REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

**Ecole Nationale Polytechnique**

المدرسة الوطنية المتعددة التقنيات
Ecole Nationale Polytechnique

**Département d'ELECTROTECHNIQUE**
Laboratoire de Recherche en Electrotechnique

Mémoire de projet fin d'études pour l'obtention du diplôme d'
**Ingénieur d'Etat en Electrotechnique**

Intitulé

# Étude du problème de l'Unit Commitment (Engagement des turbines) dans les réseaux électriques

Présenté par
**Younes SENIANI et Rania SAOUDI**

Sous la direction de **Pr. A. Hellal**
Soutenu publiquement le 19 juin 2018

**Membres du Jury**

| | |
|---|---|
| Président : | A. Mekhaldi, Professeur à l'ENP |
| Directeur : | A. Hellal, Professeur à l'ENP |
| Directeur : | L. Nezli, Professeur à l'ENP |

ENP 2018
Laboratoire de Recherche en Electrotechnique (LRE) - Ecole Nationale Polytechnique (ENP)
10, Avenue des Frères Oudek, Hassen Badi, BP. 182, 16200 El Harrach, Alger, Algérie.

# Dédicace

*On dédie ce travail à nos chers parents, nos amis, nos collègues et à tous ceux qui nous ont aidés à réaliser ce mémoire.*

# Remerciements

Nous voudrions présenter nos remerciements à notre encadreur Mr. Hellal Abdelhafid, Professeur à l'École Nationale Polytechnique . Nous voudrions également lui témoigner notre gratitude pour sa patience et son soutien qui nous a été précieux afin de mener notre travail à bon port.

Nos vifs remerciements vont également aux membres du jury :

Mr. Mekhaldi Abdelouaheb, Professeur à l'École Nationale Polytechnique,

Mr. Nazli Lezhari, Professeur à l'École Nationale Polytechnique,

pour l'intérêt qu'ils ont porté à notre recherche en acceptant d'examiner notre travail et de l'enrichir par leurs propositions.

Enfin, nous tenons à exprimer nos sincères remerciements à tous les professeurs qui nous ont enseigné et qui par leurs compétences nous ont soutenu dans la poursuite de nos études.

## ملخص

العمل المقدم يهدف إلى دراسة مشكلة برمجة وحدات الكهرباء التي تتلخص في إطفاء أو تشغيل محطات التوليد الكهربائي في شبكة الكهرباء بالاعتماد على جدولة مثالية و اقتصادية بهدف تخفيض تكاليف إنتاج الطاقة الكهربائية.على مر السنين ,تم استعمال و تطوير العديد من نظم الاستخدام الأمثل في حل هذه المشكلة.من بينها : الطرق الحتمية والطرق المتطورة (الذكاء الاصطناعي). بهدف توضيح أداء الطرق المختلفة المذكورة سابقا تم إجراء محاكاة مع عرض النتائج.



**كلمات الدالة :**برمجة وحدات الكهرباء ,الجدولة المثالية ,قائمة الأوليات ,البرمجة الديناميكية ,الخوارزميات الجينية ,طريقة أسراب الطيور ,طريقة تمثيل التخمير.



# Abstract

The present work deals with the Unit commitment problem which consists of turning on/off production units based on an optimal scheduling in the most economic way in a power system. Over the years, many methods have been used to minimize the total production cost. Among these methods, we'll focus solely on stochastic methods and metaheuristics. In order to have a better understanding of their performance, simulations have been done along with a display of results.



**Keywords** : Power systems, Unit commitment problem, Dynamic programming, Priority list, Particle swarm optimization, Simulated Annealing, Genetic Algorithm, Economic dispatch.



# Résumé

Le présent travail porte sur la résolution du problème d'engagement des turbines. Ce problème, qui consiste à allumer ou éteindre des centrales électriques dans un réseau électrique, a pour objectif de minimiser le coût de production total. Au fil des années, plusieurs méthodes d'optimisation ont été utilisées pour la résolution de ce problème. Parmi ces méthodes, on distingue les méthodes déterministes ainsi que les méthodes avancées. Afin de vérifier leurs performances, des simulations avec présentation des résultats ont été faites.



