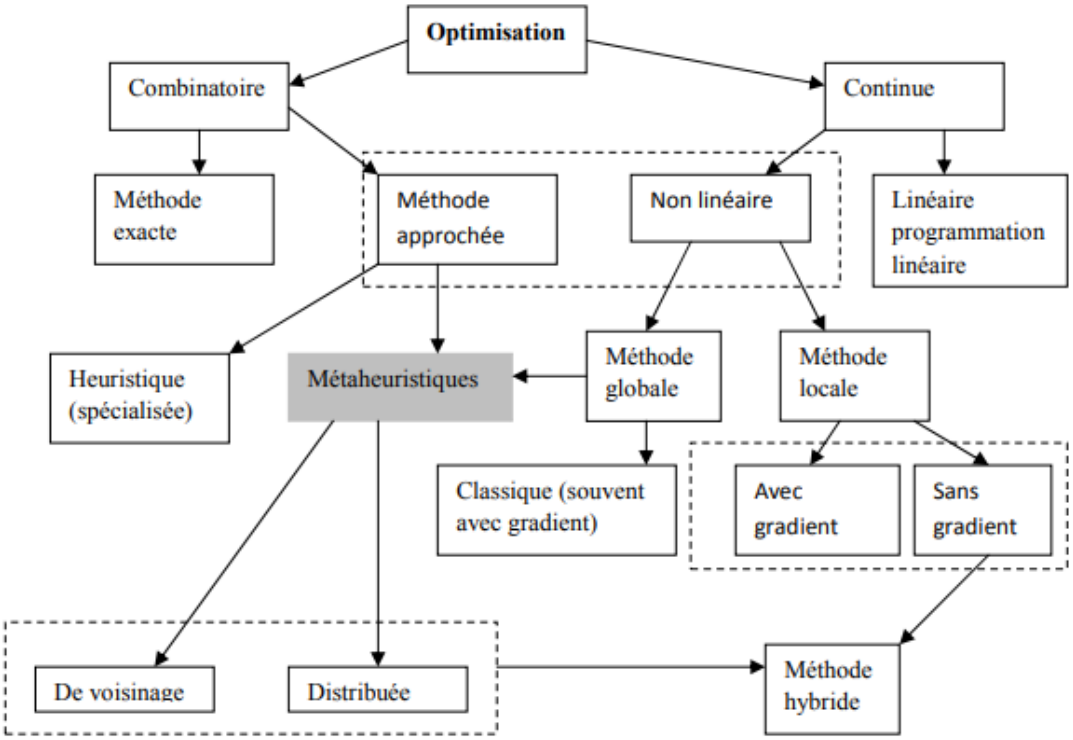


Figure 16 : Présentation des différents algorithmes d'optimisation [58].

3. Les méthodes de recherche locale de Métaheuristique :

Le mécanisme de les métaheuristiques à solution unique consiste à faire évoluer itérativement une solution dans l'espace de recherche pour atteindre l'optimum global.

Les méthodes les plus connues qui sont : la méthode de descente, le recuit simulé et la recherche tabou.

3.1 la méthode de descente :

En matière d'optimisation, l'approche par descente peut sembler la plus simple et la plus directe. En maximisant les problèmes, il est également connu sous le nom d'escalade. L'algorithme commence par sélectionner une solution au hasard, puis sélectionne la meilleure solution dans le voisinage de la solution actuelle à chaque itération. Lorsqu'aucun progrès supplémentaire n'est concevable, l'algorithme s'arrête. Il existe plusieurs stratégies différentes pour sélectionner la meilleure solution dans le voisinage : l'algorithme peut choisir celle qui présente le meilleur fitness par rapport à toutes les autres solutions du voisinage, ou il peut choisir la première solution du voisinage qui améliore la fitness, il peut aussi choisir la solution qui améliore le moins la fitness ("la pire solution"), il peut aussi choisir la solution au hasard, et ainsi de suite. L'inconvénient fondamental de cette méthode de descente est qu'elle a tendance à se coincer dans un optimum local.

Lorsqu'un optimum local est trouvé, cette technique est améliorée en lançant de nombreux redémarrages, chacun partant d'une nouvelle solution générée aléatoirement ; c'est ce qu'on appelle une descente avec algorithme de récupération (ou escalade à redémarrage aléatoire).

3.2 le recuit simulé :

Le principe du recuit simulé, souvent appelé $Simulated\ Annealing$ (SA), est dérivé de la métallurgie. Basée sur les règles de la thermodynamique de Boltzmann, elle consiste à effectuer des cycles de refroidissement et de chauffage lents d'un matériau afin de diminuer son énergie. $Kirkpatrick, C. D. Gelatt$ et $M. P. Vecchi$ a modifié cette méthode en optimisation en 1983, puis $V.$ Erny en 1985 : la fonction objectif correspond à l'énergie du matériau, et les paramètres de l'algorithme sont le critère d'arrêt, la température de début To (de nombreuses façons existent , la fonction de déclin de température, et l'exigence de pas de changement de température La température chute pendant la recherche de l'optimum, l'algorithme commence par une marche aléatoire, puis les mauvaises solutions sont de moins en moins acceptées, et la méthode converge vers une solution d'espace de recherche. De ce fait, un compromis doit être recherché pour adapter la diminution de température afin d'éviter une diminution trop brutale, ce qui risque de bloquer l'algorithme dans un optimum local. D'ailleurs, il existe plusieurs lois de décroissante de la température [52][53].

Le recuit simulé a été adapté pour résoudre les problèmes d'optimisation continue par Patrick Siarry [54], de plus il a connu un large essor dans différents domaines d'application.

3.3 La recherche Tabou :

La méthode de recherche tabou a été proposée par Fred Glover en 1986 [51], elle introduit la prise en compte du passé par l'utilisation d'une mémoire des solutions explorées lors de la recherche de l'optimum. Cette mémoire est appelée liste taboue car elle contient les solutions déjà été visitées et par lesquelles il est interdit de repasser. Parce qu'elle stocke les solutions déjà visitées et sur lesquelles il est interdit de revenir, cette mémoire est connue sous le nom de liste tabou. Cette mémoire est utilisée pour empêcher l'algorithme de rester bloqué dans un optimum local en lui permettant de parcourir les

options qui dégradent la forme physique. La taille de la mémoire, ou le nombre d'éléments mémorisés, est un paramètre de l'algorithme qui permet à l'algorithme de rendre compte de l'équilibre précité entre diversification et intensification. En effet, si la mémoire est faible, l'intensification sera favorisée car seul un nombre limité de solutions sera autorisé. Cependant, plus la taille de la mémoire augmente, et plus la diversification est favorisée, car l'algorithme pourra de moins en moins visiter les régions précédentes qui ont de grandes chances d'être voisines à la solution actuelle. La procédure de cette métaheuristique commence par une première solution initiale qui est générée aléatoirement, puis la méthode va sélectionner itérativement la meilleure solution dans son voisinage et mémoriser la solution précédente. Ceci permet alors d'éviter les phénomènes dits de cyclage (l'algorithme tourne en boucle sur les mêmes solutions).Il existe des méthodes ayant une mémoire adaptative [55], c'est-à-dire que la taille de la mémoire va pouvoir s'adapter selon le contexte de la recherche et même créer de nouvelles solutions.

4. Métaheuristiques à population de solutions :

4.1 Méthodes constructives :

L'idée de base des méthodes constructives est de produire une solution, en partant d'une combinaison vide, par l'ajout itératif des composants à la sous-solution existante (solution partielle à une itération donnée). Les choix partiels sont définitifs. Les méthodes constructives sont largement utilisées comme des auxiliaires dans d'autres méthodes. Elles sont introduites généralement pour calculer des solutions initiales, ou à l'intérieur d'autres procédures.

4.1.1 Optimisation par colonies de fourmis :

En observant une colonie de fourmis à la recherche de nourriture dans les environs du nid, on s'aperçoit qu'elle résout des problèmes tels que celui de la recherche du plus court chemin. Les fourmis résolvent des problèmes complexes par des mécanismes assez simples à modéliser. Il est ainsi assez simple de simuler leur comportement par des algorithmes. Les fourmis virtuelles ont une double nature. D'une part, elles modélisent les comportements abstraits de fourmis réelles, et d'autre part, elles peuvent être enrichies par des capacités que ne possèdent pas les fourmis réelles, afin de les rendre plus efficaces que ces dernières.

4.1.1.1Principe de l'algorithme [56] :

Les développeurs d'antNet on dû faire quelques adaptations aux algorithmes méta-heuristique déjà existants, que se soit la modélisation des fourmis ou le fonctionnement de l'algorithme.

4.1.1.2 Les fourmis du net :

Les fourmis utilisées pour la gestion des tables de routage sont de deux types différents. Ainsi, il existe une fourmi "test" qui va suivre les branches du réseau comme si elle était un paquet et une fourmi "compte-rendu" qui a pour rôle de mettre a jour les tables (fourmis respectivement nommées *F-ant* et *B-ant*)

La première fourmi est un peu plus intelligente qu'un insecte lambda car elle a la possibilité de mémoriser l'état du réseau des noeuds qu'elle traverse. En pratique, elle garde en mémoire le temps de parcours et le chemin suivi. Le comportement de cette fourmi est dicté par une loi probabiliste que nous verrons plus loin et elle est également capable d'engendrer une fourmi B-ant à la fin de son parcours avant de mourir. Les fourmis B-ant quant à elles ont un chemin pré-determiné puisqu'elles parcourent uniquement les noeuds dans le sens inverse de la F-ant l'ayant créée, mettant à jour les tables de routage de ceux-ci. En effet, grâce aux informations de la F-ant, la B-ant est au courant de l'encombrement du réseau permettant ainsi au routeur de s'adapter aux fluctuations.

4.1.1.3 Mécanisme de antNet :

L'algorithme d'antNet est très proche de ceux utilisé pour le TSP, les routeurs pouvant être associés aux villes, ceci dit il y a quelques différences notables :

– il n'y a aucune contrainte sur les villes à visiter, c'est-à-dire qu'une fourmi n'est pas obligée de visiter tous les noeuds.

– il y a une piste de phéromone différente pour chaque nœud de destination.

– la dimension temporelle du problème rend plus délicate son évaluation.

Algorithme :

1. Chaque nœud lance une fourmi $F - ant$ avec une destination aléatoire [56].
2. Chaque fourmi est routée grâce à la fonction stochastique f prenant en paramètre $\tau_{ijd}(t)$, le taux de phéromones sur la piste et $\mu_{ij}(t)$, l'information spécifique au problème (ici μ_{ij} est proportionnel à la liste d'attente entre les noeud i et j).
    i. $p_{ijd}^{k}(t) = f(\tau_{ijd}d(t), \mu_{ij}(t))$
3. Chaque fourmi mémorise son parcours et le délai entre chaque nœud. Elle incrémente également le taux de phéromone des pistes traversées.
4. Une fois arrivé à destination $F - ant$ génère une $B - ant$, celle-ci hérite des informations de la première et emprunte le même chemin en sens inverse.
5. Les $F - ant$ sont effacées… On met a jour les $\tau_{ijd}(t)$ pour simuler l'évaporation.
6. La $B - ant$ met à jour les tables de routage. Bien entendu pour être efficace ce processus doit être lancé périodiquement de façon à pouvoir suivre les fluctuations temporelles de l'encombrement du réseau.

5. Métaheuristiques à population de solutions : (Algorithmes évolutionnaires)

Les algorithmes évolutionnaires (AE) appartiennent à la famille des algorithmes d'optimisation bioinspirés [13], Le principe de la sélection est fondé sur la lutte pour la vie, due à une population tendant à s'étendre mais disposent d'un espace et ressources finis. Il en résulte que les individus les plus adoptés tendant à survivre plus longtemps et à se reproduire plus aisément. Le principe d'adaptation d'une population d'individus aux conditions naturelles définis par environnement, inclut les lois de variations telles que le croisement et la mutation qui expliquent l'apparition des variations aux niveaux individuelles. Cette appariation explique bien le phénomène d'évolution et d'adaptation sans avoir besoin à une modification directe des individus. Le principe ainsi défini conduit une population à s'évoluer et s'adapter de génération en génération sans tendre vers un objectif dicté.

5.1 Algorithme génétique :

Les algorithmes génétiques (AG) sont peut-être les algorithmes évolutionnaires les plus populaires, appliqués dans de nombreux domaines tels que le traitement d'images, l'optimisation de réseaux, l'optimisation de fonction numériques difficiles, la planification, etc.

Principe général d'un algorithme génétique :

5.1.1 Codage des Variables :

L'étape clef dans un algorithme génétique est de définir et coder convenablement les variables d'un problème donnée. On retrouve différentes techniques de codages. Le codage est un processus de représentation des gènes. Le processus peut être effectué par utilisation des : bits, nombres, arbres, tableaux, listes ou tous autres objets. La littérature définit deux types de codage : binaire et réel.

2.1.1 Codage binaire :

C'est la représentation la plus fréquente, soit f une fonction à optimiser de paramètres x.La variable x représente un individu de la population, il est codé sous forme d'une chaîne n bits. Soit $x \in [x_{min}, x_{max}]$ avec $x \in R$ et x a un nombre de décimale noté d.

Dans une représentation binaire, la taille de l'individu n vérifie l'inéquation suivante :

$$|x_{min} - x_{max}| * 10^d \leq 2^n$$

Algorithme :

1.Initialisation :

- K : =nombre total de générations ;
- N : =taille de la population ;
- Pc : =Probabilité de croisement ;
- Pm : =probabilité de mutation ;
- POP : =population courante ;
- PINT : =population intermédiaire contenant N éléments chacun ;

2.Répéter

- Pour i=0 jusqu'à K
- Évaluer [POP]
- Sélection [PINT, POP]
- Croisement [PINT, Pc]
- Mutation [PINT, Pm]

3.Remplacement POP=PINT

- Evaluer [POP]

Solution meilleur=meilleur [POP]

5.2 Optimisation du loup gris :

5.2.1 Définition :

L'optimiseur de loup gris (GWO) est un algorithme de méta-heuristique basé sur la population qui simule la hiérarchie de leadership et le mécanisme de chasse des loups gris dans la nature, et il est proposé par Seyedali Mirjalili et al. En 2014. [59]

Les loups gris sont considérés comme des prédateurs au sommet, qui sont au sommet du string alimentaire, préfèrent vivre en groupe (meute), chaque groupe contient en moyenne 5 à 12 individus. [59]

Figure 17 : les étapes de la chasse des loups [62].

Le chef de la meute est représenté par Alpha (α). Alpha peut être un mâle ou une femelle. Il a le pouvoir de décider le lieu de repos, le moment de la chasse, et ainsi de suite. Les ordres du chef sont suivis par la meute. Intéressement, le chef n'est pas nécessairement le membre le plus fort de la meute, mais le meilleur en matière de gestion du groupe. Le deuxième niveau dans la hiérarchie, sont les Bêtas (β), ils sont des loups subalternes qui aident l'Alpha dans la prise de décision ou d'autres activités de meute. Un loup Bêta est probablement le meilleur candidat pour prendre la position de l'Alpha au cas où l'Alpha décède ou devient très vieux. Les loups les plus bas dans la hiérarchie sont les Omégas (ω). Ils doivent suivre les ordres de tous les autres loups dominants, et ils sont les derniers loups autorisés à manger. Si un loup n'appartient à aucune des catégories précédentes, il est appelé Delta (δ). Les loups Deltas dominent les Omégas et suivent les ordres des Alphas et des Bêtas.[60]

5.2.2 Hiérarchie sociale des loups gris :

Tous les individus du groupe ont une hiérarchie de dominance sociale très stricte,

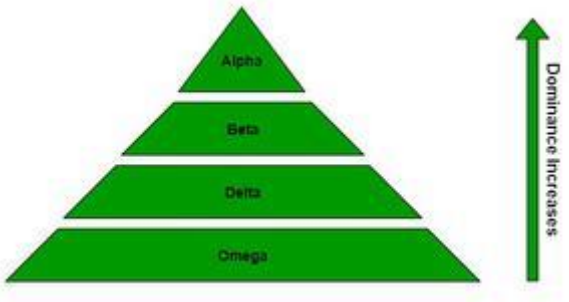


Figure 18 : Hiérarchie sociale des loups gris

1. Le loup alpha α est considéré comme le loup dominant de la meute et ses ordres doivent être suivis par les membres de la meute.
2. Les bêta β sont des loups subordonnés, qui aident l'alpha dans la prise de décision et sont considérés comme le meilleur candidat pour être l'alpha.
3. Les loups delta δ doivent se soumettre à l'alpha et à la bêta, mais ils dominent l'oméga
4. Les loups Omega ω sont considérés comme le bouc émissaire de la meute, sont les individus les moins importants de la meute et ne sont autorisés à manger qu'enfin. [59]

5.2.3 Modélisation mathématique :

Selon Muro et al., [62] tous les agents (loups) sont homogènes, et il n'y a aucune hiérarchie a priori. Les trois meilleures solutions sont respectivement Alpha, Bêta, et Delta. Les autres solutions sont supposées être des Omégas. La chasse est guidée par les trois meilleurs loups [61].

3.3 Encercler la proie

Les loups dans la nature encerclent leur proie, ce comportement a été modélisé comme suit [61] :

$$\vec{D} = | \vec{C} . \vec{X}_P(t) - \vec{X} |$$
$$\vec{X}(t+1) = \vec{X}_P(t) - \vec{A}. \vec{D}$$

Où $\vec{D}$ représente la distance entre le loup et la proie, t indique l'itération actuelle, $\vec{X}_P$ est la position de la proie, et $\vec{X}$ représente la position du loup. $\vec{A}$ et $\vec{C}$ sont des vecteurs de coefficients donnés par [61] :

$$\vec{A} = 2. \vec{a}. \overrightarrow{r_1} - \vec{a}$$
$$\vec{C} = 2. \overrightarrow{r_2}$$

Où les composantes de a sont linéairement diminuées de 2 à 0 au cours des itérations, et $\overrightarrow{r_1}$, et $\overrightarrow{r_2}$ sont des vecteurs aléatoires dans l'intervalle [0,1]. La valeur du vecteur $\vec{a}$ dans une itération quelconque t peut être obtenue par l'équation suivante [123] :

$$\vec{a} = 2 - t\left(\frac{2}{N_{iter}}\right)$$

Un loup peut se déplacer par les deux équations (71) et (72) vers différentes positions autour de la proie en ajustant la valeur des vecteurs $\vec{A}$ et $\vec{C}$. Par exemple, un loup $\vec{X}$ (X, Y) dans un espace de deux dimensions peut atteindre la position $(X^* - X, X^*)$ en mettant A = (1,0) et C = (1,1). La figure ci-dessous représente des exemples de déplacements possibles dans les deux plans de deux et de trois dimensions.

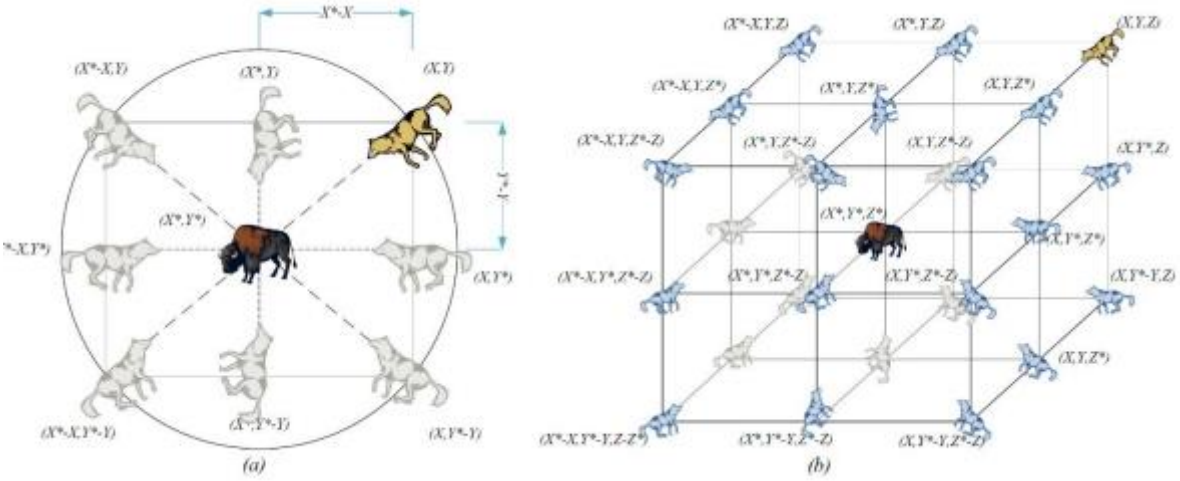


Figure 19 : 2D et 3D positions des loups et leurs prochaines positions possibles [61].