**Mots-clés** : Engagement des turbines, Programmation dynamique, Algorithme génétique, Optimisation par essaims particulaires, Liste de priorité, Recuit simulé, Répartition économique.

# Table des matières

# Liste des tableaux

# Liste des figures

# Liste de notations

$U_i(t)$ : État de l'unité i à un instant t (en fonctionnement si égale à 1, en arrêt si égale à 0).

$F_T$ : Coût total de production.

$P_i(t)$ : Puissance produite par l'unité à l'instant t.

$P_D(t)$ : Puissance demandée à l'instant t.

$P_R(t)$ : Réserve tournante à l'instant t.

$P_{\max i}$ : Puissance maximale de l'unité i

$P_{\min i}$ : Puissance minimale de l'unité i

$N$ : Nombre total d'heures.

$N_t$ : Nombre total d'unités.

$a_i$ : Coût à vide de l'unité i.

$b_i$ : Coefficient de coût linéaire de l'unité i.

$c_i$ : Coefficient de coût quadratique de l'unité i.

$X_i^{OFF}$ : Temps durant lequel l'unité i est éteinte.

$X_i^{ON}$ : Temps durant lequel l'unité i est allumée.

$ST_i(t)$ : Coût de redémarrage de l'unité i à l'instant t.

$CSC_i$ : Coût de redémarrage à froid de l'unité i.

$HSC_i$ : Coût de redémarrage à chaud de l'unité i.

$FLC_i$ : Coût de production calculé lors d'un fonctionnement en pleine charge de l'unité i.

$SC_i$ : Durée de redémarrage à froid de l'unité i.

$MUT_i$ : Temps minimum de redémarrage de l'unité i.

$MDT_i$ : Temps minimum d'arrêt de l'unité i.

# Introduction générale

Étant donné que les réseaux électriques sont des systèmes directement liés aux activités humaines, la demande varie d'heure en heure selon ces activités. En effet, la demande est généralement plus importante pendant la journée et en début de soirée tandis qu'elle l'est moins en fin de soirée. Il serait simple d'engager des unités de production pour satisfaire la demande et les garder à cet état, cependant ceci n'est pas économique car l'engagement de plusieurs unités en même temps tend à être trop coûteux. C'est pour cela qu'on envisage l'extinction des unités non nécessaires.

L'énergie électrique est obtenue par la conversion des énergies primaires (mécanique, chimique, nucléaire) cependant, ces énergies peuvent être renouvelables. La conversion des énergies primaires en énergie électrique se fait au niveau des centrales électriques. Dans notre problème UCP, on a pris seulement les centrales thermiques comme étant les unités de production qui sont constitués de plusieurs groupes.

Le problème d'engagement des turbines est donc un problème de minimisation du coût de production total en se basant sur l'allumage et l'extinction des unités de production de façon à satisfaire la demande fortuite. La résolution de ce problème n'est pas simple car le nombre de combinaisons possibles d'unités engagées peut être assez grand. Au fil des années, plusieurs méthodes ont été appliquées à ce problème. Ces dernières comprennent les méthodes déterministes : liste de priorité, programmation dynamique, relaxation lagrangienne…etc Cependant, de nos jours, la recherche est beaucoup plus axée sur les méthodes avancées telles que les algorithmes génétiques ou le recuit simulé.

La structure de travail du mémoire est comme suit :

Le chapitre I concerne la formulation mathématique de l'UCP, le sous-problème de répartition économique et l'état de l'art des méthodes de résolution de l'UCP.

Le chapitre II est consacré à deux méthodes classiques : méthode de liste de priorité et programmation dynamique. Les méthodologies de ces derniers sont exposées suivies de simples exemples.

Dans le chapitre III, on se base sur trois méthodes méta heuristiques : algorithme génétique, optimisation par essaims particulaires et recuit simulé. Des hybridations entre les méthodes sont également proposées.

Enfin, dans le chapitre IV, des applications des méthodes classiques et avancées ont été effectuées sur des systèmes de différentes tailles par le biais d'un logiciel développé spécifiquement pour la résolution de l'UCP.

# CHAPITRE I :

# Introduction au problème d'Engagement des turbines

# I.1. Définition du problème :

L'engagement des turbines (Unit Commitment) se réfère à l'action de démarrer des unités de production dans un réseau électrique et d'éventuellement arrêter ces unités d'une façon économique. Le but principal est de trouver la configuration optimale des unités à notre disposition de façon à réduire le coût de production total à condition de bien respecter un certain nombre de contraintes. Ceci peut être traduit en un problème d'optimisation combinatoire non-linéaire soumis à deux types de contraintes : les contraintes d'association et les contraintes locales. Les contraintes d'association concernent la somme de génération de toutes les unités engagées. Cette somme doit non seulement satisfaire la demande prévisionnelle, mais également garantir la réserve tournante. Les contraintes locales sont appliquées à chaque unité individuellement et imposent une limite de fonctionnement sur les unités.

Les unités de production sont d'habitude divisées en trois catégories : unités must-run de base, les unités de charge intermédiaire et les unités de charge de pointe.

Les unités must-run ont la particularité d'être moins coûteuses, d'avoir un coût de redémarrage assez importants et des temps minimum d'arrêt et de redémarrage plutôt longs. Ces unités devraient être opérationnelles pour de longues heures pour des raisons économiques et pour des exigences de fiabilité.

Les unités intermédiaires sont sujettes à des contraintes de temps minimum d'arrêt et de redémarrage et ont des coûts moyens de production et de redémarrage. Généralement parlant, ces unités sont opérationnelles lorsque les unités must-run ne satisfont pas la demande.

Les unités de demande de pointe sont des unités qui peuvent être redémarrées instantanément car elles ne sont généralement pas soumises aux contraintes de temps minimum d'arrêt et de redémarrage. Ces unités sont nécessaires lors des heures de pointes.

# I.2. Formulation du problème :

## I.2.1. Fonction objective :

Le coût de production total contient le coût de production de chaque unité et le coût de redémarrage.

La fonction coût de production de chaque unité, qui est calculée à partir du taux de chaleur et du coût du fuel, peut être représentée sous la forme quadratique suivante :

$$F_i\big(P_i(t)\big) = a_i + b_i P_i(t) + c_i P_i^2(t) \tag{1. 1}$$

Ou bien sous la forme linéaire suivante :

$$F_i\big(P_i(t)\big) = Noload\ cost + incremental\ cost * P_i(t) \tag{1. 2}$$

Le coût total de production pendant une période $N_t$ est le suivant :

$$F_T = \sum_{t=1}^{N_t} \sum_{i=1}^{N} \big[F_i\big(P_i(t)\big) U_i(t)\big] \tag{1. 3}$$

À ce coût on ajoute le coût de redémarrage (ST) qui est exprimé en termes de temps (heures) :

$$ST_i(t) = \begin{cases} HSC_i \ \ , si\ MDT_i \leq X_i^{OFF} \leq MDT_i + SC_i \\ CSC_i \ \ , si\ X_i^{OFF} > MDT_i + SC_i \end{cases} \tag{1. 4}$$

Finalement la formulation mathématique du problème à résoudre est la suivante [1] :

$$F_T = \min_{P_i(t), U_i(t)} \Big[\sum_{t=1}^{N_t} \sum_{i=1}^{N} \big[F_i\big(P_i(t)\big) U_i(t) + ST_i(t) U_i(t)\big]\Big] \tag{1. 5}$$

$$i = 1, \dots \dots N$$

$$t = 1, \dots \dots N_t$$

## I.2.2. Contraintes du problème d'engagement des turbines :

Il existe plusieurs types de contraintes qui permettent de limiter le choix d'allumer ou éteindre une unité de production. On envisage donc d'imposer autant de contraintes possibles pour limiter l'espace de recherche et donc réduire la dimensionnalité du problème. On distingue deux types de contraintes : Les contraintes globales de demande et de réserve, et les contraintes liées aux caractéristiques individuelles de chaque unité. La

liste des contraintes citées dans ce qui suit n'étant pas exhaustive, on peut faire subir au réseau différentes contraintes selon le besoin.