6. Conclusion :

Dans ce chapitre, nous avons présenté ce qu'est un problème d'optimisation, puis nous avons distingué les heuristiques des métaheuristiques. Ensuite, nous avons présenté les principales métaheuristiques à solution unique, puis à population de solutions.

Nous avons conclu que les domaines d'application de métaheuristique. Le chapitre suivant sera consacré pour la Conception et la modélisation d'un application pour un Problème d'emploi du temps des coures universitaires.

4

Conception et Modélisation

1. Introduction :

Ce chapitre sera consacré à la présentation et la conception de notre vision pour l'approche basée clustering pour le problème d'emploi du temps des cours universitaires.

2. GWO pour le Clustering [60] :

L'application de l'approche des loups gris pour le clustering peut être faite de différentes manières. La plus simple est de combiner cette méthode avec l'heuristique k-means. Cependant, elle va être très coûteuse en matière de complexité temporelle. L'idée la plus efficace est de générer des centres de clusters aléatoires $c_i(i = 1, \dots, k)$ (k est le nombre de clusters), et affecter les objets au cluster du centre le plus proche.

La population de loups PL est constituée de $|PL| = pl$ loups, $PL = (l_1, \dots, l_{pt})$. Chaque loup représente une solution, une solution possède k points, ces points représentent les centres des clusters. Une solution est alors représentée par un vecteur de k centres de clusters l_i
$(c_{i1}, .., c_{ik})$ où $i = 1, \dots, pl$. Comme chaque centre est un point de l'espace d'attributs, il est alors représenté par un vecteur de d valeurs (nombre d'attributs), un agent est alors représenté par un vecteur de $d * k$ valeurs :

$$l_1 = ((a_{i11}, \dots., a_{i1d}), \dots, (a_{ik1}, \dots, a_{i1d})$$

Où a_{ijm} est l'attribut numéro m du centre numéro j de l'i ème loup, et $j = 1, \dots, k$, $m = 1, \dots, d$.

L'algorithme commence par initialiser les agents de recherche par k points aléatoires de l'espace d'attributs, puis on ajoute chaque objet au cluster du centre le plus proche pour chaque loup, et on évalue les résultats obtenus par la fonction d'aptitude f. Les trois agents possédant les trois meilleures valeurs de la fonction d'aptitude f sont respectivement Alpha, Bêta, et Delta. La mise à jour des positions d'agents se fait par l'équation (78) Généralement, elle produit des valeurs réelles comme résultats, la valeur de $\vec{X}_1$ plus celle de $\vec{X}_2$ plus la valeur de $\vec{X}_3$le tout sur trois (calcul de la moyenne).

Puis on mit à jour les valeurs des vecteurs $\vec{a}$, $\vec{A}$ et $\vec{C}$ et on classifie les données pour chaque loup. On recalcule la valeur de la fonction objectif f pour chaque loup et on mit à jour la position d'Alpha, Bêta, et Delta. L'algorithme répète les instructions de la mise à jour des positions jusqu'au nombre maximum d'itérations.

2.1 Fonction d'aptitude :

La fonction utilisée pour évaluer le résultat du clustering est la somme des distances intra-cluster. La fonction de distance utilisée est celle de Gower.

2.2 Mise à jour de positions :

La mise à jour de positions est faite d'une manière plus intelligente. On a les trois positions $\vec{X}_1, \vec{X}_2$ et $\vec{X}_3$, l'opération simple est d'additionner les composantes du premier centre de chaque $\vec{X}_i$ et diviser le résultat par trois pour obtenir le premier centre, le deuxième centre et le troisième sont calculés de la même façon. Ce processus peut égarer les loups loin de la position de la proie, car la position des centres n'a pas de sens. C.-à-d. le clustering d'un loup l_i (C_1, C_2, C_3)donne le même résultat du clustering d'un autre loup l_i (C_3, C_2, C_1)possédant les mêmes centres mais dans un ordre différent. Donc la méthode la plus pertinente est de trouver de chaque centre de la position $\vec{X}_1$le centre le plus proche de lui que des autres positions $\vec{X}_2$, et $\vec{X}_3$.

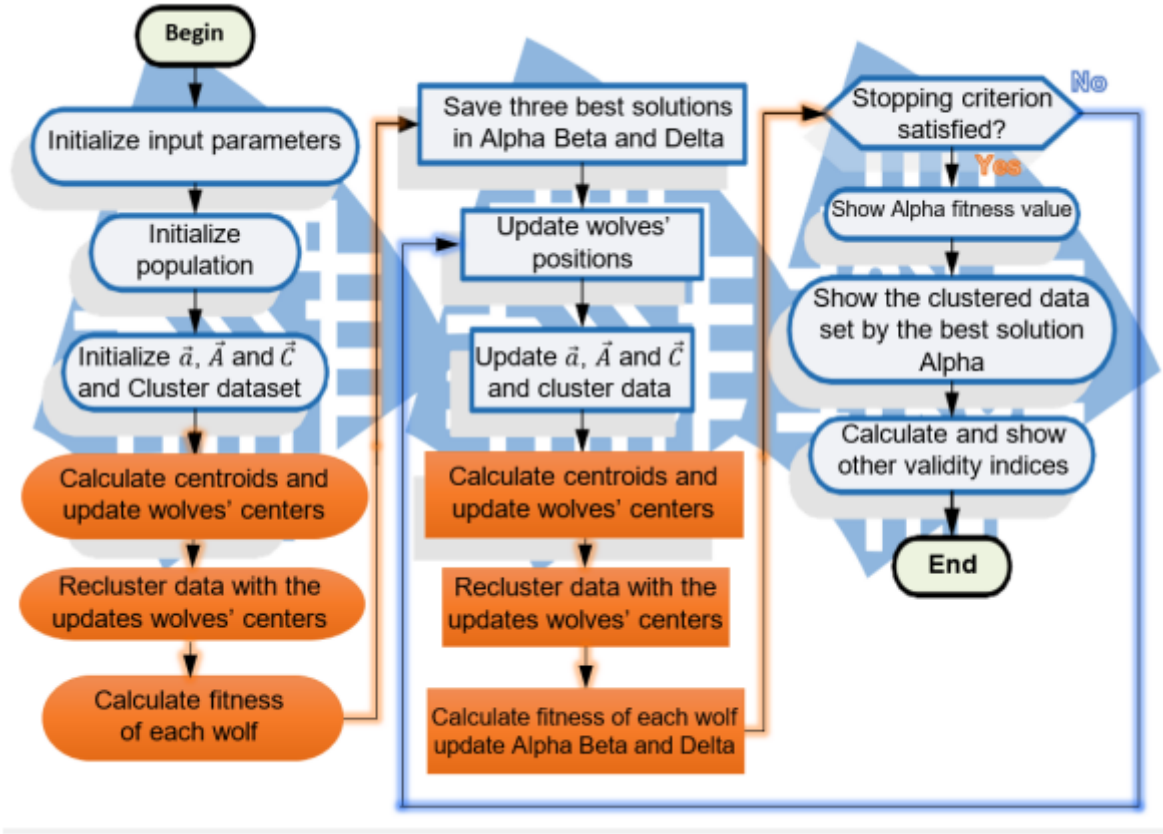


Figure 20 : Organigramme descriptif de la variante 3 de l'algorithme GWO pour le Clustering.

2.3 Algorithme GWO pour le clustering :

L'algorithme a peu de paramètres à définir, nombre de clusters, nombre de loups, et le nombre maximum d'itérations. L'algorithme est défini comme suit :

Algorithme GWO for clustering

Input :

Dataset D ;
Population size pl ;
Maximum number of itérations $Max\_number\_of\_iterations$;

Output:

Data clustered

Begin

1) Initialize the grey wolf population $X_i(i = 1,2,..n)$
2) Initialize $\vec{a}$, $\vec{A}$ and $\vec{C}$
3) Cluster data for each search agent
4) Calculate the fitness of each search agent
5) $\vec{X}_\alpha$, $\vec{X}_\beta$, $\vec{X}_\delta$ are the best three agents
6) **While** ($t < Max\_number\_of\_iterations$)
   for each search agent
   Update the position of the current search agent by equation
   end for
   Update $\vec{a}$, $\vec{A}$ and $\vec{C}$
   Cluster data for each search agent
   Calculate the fitness of all search agents
   Update $\vec{X}_\alpha$, $\vec{X}_\beta$ And $\vec{X}_\delta$
7) **end while**
8) Return data clustered by $\vec{X}_\alpha$

End

---

3. Le problème de coloriage de graphe et le problème TTP :

Le problème d'emploi du temps peut être vu comme un problème de coloriage de graphe ou on vise á grouper les séances d'enseignement dans des périodes de temps de tel façon qu'aucune séance ne soit en conflit avec les autres dans la même période.