a. Contrainte d'équilibre de puissance :

La somme des puissances de sortie de chaque générateur opérationnel doit être égale à la somme de la charge totale et les pertes :

$$\sum_{i=1}^{N} P_i(t)U_i(t) = P_D(t) + P_L(t) \quad (1.6)$$

b. Contrainte de réserve :

Il est important de prendre en compte l'incertitude liée à toute prévision. C'est pour cela qu'il faut garantir la disponibilité d'une réserve de capacité de production disponible sans avoir à démarrer de nouvelles centrales. La réserve tournante joue donc le rôle de compensation des pertes lorsqu'une ou plusieurs unités ne sont plus disponibles. Cette réserve est de 10% de la demande. On note $P_R(t)$ la quantité de réserves tournantes.

$$\sum_{i=1}^{N} P_i^{max}(t)U_i(t) \geq P_D(t) + P_R(t) \quad (1.7)$$

c. Limites de production des unités :

La production délivrée d'une centrale est bornée entre deux valeurs extrêmes.

$$P_i^{min} \leq P_i(t) \leq P_i^{max} \quad (1.8)$$

d. Le temps minimum de redémarrage :

L'extinction d'une unité ne peut se faire que si elle reste opérationnelle pendant le temps minimum de démarrage.

$$MUT_i \leq X_i^{ON} \quad (1.9)$$

e. Le temps minimum d'arrêt :

De même, le démarrage ne peut se faire que si on dépasse un temps minimum d'extinction.

$$MDT_i \leq X_i^{OFF} \quad (1.10)$$

## I.3. Sous-problème : Répartition économique

La répartition économique est un sous problème de l'engagement des turbines qui consiste à déterminer la production optimale des unités dans le but de minimiser les coûts de production [1]. Ainsi, l'état des unités doit être préalablement déterminé et afin d'évaluer la planification, il est impératif d'effectuer une répartition économique. C'est alors un secteur d'étude essentiel dans les réseaux électriques.

La variable à optimiser est donc le coût de production étant donné qu'il faut distribuer la puissance active demandée entre les différentes centrales du réseau d'une manière la plus économiquement possible. Le type de problème traité dans le document présent a plusieurs limitations qui réduisent sa complexité. En effet, la puissance active générée par la centrale est le seul paramètre pris en considération pouvant influencer le coût de production et les pertes de puissances sont négligées. De plus, la répartition économique est un problème d'optimisation statique, c'est-à-dire que le problème est résolu à un instant précis. Lorsque le problème prend une ampleur dynamique, on fait donc face au problème « unit commitment ».

### I.3.1. Méthodologie de résolution de la répartition économique :

Plusieurs méthodes classiques et avancées existent, cependant afin de mieux évaluer les différentes méthodes de résolution de l'UCP traitées dans les prochains chapitres, la méthode classique de lambda-logique est utilisée [3]. Cette méthode est simple, rapide et se fait en une seule itération. Supposant qu'on veuille effectuer une répartition économique sur un certain nombre d'unités de production au cours d'un certain nombre d'heures, les étapes à effectuer sont les suivantes :

**Étape 1** : Pour la répartition économique d'un certain nombre d'unités de production K, les conditions de répartition sont les suivantes :

$$\frac{dF_1}{dP_1} = \frac{dF_2}{dP_2} = \frac{dF_3}{dP_3} = \cdots = \frac{dF_K}{dP_K} = \lambda \qquad (1.\ 11)$$

$$P_1 + P_2 + \cdots + P_K = P_D \qquad (1.\ 12)$$

Tel que lambda représente la variation du coût de production et $P_D$ représente la demande prévisionnelle.

À l'unité i :

$$F_i(P_i) = a_i + b_i P_i + c_i P_i^2 \ \$/hr \quad (1.\ 13)$$

$$\frac{dF_i}{dP_i} = \lambda = \alpha_i + \beta_i P_i \qquad \$/Mwhr \quad (1.\ 14)$$

Ainsi l'équation (1. 14) donne la variation linéaire du coût incrémental en fonction de la puissance produite :

$$P_i = \frac{(\lambda - \alpha_i)}{\beta_i}; i = 1 \text{ à } K. \quad (1.\ 15)$$

**Étape 2** : Cette étape consiste à calculer les variations de coûts minimales et maximales :

$$\lambda_{i\,min} = \frac{dF_i}{dP_{i\,min}} \quad (1.\ 16)$$

$$\lambda_{i\,max} = \frac{dF_i}{dP_{i\,max}} \quad (1.\ 17)$$

Puis, l'arrangement des lambdas minimales et maximales est effectué afin d'établir une liste par ordre croissant.

**Étape 3** : Cette étape consiste à préparer les puissances demandées et les pentes.

$$PPD_j = \sum_{i=1}^{N} P_i \ avec \ P_i = \begin{cases} P_{\max i} & si\ \lambda_j < \lambda_{min\,i} \\ P_{\min i} & si\ \lambda_j > \lambda_{max\,i} \\ \frac{(\lambda_j - b_i)}{2c_i} & si\ \lambda_{min\,i} > \lambda_j > \lambda_{max\,i} \end{cases} \quad (1.\ 18)$$

$$m_j = \frac{\lambda_{j+1} - \lambda_j}{PPD_{j+1} - PPD_j} \quad (1.\ 19)$$

**Étape 4 :** L'estimation des puissances se fait à partir de la puissance demandée $P_D$. En effet, on identifie l'intervalle J et J+1 où se trouve la puissance demandée parmi les puissances PPD. Ainsi, on calcul $\lambda_{NEW}$ à partir de l'équation suivante :

$$\lambda_{NEW} = m_J\left(P_D(h) - PPD_J\right) + \lambda_J \quad (1.\ 20)$$

En utilisant cette valeur, on détermine la puissance délivrée par chaque unité :

$$P_i = \begin{cases} P_{\max i} & si\ \lambda_{NEW} < \lambda_{min\,i} \\ P_{\min i} & si\ \lambda_{NEW} > \lambda_{max\,i} \\ \frac{(\lambda_j - b_i)}{2c_i} & si\ \lambda_{min\,i} > \lambda_{NEW} > \lambda_{max\,i} \end{cases} \quad (1.\ 21)$$