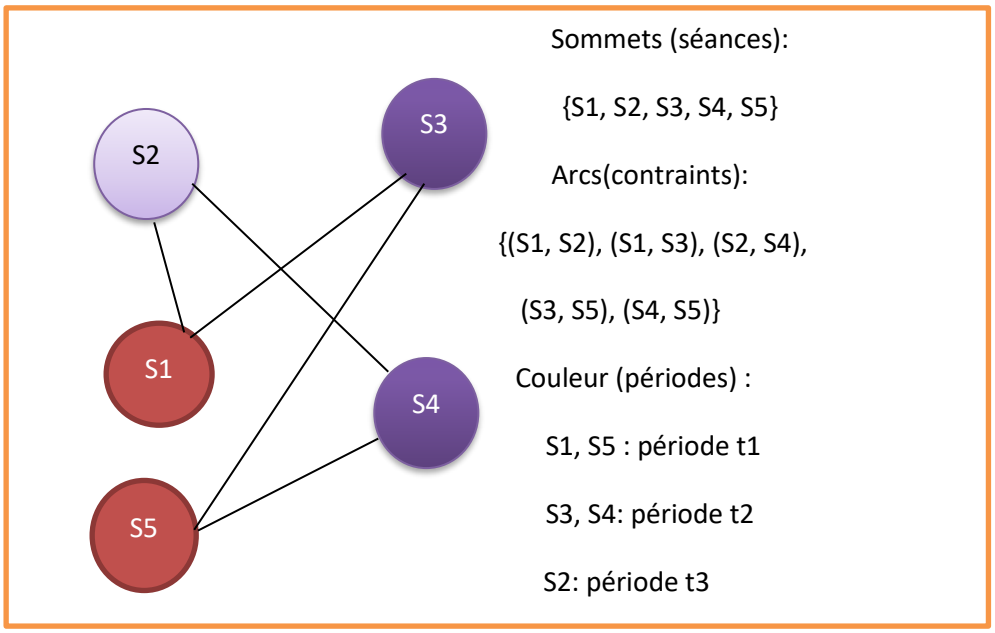


Figure 21 : Exemple explicatif pour la représentation d'emploi du temps par un graph

Sur la base de l'algorithme RND, les sessions sont attribuées à des groupes pouvant être menés simultanément, en fonction du nombre et du type de salles disponibles. Les formations pédagogiques associent des enseignants à des cours, des modules d'un TD ou d'un TP, et des groupes (ou sections) d'étudiants.

L'organigramme présenté dans la figure suivante montre l'algorithme qu'on a utilisé pour la création des solutions initiales.

Remarque 1 : on limite le nombre de solutions initialesà12 solutions par respect au nombre de loups dans une meute.

Remarque 2 : l'utilisation de RND Radom algorithme a pour but d'augmenter la diversification des solutions initiales.

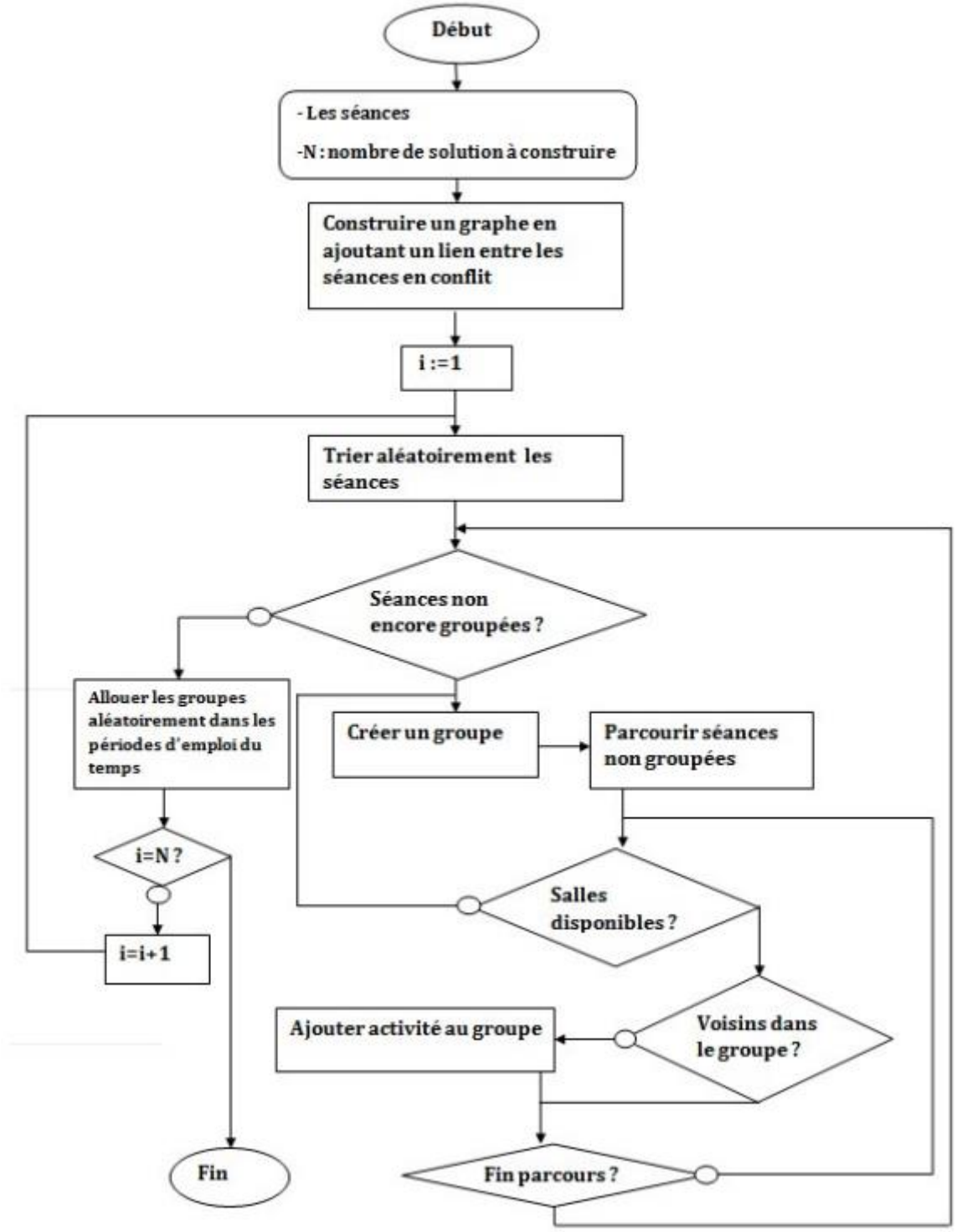


Figure 22 : Organigramme pour l'algorithme de création des solution initiale.

4. UML :

UML (Unified Modeling Language) a été conçu comme un langage de modélisation visuel polyvalent, sémantiquement et syntaxiquement riche. Il est destiné à être utilisé dans l'architecture, la conception et la mise en œuvre de systèmes logiciels complexes à travers sa structure et son comportement.

L'UML a des applications qui vont au-delà du développement logiciel, notamment pour les flux de processus dans l'industrie.

L'UML, dans son état actuel, définit une notation et un méta-modèle. La notation est le truc graphique qu'on voie dans les modèles (la syntaxe graphique du langage de modélisation). Par exemple, la notation de diagramme de classe définit comment les éléments et les concepts, tels que la classe, l'association et la multiplicité, sont représentés. Un méta-modèle est un diagramme de diagramme, généralement un diagramme de classe qui définit les concepts du langage [63].

4.1 Diagrammes UML :

UML est un langage de modélisation normalisé composé d'un ensemble intégré de diagrammes, la version 2.0 d'UML propose treize types de diagrammes officiels, Ces types de diagrammes sont répartis en deux grands groupes : diagrammes structurels et diagrammes comportementaux. On va présenter une modélisation avec deux diagrammes, diagramme cas d'utilisation et diagramme de classe :

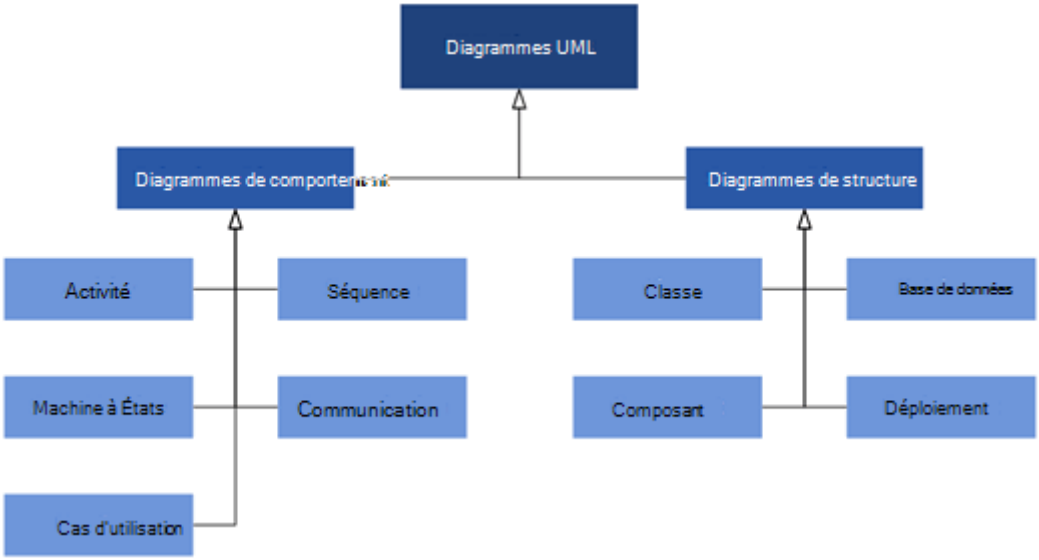


Figure 22 : Diagrammes d'UML.

4.1.1 Le diagramme des cas d'utilisation :

Il représente une fonction spécifique dans un système et est conçu pour illustrer comment différentes fonctions sont interconnectées et pour montrer leurs contrôleurs (ou acteurs) internes et externes.