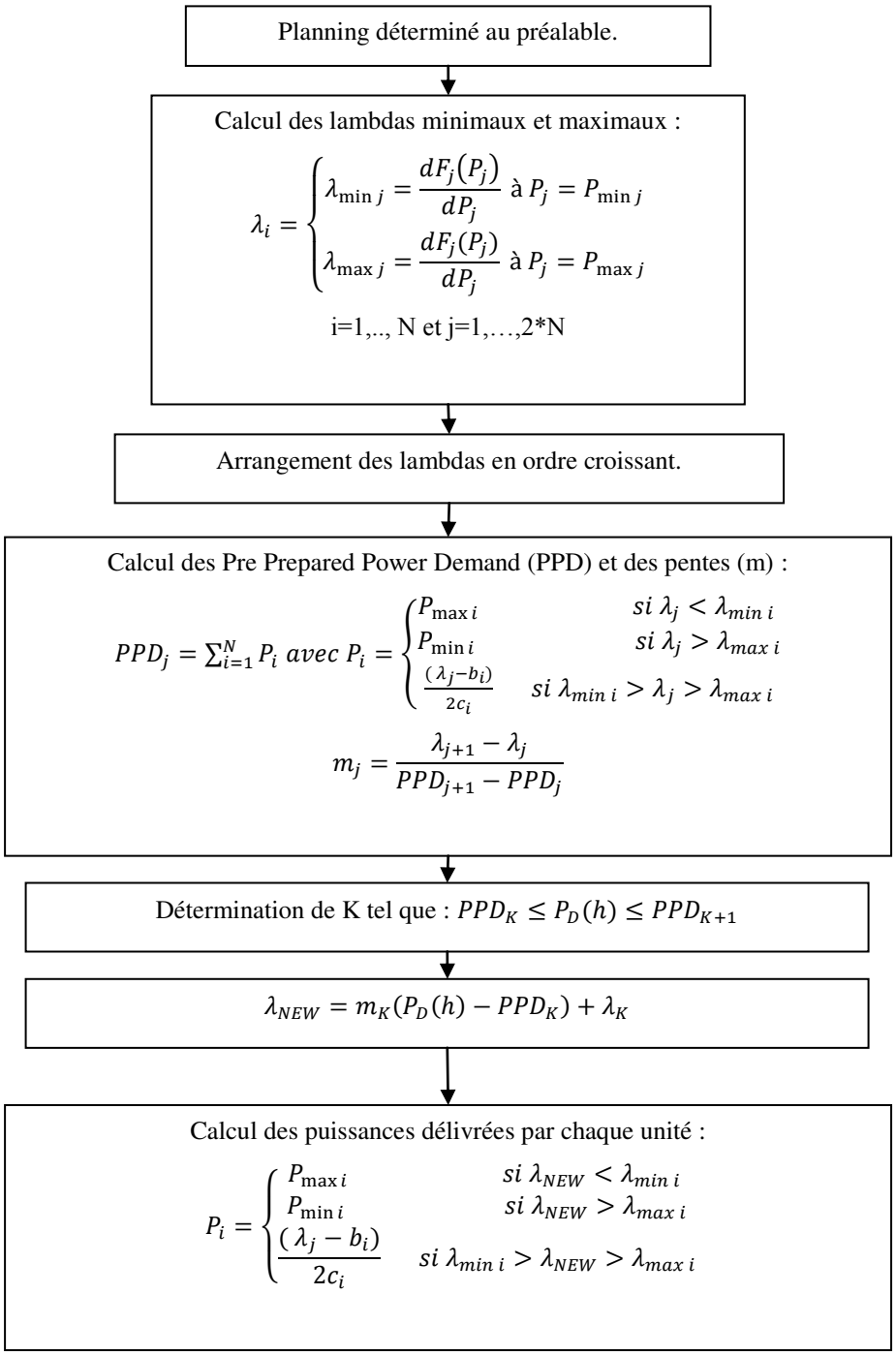



Figure 1. 1 Méthodologie de la méthode de lambda logique pour la répartition économique.

## I.3.2. Exemple d'application :

En appliquant la méthode de lambda logique sur un système de 4 unités (Annexe A), la première étape consiste à calculer les lambdas minimaux et maximaux à partir des dérivées fonctions coût de production et des limites de production (tableau I).

Tableau I. Les lambdas minimaux et maximaux.

| Unité | Fonction coût de production | Limite de production (MW) | | Lambdas ($/MW) | |
|---|---|---|---|---|---|
| | | $P_{min}$ | $P_{max}$ | $\lambda_{min}$ | $\lambda_{max}$ |
| 1 | $0.0021P^2 + 16.83P + 684.74$ | 300 | 75 | 17.1450 | 18.0900 |
| 2 | $0.0042P^2 + 16.95P + 585.62$ | 250 | 60 | 17.4540 | 19.0500 |
| 3 | $0.0018P^2 + 20.74P + 213$ | 80 | 25 | 20.8300 | 21.0280 |
| 4 | $0.0034P^2 + 23.6P + 252$ | 60 | 20 | 23.7360 | 24.0080 |

La deuxième étape consiste à calculer les Pre-Prepared Power Demand (PPD) :

Tableau II. PPD

| J | $\lambda(J)$ | $PPD_j = \sum_{i=1}^{N} P_i$ | Puissances de génération des unités | | | |
|---|---|---|---|---|---|---|
| | | | Unité 1 | Unité 2 | Unité 3 | Unité 4 |
| 1 | 17.1450 | 180.0000 | 75.0000 | 60.0000 | 25.0000 | 20.0000 |
| 2 | 17.4540 | 253.5714 | 148.5714 | 60.0000 | 25.0000 | 20.0000 |
| 3 | 18.0900 | 480.7143 | 300.0000 | 135.7143 | 25.0000 | 20.0000 |
| 4 | 19.0500 | 595.0000 | 300.0000 | 250.0000 | 25.0000 | 20.0000 |
| 5 | 20.8300 | 595.0000 | 300.0000 | 250.0000 | 25.0000 | 20.0000 |
| 6 | 21.0280 | 650.0000 | 300.0000 | 250.0000 | 80.0000 | 20.0000 |
| 7 | 23.7360 | 650.0000 | 300.0000 | 250.0000 | 80.0000 | 20.0000 |
| 8 | 24.0080 | 690.0000 | 300.0000 | 250.0000 | 80.0000 | 60.0000 |

Si l'on prend une puissance demandée : $P_D = 460\ MW$ et après identification de J tel que $PPD(J) \leq P_D \leq PPD(J+1)$, on obtient : $\lambda_{NEW} = 18.0320$.

Ainsi, on détermine les puissances délivrées de chaque unité :

$P_1 = 286.19$ MW; $P_2 = 128.81$ MW; $P_3 = 25$ MW; $P_4 = 20$ MW.

# I.4. État de l'art : Les différentes méthodes de résolution de l'UCP

Il est difficile de résoudre le problème d'engagement des turbines d'une façon rigoureuse à cause de sa nature combinatoire et de sa dimension importante. Cela dit, on trouve dans la littérature plusieurs méthodes simplificatrices qui prennent en compte des approximations. Ces méthodes peuvent être classées sous trois catégories : les méthodes déterministes, méta heuristiques et hybrides.

Dans ce qui suit, on commence par décrire les approches classiques basées sur des méthodes exactes. Puis, on décrit les approches basées sur les méta-heuristiques et des hybridations entre ces dernières et les méthodes déterministes.

## I.4.1. Méthodes déterministes :

En 1987, Cohen a étudié un bon nombre de références et a discuté les différentes méthodes disponibles à ce temps là [2] : Les algorithmes énumératives (programmation dynamique et méthode de branch-and-bound) et les méthodes de relaxation et de décomposition (relaxation de Lagrange et la décomposition de Bender).

La programmation dynamique est l'une des premières méthodes d'optimisation à être appliquée au problème d'UC. Son avantage par rapport à la méthode de liste de priorité est de maintenir la faisabilité de la solution. Cependant, la méthode fait face au problème de dimensionnalité. En effet, la nature énumérative de la méthode engendre des temps de calculs inacceptables pour un nombre d'unités de production important. Cet aspect est discuté en détails dans les chapitres suivants.

La relaxation lagrangienne, quant à elle, est capable de résoudre des problèmes d'engagement des turbines à grandes tailles en un temps de calcul assez réduit. Cette méthode est plus récente que DP et a été appliquée pour la première fois au problème d'UC sans prendre en considération quelques contraintes [4]. À partir de cette approche, on ajoute des contraintes de couplage à la fonction coût en utilisant des multiplicateurs de Lagrange, ainsi on détend le problème et on le décompose en sous-problèmes. La méthode LR a donc l'avantage d'être plus au moins rapide car le temps de calculs augmente linéairement par rapport à la dimension du problème [5]. En 1994, Sheble est arrivé à la

conclusion que LR était la meilleure méthode déterministe pour résoudre l'UCP en un temps raisonnable [6].

## I.4.2. Méthodes métaheuristiques :

Les méthodes méta heuristiques sont particulièrement intéressantes à utiliser : elles s'adaptent facilement à différentes variantes du problème, elles peuvent aborder des fonctions de coût complexes et peuvent résoudre le même problème pour des fonctions objectives complètement différentes. Récemment, des recherches ont été faites sur une multitude de méthodes. On discerne deux types de méta heuristiques : les algorithmes à solution unique et les algorithmes à population.