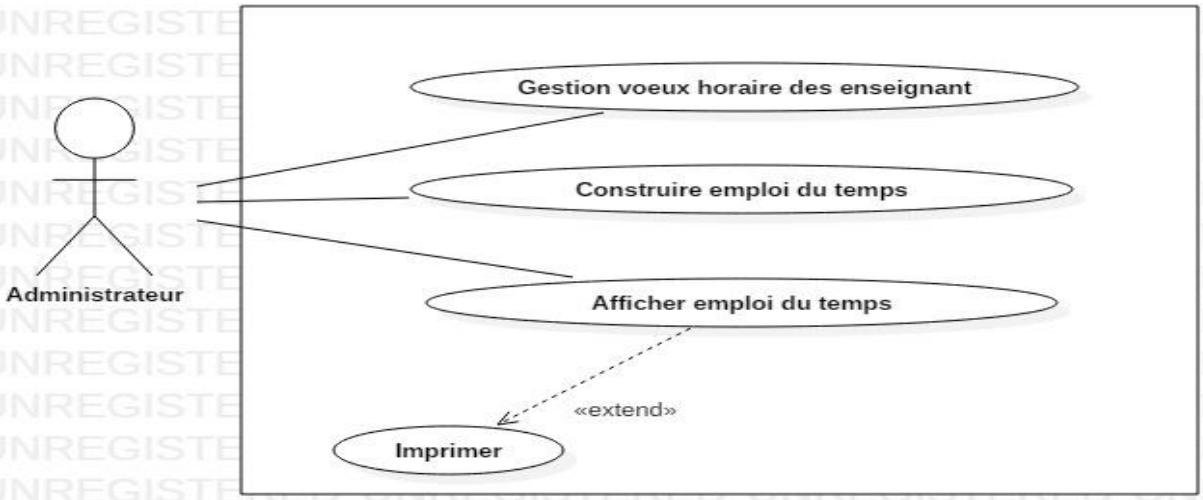


Figure 23 : Diagramme de cas d'utilisation.

4.1.2 Le diagramme de classe :

Diagramme UML le plus couramment utilisé et fondement de toute solution orientée objet. Classes d'un système, attributs et opérations, et relations entre chaque classe. Les classes sont regroupées pour créer des diagrammes de classe lors de la modélisation de systèmes de grande taille.[63]

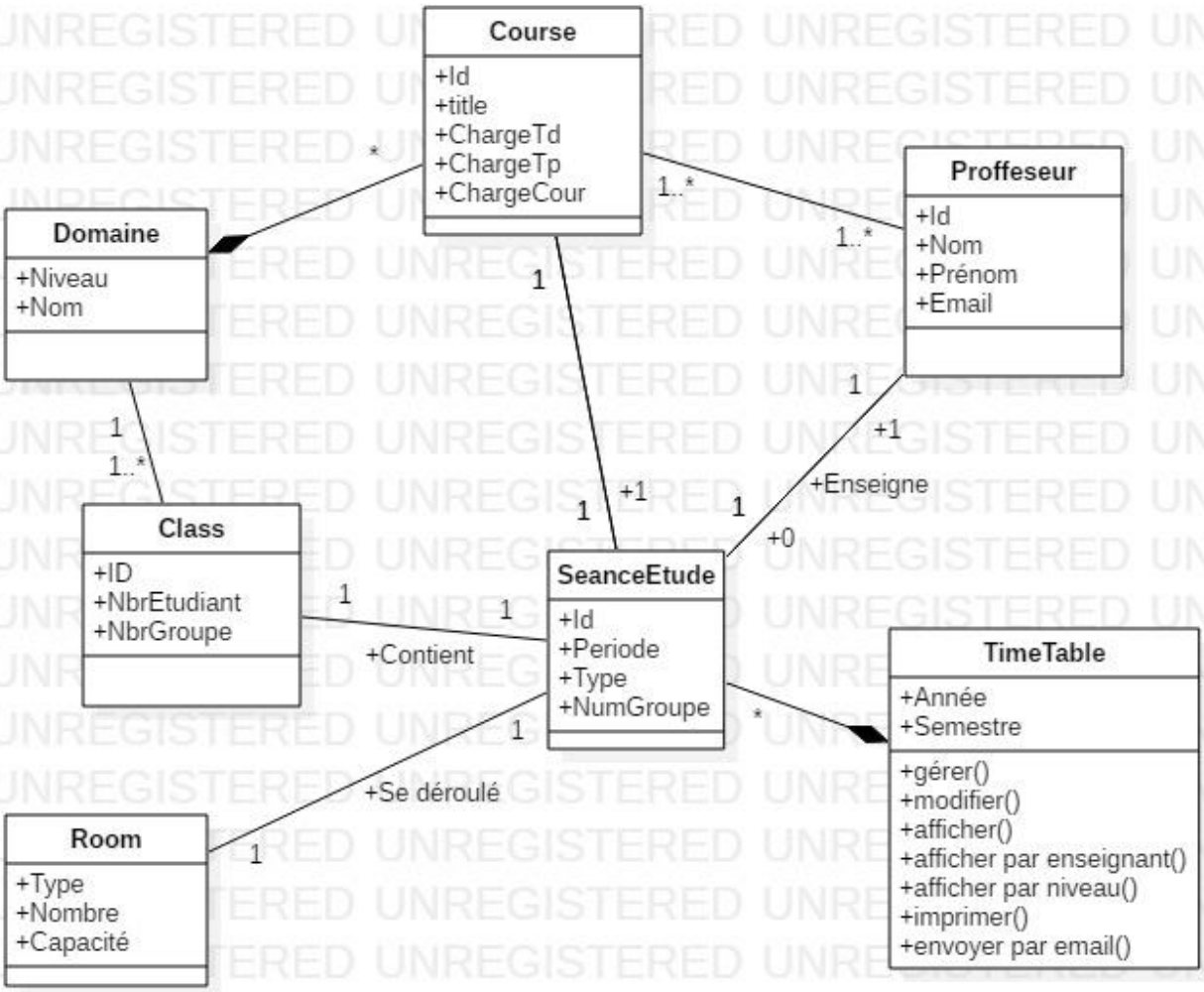


Figure 24 : Diagramme de classe.

5. Conclusion :

Dans ce chapitre nous avons présenté la modélisation de notre application l'approche basée clustering pour le problème d'emploi du temps des cours universitaires.

Dans le chapitre suivant, nous allons présenter l'implémentation et le résultat de notre application.

5

Implémentations et résultats

1. Introduction :

Dans ce chapitre, nous allons présenter l'implémentation du L'application pratiquement et présenter l'interface utilisée un emploi du temps des coures universitaire. Tout d'abord, nous présentons les des outils de développement matériels et logiciels. Nous finissons par présenter l'interface graphique utilisée dans l'application.

2. Environnement de travail :

2.1 Environnement matériel :

L'application a été développée et exécutée sur un ordinateur ASUS ayant les caractéristiques suivantes:

- Processeur: Intel ® Core ™ i3 CPU N3050 @ 1.60GHz
- RAM : 4 GO.

2.2 Environnement logiciel :

2.2.1 Système d'exploitation :

Le système d'exploitation utilisé pour la réalisation et l'exécution de notre application est :

- Windows-10 64-bit

2.2.2 Langages et bibliothèques de développement [64] :

Python :

C'est un langage de programmation qui peut s'utiliser dans de nombreux contextes et s'adapter à tout type d'utilisation grâce à des bibliothèques spécialisées. Il est cependant particulièrement utilisé comme langage de script pour automatiser des tâches simples mais fastidieuses, comme un script qui récupérerait la météo sur Internet ou qui s'intégrerait dans un logiciel de conception assistée par ordinateur afin d'automatiser certains enchaînements d'actions répétitives. On l'utilise également comme langage de développement de prototype lorsqu'on a besoin d'une application fonctionnelle avant de l'optimiser avec un langage de plus bas niveau. Il est particulièrement répandu dans le monde scientifique, et possède de nombreuses bibliothèques optimisées destinées au calcul numérique.[64]

Pandas :

C'est une bibliothèque écrite pour le langage de programmation Python permettant la manipulation et l'analyse des données. Elle propose en particulier des structures de données et des opérations de manipulation de tableaux numériques et de séries temporelles.

NumPy :

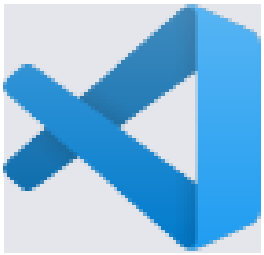
C ' est une bibliothèque pour langage de programmation Python, destinée à manipuler des matrices ou tableaux multidimensionnels ainsi que des fonctions mathématiques opérant sur ces tableaux.

Jupyter :

C ' est une application web utilisée pour programmer dans plus de 40 langages de programmation, dont Python, Julia, Ruby, R, ou encore Scala. C'est un projet communautaire dont l'objectif est de développer des logiciels libres, des formats ouverts et des services pour l'informatique interactive. Jupyter est une évolution du projet IPython. Jupyter permet de réaliser des calepins ou notebooks, c'est-à-dire des programmes contenant à la fois du texte en markdown et du code. Ces calepins sont utilisés en science des données pour explorer et analyser des données.

Visual Studio Code :

C’est un éditeur de code extensible développé par Microsoft pour Windows, Linux et macOS.



3. Présentation de l'application :

3.1 Fenêtre d'accueil :

Cette fenêtre est la fenêtre principale. Elle contient les boutons représentant les fonctionnalités principales de l'application :

Show TimeTabling,

Build TimeTabling,

GWO TimeTabling.

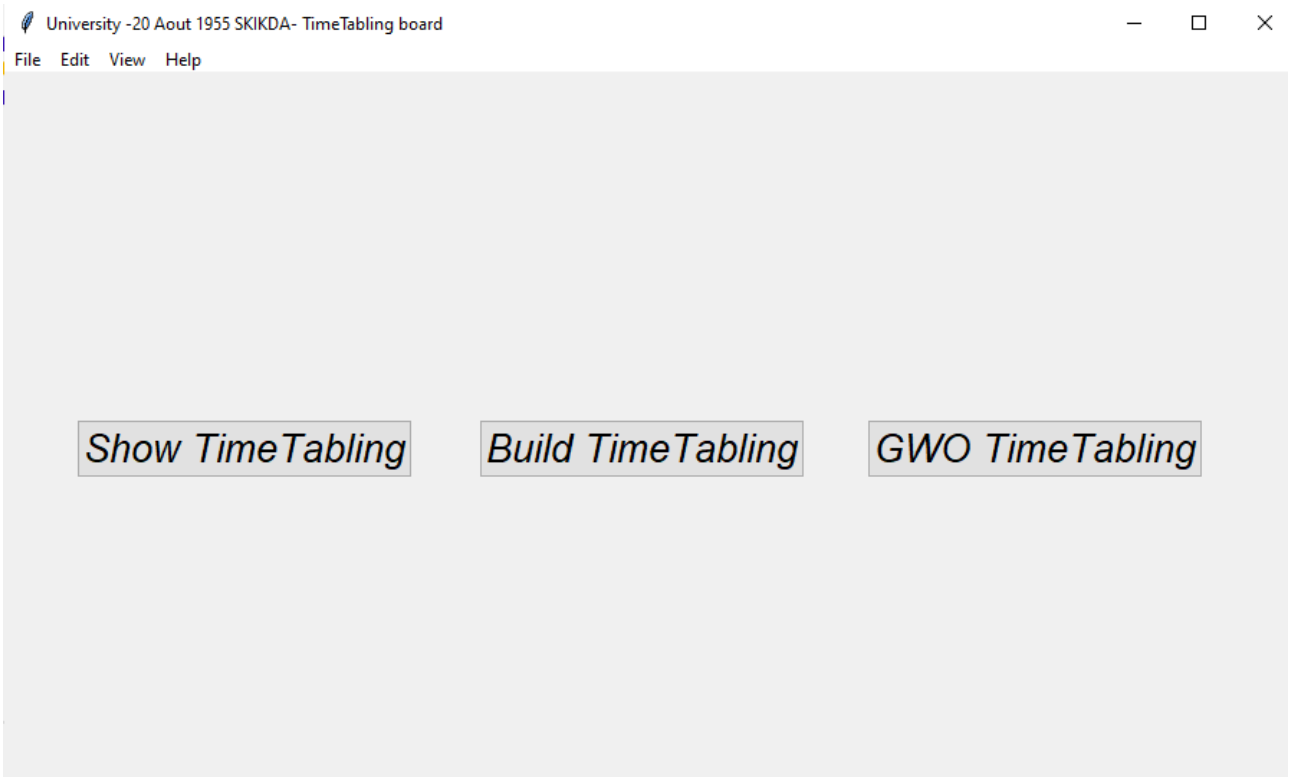


Figure 25 : l'interface de l'application.

✓ *Fenêtre de Build TimeTabling :*

Cette fenêtre permet de crée emploi du temps à partir le clustering :

1. Préparation La base de données :

Le tableau suivant figure les données du département de l'informatique de l'université 20 Août 1955 Skikda de l'année universitaire 2021-2022 de promo Master 2 Système Informatique :

	id	professoes	classe	course	type	groupe	nbr_etud
0	1	19	'm2si'	'Rech_Oper_App'	'Cour'	0	60
1	2	19	'm2si'	'Rech_Oper_App'	'TD'	1	30
2	3	19	m2si	Rech_Oper_App	TD	2	30
3	4	19	m2si	Rech_Oper_App	TP	1	30
4	5	19	m2si	Rech_Oper_App	TP	2	30
5	6	46	m2si	Anglais	Cour	0	60
6	7	46	m2si	Anglais	TD	1	30
7	8	46	m2si	Anglais	TD	2	30
8	9	47	m2si	M.R_Doc	Cour	0	60
9	10	47	m2si	M.R_Doc	TD	1	30
10	11	47	m2si	M.R_Doc	TD	2	30
11	12	25	m2si	Meth_Formel	Cour	0	60
12	13	25	m2si	Meth_Formel	TD	1	30
13	14	25	m2si	Meth_Formel	TD	2	30
14	15	22	m2si	Rec_Forme	Cour	0	60
15	16	22	m2si	Rec_Forme	TD	1	30
16	17	22	m2si	Rec_Forme	TD	2	30
17	18	22	m2si	Rec_Forme	TP	1	30
18	19	22	m2si	Rec_Forme	TP	2	30

2. Preprocessing la base de données :

Il permet de transformer les données en un format qui est traité plus facilement et plus efficacement dans l'exploration de données.

	professoes	classe	course	type	groupe	nbr_etud
0	19	0	0	0	0	60
1	19	0	0	1	1	30
2	19	1	8	3	2	30
3	19	1	8	4	1	30
4	19	1	8	4	2	30
5	46	1	1	2	0	60
6	46	1	1	3	1	30
7	46	1	1	3	2	30
8	47	1	4	2	0	60
9	47	1	4	3	1	30
10	47	1	4	3	2	30
11	25	1	5	2	0	60
12	25	1	5	3	1	30
13	25	1	5	3	2	30
14	22	1	7	2	0	60
15	22	1	7	3	1	30
16	22	1	7	3	2	30
17	22	1	7	4	1	30

3. Clustering Data :

```
[[2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
2 2 2 2 2 2 2 2 2 2 2\n 2 2 2 2 2 2 2 2 2 2 2 2 2 2 0 0 0 1 0 1 0 1 0 1 1 1 1 0 1 0 1 1 0 1 0 1\n 0 0 0 0 0 0 0 1 1 1 1 0 1 0
0 0 1 1 1 0 1 1 1 1 1 0 1 1 0 0 0 0 0 0 1 0\n 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0\n 0 0 0
0 0 0]]']
```

- ✓ *Fenêtre Show TimeTabling :*

La figure illustre la fenêtre qui affiche l'emploi du temps. On peut l'afficher par un groupe de promo en utilise une matière pour afficher :

	Dimanche	**Lundi**	**Mardi**	**Mercredi**	**Jeudi**
08:00-09:30			0		
09:30-11:00					1
11:00-12:30					
12:30-02:00					
02:00-03:30					
03:30-05:00			2		

✓ *Fenêtre GWO TimeTabling :*

La figure illustre les effets de l'algorithme GWO Clustering :

```
GWO is optimizing  "F1"
['At iteration 0 the best fitness is 166.0094722629275']
['At iteration 1 the best fitness is 151.39337325097202']
['At iteration 2 the best fitness is 151.39337325097202']
['At iteration 3 the best fitness is 151.39337325097202']
['At iteration 4 the best fitness is 151.39337325097202']
['At iteration 5 the best fitness is 151.39337325097202']
['At iteration 6 the best fitness is 151.39337325097202']
['At iteration 7 the best fitness is 151.39337325097202']
['At iteration 8 the best fitness is 151.39337325097202']
['At iteration 9 the best fitness is 151.39337325097202']
['At iteration 10 the best fitness is 151.39337325097202']
['At iteration 11 the best fitness is 151.39337325097202']
['At iteration 12 the best fitness is 148.52291373101275']
['At iteration 13 the best fitness is 148.52291373101275']
['At iteration 14 the best fitness is 148.52291373101275']
['At iteration 15 the best fitness is 148.52291373101275']
['At iteration 16 the best fitness is 148.52291373101275']
['At iteration 17 the best fitness is 148.52291373101275']
['At iteration 18 the best fitness is 148.52291373101275']
['At iteration 19 the best fitness is 148.52291373101275']
['At iteration 20 the best fitness is 148.52291373101275']
['At iteration 21 the best fitness is 148.52291373101275']
['At iteration 22 the best fitness is 148.52291373101275']
['At iteration 23 the best fitness is 148.52291373101275']
['At iteration 24 the best fitness is 148.52291373101275']
['At iteration 25 the best fitness is 148.52291373101275']
['At iteration 26 the best fitness is 148.52291373101275']
['At iteration 27 the best fitness is 140.89416620938297']
['At iteration 28 the best fitness is 140.89416620938297']
['At iteration 29 the best fitness is 140.89416620938297']
```

```
['At iteration 30 the best fitness is 140.89416620938297']
['At iteration 31 the best fitness is 140.89416620938297']
['At iteration 32 the best fitness is 140.89416620938297']
['At iteration 33 the best fitness is 140.89416620938297']
['At iteration 34 the best fitness is 140.89416620938297']
['At iteration 35 the best fitness is 140.89416620938297']
['At iteration 36 the best fitness is 140.89416620938297']
['At iteration 37 the best fitness is 140.89416620938297']
['At iteration 38 the best fitness is 140.89416620938297']
['At iteration 39 the best fitness is 140.89416620938297']
['At iteration 40 the best fitness is 140.89416620938297']
['At iteration 41 the best fitness is 140.89416620938297']
['At iteration 42 the best fitness is 140.89416620938297']
['At iteration 43 the best fitness is 140.89416620938297']
['At iteration 44 the best fitness is 140.89416620938297']
['At iteration 45 the best fitness is 140.89416620938297']
['At iteration 46 the best fitness is 140.89416620938297']
['At iteration 47 the best fitness is 140.89416620938297']
['At iteration 48 the best fitness is 140.89416620938297']
['At iteration 49 the best fitness is 140.89416620938297']
['At iteration 50 the best fitness is 140.89416620938297']
['At iteration 51 the best fitness is 140.89416620938297']
['At iteration 52 the best fitness is 140.89416620938297']
['At iteration 53 the best fitness is 140.89416620938297']
['At iteration 54 the best fitness is 140.89416620938297']
['At iteration 55 the best fitness is 140.89416620938297']
['At iteration 56 the best fitness is 140.89416620938297']
['At iteration 57 the best fitness is 140.89416620938297']
['At iteration 58 the best fitness is 140.89416620938297']
['At iteration 59 the best fitness is 140.89416620938297']
['At iteration 60 the best fitness is 140.89416620938297']
```

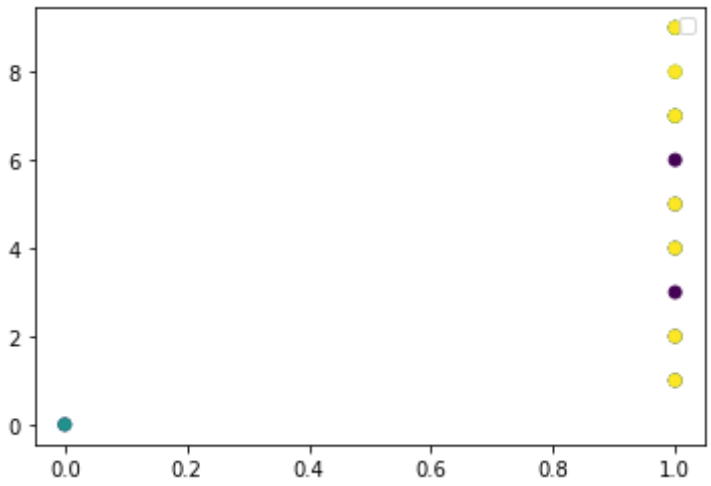
```
['At iteration 61 the best fitness is 140.89416620938297']
['At iteration 62 the best fitness is 140.89416620938297']
['At iteration 63 the best fitness is 140.89416620938297']
['At iteration 64 the best fitness is 140.89416620938297']
['At iteration 65 the best fitness is 140.89416620938297']
['At iteration 66 the best fitness is 140.89416620938297']
['At iteration 67 the best fitness is 140.89416620938297']
['At iteration 68 the best fitness is 140.89416620938297']
['At iteration 69 the best fitness is 140.89416620938297']
['At iteration 70 the best fitness is 140.89416620938297']
['At iteration 71 the best fitness is 140.89416620938297']
['At iteration 72 the best fitness is 140.89416620938297']
['At iteration 73 the best fitness is 140.89416620938297']
['At iteration 74 the best fitness is 140.89416620938297']
['At iteration 75 the best fitness is 140.89416620938297']
['At iteration 76 the best fitness is 140.89416620938297']
['At iteration 77 the best fitness is 140.89416620938297']
['At iteration 78 the best fitness is 140.89416620938297']
['At iteration 79 the best fitness is 140.89416620938297']
['At iteration 80 the best fitness is 140.89416620938297']
['At iteration 81 the best fitness is 140.89416620938297']
['At iteration 82 the best fitness is 140.89416620938297']
['At iteration 83 the best fitness is 140.89416620938297']
['At iteration 84 the best fitness is 140.89416620938297']
['At iteration 85 the best fitness is 140.89416620938297']
['At iteration 86 the best fitness is 140.89416620938297']
['At iteration 87 the best fitness is 140.89416620938297']
['At iteration 88 the best fitness is 140.89416620938297']
['At iteration 89 the best fitness is 140.89416620938297']
['At iteration 90 the best fitness is 140.89416620938297']

['At iteration 91 the best fitness is 140.89416620938297']
['At iteration 92 the best fitness is 140.89416620938297']
['At iteration 93 the best fitness is 140.89416620938297']
['At iteration 94 the best fitness is 140.89416620938297']
['At iteration 95 the best fitness is 140.89416620938297']
['At iteration 96 the best fitness is 140.89416620938297']
['At iteration 97 the best fitness is 140.89416620938297']
['At iteration 98 the best fitness is 140.89416620938297']
['At iteration 99 the best fitness is 140.89416620938297']
```

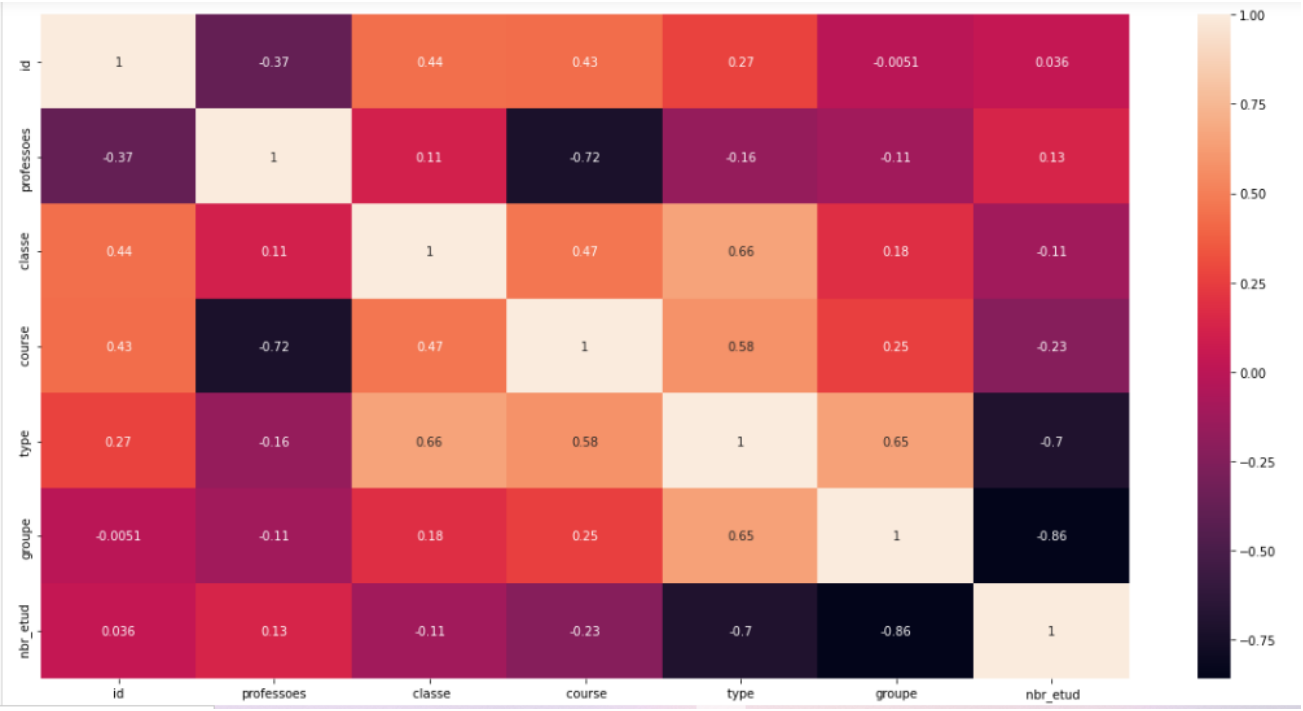
The best fitness fixé après 26 itérations :

```
['The total intra-cluster cost: 140.89416620938297
```

✓ Représentation graphique des données :



✓ La corrélation des données :



4. Conclusion :

Nous avons présenté dans ce chapitre l'implémentation de notre application et les resultats en montrant les outils de développement, avec la présentation détaillée des interfaces.
Pour récapituler, les résultats précédents vérifient l'approche de clustering en utilise L'algorithme de grey wolf optimizer et ses effets a la base de données d'une probleme d'emploi du temps des coures universitaire.

Conclusion Générale

Le problème d'emploi du temps est NP-complet et difficile à résoudre. La difficulté est de considérer beaucoup d'informations en plus de diverses contraintes, notamment en milieu scolaire.

Notre stratégie de travail est d'allier connaissances théoriques et pratiques afin d'élaborer un emploi du temps répondant aux plus grands besoins en termes de gestion du temps, de matériel et de ressources humaines. Nous avons traité de la génération distribuée des emplois du temps scolaire en prennent ce problème pour un problème de clustering.

Il y a beaucoup d'issues de recherches à explorer au sujet de problème de l'emploi du temps dont on peut citer :

- ✓ Plus de travail doit être fait pour développer de meilleurs algorithmes, en particulier pour plusieurs variables locales. Dans ce cas, le professeur se concentrera sur plusieurs leçons durant la semaine en fonction de son temps libre.
- ✓ Nous ne pouvons pas compter qu'un algorithme simple peut résoudre d'une manière efficace tous les types de ce problème. On espère étendre notre travail dans les futures recherches en réalisant les perspectives suivantes :
- ✓ Inclure d'autres contraintes de performance (respecter au mieux la préférence des enseignants pendant l'affection des séances, minimiser les erreurs commises lors de la programmation…).
- ✓ Modifier la méthode d'affectation des ressources de manière à programmer pour chaque séance toutes les tâches pouvant se dérouler en parallèle.
- ✓ Il est important de noter que les systèmes multi agents sont des outils d'optimisation innés très puissants surtout dans les domaines de la résolution distribuée de problème.

Références Bibliographie

[1] Khatir Nadjia, « Les technique de clustering dédie aux donnée multimédia », Thèse de Doctorat à L'université de AHMED BEN BELLA ORAN, 2019.

[2] Guillaume Cleuziou, Une méthode de classification non-supervisée pour l'apprentissage de règles et la recherche d'information », Thèse de Doctorat à L'université d'Orléans FRANCE, 2004.

[3] Halkidi, Maria, Batistakis, Yannis, and Vazirgiannis, Michalis « Cluster validity methods: part 1» ACM Sigmod Record, 31(2):40-45, 2002.

[4] Eisson (G.). La similarité une notion symbolique/numérique. Apprentissage symbolique-numérique (tome 2), 2000.

[5] J. HAN & M. KAMBER, Data mining : Concepts and techniques, Management Systems (The Morgan Kaufmann Series in Data Management Systems),MORGAN KAUFFMAN, ISBN 1-55860-901-6, 2006.

[6] Hamidouche Saddek et Idjeraoui Tayeb « Clustering : Approche par la théorie des jeux» Mémoire de Master à L'université de MIRA ABDERRAHAME BEJAIA, 2013

[7] E. HART & D. G. STORK P.DUDA, Pattern classification ; Wiley New York, 2001

[8] L. ROKACH & O. MAIMON, Clustering Methods, Department of Industrial Engineering , Tel-Aviv University, 2002

[9] J. B. MacQueen (1967). « Some Methods for classification and Analysis of Multivariate Observations [archive] » dans Proceedings of 5th Berkeley Symposium on Mathematical Statistics and Probability 1: 281–297 p.. Consulté le 7 avril 2009.

[10] Hae-Sang Park, Chi-Hyuck Jun « A simple and fast algorithm for K-medoids clustering »Department of Industrial and Management Engineering, POSTECH, San 31 Hyoja-dong, Pohang 790-784, South Korea, 2009

[11] Lubna Sulaiman Al-Henaki « A GENETIC-FROG LEAPING ALGORITHM FOR TEXT

DOCUMENT CLUSTERING »Department of Computer Science, King Saud University, 2018/2019

[12] Yu-Zhong Chen1 and Ying-Cheng La «Sparse dynamical Boltzmann machine for reconstructing complex networks with binary dynamics»1School of Electrical, Computer and Energy Engineering, Arizona State University, Tempe, Arizona 85287, USA 2Department of Physics, Arizona State University, Tempe, Arizona 85287, USA
(Received 7 April 2017; revised manuscript received 9 August 2017; published 28 March 2018)

[13] Alireza Entezami, Hassan Sarmadi ,Behzad Saeedi Razavi An innovative hybrid strategy for structural health monitoring by modal fexibility and clustering methods Journal of Civil Structural Health Monitoring Received: 7 December 2019 / Revised: 25 June 2020 / Accepted: 1 July 2020

[14] Asif Afzal a,* , Zahid Ansari b , Saad Alshahrani c , Arun K. Raj d , Mohamed Saheer Kuruniyan e , C. Ahamed Saleel c,* , Kottakkaran Sooppy Nisarf, Clustering of COVID-19 data for knowledge discovery using c-means and fuzzy c-means, Results in Physics, Received 11 July 2021; Received in revised form 30 July 2021; Accepted 31 July 2021.

[15] A. JAIN, M. MURTY et P. FLYNN, Data Clustering: A Review, The Ohio State University, 1999.

[16] A. K. Jain et R. C. Dubes, Algorithm for Clustering Data, 1988.

[17] R. Besançon, A. L. Daquo ; Clustering de documents dans des collections hétérogènes ; Document numérique ; Vol 18 ; 2015 ; pages 81 - 100.

[18] J. Han, M. Kamber ; Data mining : Concepts and techniques, Management Systems (The Morgan Kaufmann Series in Data Management Systems) ; MORGAN KAUFFMAN ; ISBN 1-55860-901-6 ; 2006.

[19] A. Yahi ; Clustering des données de puces à ADN ; Thèse de master ; Université M. BOUDIAF ; M'SILA ; 2019.

[20] N. Beck ; Application de méthodes de clustering traditionnels et extension au cadre multicritère ; thèse de magister ; Université libre de Bruxelles ; 2006.

[21]J. Han, M. Kamber ; Data mining : Concepts and techniques, Management Systems (The Morgan Kaufmann Series in Data Management Systems) ; MORGAN KAUFFMAN ; ISBN 1-55860-901-6 ; 2006.

[22] M. Koudri ; Segmenter via Expectation Maximization ; Thèse de master ; université de Tlemcen ; pages12 - 45.

[23] K. Zeitouni ; Techniques de data mining ; cours de Master Profssionnel ; ASS ; Edition 2009.

[24] P. RAI & S. SINGH, A survey of clustering techniques, International Journal of Computer Applications (0975 - 8887) Volume 7- No.12, October 2010.

[25] P. Rai, S. Singh ; A survey of clustering techniques ; International Journal of Computer

Applications (0975 - 8887) Volume 7- No.12 ; October 2010.

[26] R. Pradeep et S. Shubha, «A survey of clustering techniques, » International Journal of Computer Applications, 2010.

[27] J. Han, M. Kambe et J. Pei, «Cluster Analysis: Basic concepts and Methods,» chez Data Mining (Third Edition), Morgan Kaufmann, 2012, pp. 443 - 495

[28] X. Xu, M. Ester, H. P. Kriegel, J. Sander ; A distribution-based clustering algorithm for

mining in large spatial databases ; In 14th International Conference on Data Engineering

; USA ; 1998 ; pages 324 - 331

[29] P. Rai, S. Singh ; A survey of clu

stering techniques ; International Journal of Computer Applications (0975 - 8887) Volume 7- No.12 ; October 2010

[30] P. H. A. Sneath, R. R. Sokal ; Numerical taxonomy - the principles and practice of numerical classiffication ; Technical report ; San Francisco ; 1973.

[31] R. Yogita, R.Harish ; A Study of Hierarchical Clustering Algorithm ; International Journal of Information and Computation Technology ; 2013.

[32] A.Blum, T.Mitchell ; combining labeled and unlabeled data with co-training ; COLT : Proceedings of the Workshop on Computational Learning Theory ; Morgan Kaufmann ; 1998 ; pages 92-100.

[33] S. TUFFÉRY ; Data mining et statistique décisionnelle : l'intelligence dans les bases decisionnelle ; Edition Technip ; Paris ; 2005 ; pages 133 - 146.

[34] P. Berkhin, «Survey of clustering data mining techniques,» chez Grouping Multidimensional Data, pp. 25-71.

[35] P. RAI & S. SINGH, A survey of clustering techniques, International Journal of Computer Applications (0975 - 8887) Volume 7- No.12, October 2010

[36] A. Yahi ; Clustering des données de puces à ADN ; Thèse de master ; Université M.

BOUDIAF ; M'SILA ; 2019.

[37] D. Renaudie ; méthodes d'apprentissage automatique pour la modélisation de l'élève en

algèbre ; thèse de doctorat ; institut national Polytechnique de grenoble ; 2005.

[38] Y. Batistakis, M.vazirgiannis, M. Halkidi ; Clustering validity checking methods ;

[39] Driver and Kroeber (1932). "Quantitative Expression of Cultural Relationships". University of California Publications in American Archaeology and Ethnology. Berkeley, CA: University of California Press. Quantitative Expression of Cultural Relationships: 211–256.

[40] Zubin, Joseph (1938). "A technique for measuring like-mindedness". The Journal of Abnormal and Social Psychology. 33 (4): 508–516. doi:10.1037/h0055441. ISSN 0096-851X.

[41] Tryon, Robert C. (1939). Cluster Analysis: Correlation Profile and Orthometric (factor) Analysis for the Isolation of Unities in Mind and Personality. Edwards Brothers.

[42] Cattell, R. B. (1943). "The description of personality: Basic traits resolved into clusters". Journal of Abnormal and Social Psychology. 38 (4): 476–506. doi:10.1037/h0054116.

[43] F.W. Wilmink, M.A., M.D., and H.T. Qytterschaut, Ph. D.

Lab. of Anatomy and Embryology, University of Groningen,

The Netherlands CLUSTER ANALYSIS, HISTORY, THEORY AND APPLICATIONS

[44] Hamidani Hicham,Chikha Belgacem Abderrazzak Conception et développement d'une application d'optimisation basée sur les systèmes multi-agents cas d'étude « le problème de l'emploi du temps » Thèse de master ; Université Kasdi Merbah Ouargla , 2013

[45] ABBES FAICEL et OMRI ABD EL OUAHEBE et COULIBALY SOULEYMAN La mise au point d'un système de génération automatique de l'emploi du temps basé sur les Systèmes Multi-agents. université 08 mai 1945 guelma (2006).

[46] GOTHA, Les problèmes d'ordonnancement, Revue française d'automatique, d'informatique et de recherche opérationnelle. Recherche opérationnelle, tome 27, n°1 p. 77-150. 1993

[47] Houssem Eddine Nouri, Olfa belkahla (résolution multi agents du problème d'emploi du temps universitaire) Edition universitaire européennes Saarbrücken, Deutschland 2014

[48] Burke, E. K. and S. Petrovic (Accepted for publication in 2002). Recent research directions in automated timetabling, European Journal of Operational Research. 2002

[49] Thierry Moyaux, Brahim Chaib-draa et Sophie D'Amours. Satisfaction distribuée de contraintes et son application à la génération d'un emploi du temps d'employés. 2003.

[50]Optimisation par colonies de fourmis COSTANZO Andrea LUONG Thé Van MARILL Guillaume 19 mai 2006

[51] Glover, F., 1986. Future paths for integer programming and links to artificial intelligence. Computers and Operations Research 13, 533–549.

[52] Dréo, J., Pétrowski, A., Siarry, P., Taillard, E., 2003. Métaheuristiques pour l'optimisation difficile. Eyrolles.

[53] Eglese, R., 1990. Simulated annealing : a tool for operational research. European Journal of Operational Research 46 (3), 271–281.

[54] Siarry, P., Berthiau, G., Durdin, F., Haussy, J., 1997. Enhanced simulated annealing for globally minimizing functions of many-continuous variables. ACM Transactions on Mathematical Software (TOMS) 23 (2), 209–228.

[55] Taillard, E. D., 1998. La programmation à memoire adaptative et les algorithmes pseudo-gloutons : nouvelles perspectives pour les meta-heuristiques. Tech. rep., Istituto Dalle Molle Di Studi Sull Intelligenza Artificiale.

[56] [Dréo 04] Un chapitre dans son livre qui parle de la méta-heuristique ACO, d'une manière simplifiée

[57] Eberhart, R., Kennedy, J., 1995. A new optimizer using particle swarm theory. In : Proceedings of the sixth International Symposium on Micro Machine and Human Science. pp. 39–43.

[58] J.Dréo, A.Pétrowski, P.Siarry, E.Taillard : Métaheuristiques pour l'optimisation difficile. Edition Eyrolles, 2003.

[59] https://fr.acervolima.com/optimisation-du-loup-gris-introduction/

[60] Zebiri Ibrahim« Optimisation par loups gris adaptée pour le Clustering de données », Thème de Master à L'université de 20 AOUT 1955 SKIKDA, 2020.

[61] S. Mirjalili, S. M. Mirjalili et A. Lewis, «Grey Wolf Optimizer,» 2014.

[62] C. Muroa, R. Escobedoa, b. L. Spectorc et R. P. Coppingerc, «Wolf-pack (Canis lupus) hunting strategies emerge from simple rules in computational simulations,» 2011.

[63] https://www.lucidchart.com/pages/fr/langage-uml

[64] Wikipedia