### I.4.2.1. Méthodes à solution unique :

La méthode du recuit simulé (simulated annealing) simule la procédure de refroidissement graduelle du métal, jusqu'à ce que l'énergie du système acquière la valeur minimale globale. Dans une méthode conventionnelle du recuit simulé, on effectue des perturbations aléatoires à chaque température et on évalue le coût généré. À partir de ce coût, soit on accepte, soit on rejette la perturbation [11].

La recherche taboue (TS) est une puissante procédure d'optimisation qui a été appliqué avec succès à plusieurs problèmes d'optimisation combinatoires. Cette méthode utilise un système de mémoire flexible et permet d'éviter les minima locaux [12].

### I.4.2.2. Méthodes à population :

Parmi ces méthodes, la programmation évolutionnaire EP [7] où des populations d'individus évoluent à travers des changements aléatoires, des compétitions et des sélections. Le problème UC est vu comme étant un candidat de reproduction après codage en une chaine de symboles.

L'optimisation par essaims particulaires (PSO) est une méthode qui simule le comportement social des oiseaux en formation. Les entités dans la population sont nommées particules qui correspondent à des solutions faisables pour le problème de l'UCP.

On peut également citer d'autres méthodes qu'on retrouve fréquemment dans la littérature : algorithmes mémétiques [13], algorithmes génétiques (GA) [14], recherche par colonies de fourmilles (ant colony search algorithm) [15]…etc.

### I.4.3. Méthodes hybrides :

L'intérêt que les chercheurs portent aux méthodes hybrides ne cesse de grandir dans le domaine de l'optimisation. Ainsi, le problème de l'engagement des turbines a été traité plusieurs fois dans le passé en combinant des métaheuristiques à d'autres métaheuristiques ou bien d'autres méthodes déterministes.

Il est intéressant de combiner les métaheuristiques à population et les métaheuristiques à solution unique car ces deux familles de méthodes ont des caractéristiques complémentaires : les méthodes à solution unique favorisent l'exploitation tandis que les méthodes à population favorisent l'exploration [8]. Ainsi, des méthodes hybrides permettant de diversifier l'espace de recherche et d'intensifier la recherche localement ont été appliquées à l'UCP : programmation évolutionnaire et recherche tabou [20], recherche locale et algorithme génétique [46],…etc. Il est également possible d'hybrider des méthodes de même famille : optimisation par essaims particulaires et algorithme de recherche gravitationnelle [9], algorithme génétique et optimisation par essaims particulaires [45], recuit simulé et recherche tabou [18],…etc.

En ce qui concerne l'hybridation entre les métaheuristiques et les méthodes déterministes, le but est d'obtenir des solutions optimales en un temps de calcul réduit. En effet, les méthodes déterministes sont connues pour être lentes et requièrent une mémoire de grande taille. Ainsi, en les combinant aux méthodes métaheuristiques, il est possible d'améliorer leur efficacité. Parmi ces méthodes hybrides appliquées à l'UCP on trouve dans la littérature : relaxation lagrangienne et algorithme génétique [16], relaxation lagrangienne et programmation évolutionnaire [17], programmation dynamique et optimisation par essaims particulaires [10], programmation dynamique et logique floue [19], …etc.

# CHAPITRE II :

# Méthodes déterministes pour la résolution de l'UCP

# II.1. Liste de priorité :

## II.1.1. Introduction :

La méthode de liste de priorité est la méthode la plus basique de résolution du problème d'engagement des turbines. Le but est d'engager d'abord les unités les plus économiques qui respectent certaines contraintes de l'UCP. Cette méthode est donc simple et très rapide mais ne respecte pas quelques contraintes techniques.

## II.1.2. Méthodologie :

Le principe consiste à établir une liste où les unités sont classées en fonction de leur coût de production calculé lors d'un fonctionnement en pleine charge FLC (Full Load Cost) [1]. Le classement est fait par ordre croissant à partir d'une des expressions suivantes :

$$FLC_i\big(P_i(t)\big) = (a_i + b_i P_{max\,i}(t) + c_i P^2_{max\,i}(t))/P_{max\,i}(t) \quad (2.\ 1)$$

$$FLC_i\big(P_i(t)\big) = Noload\ cost + incremental\ cost * P_{max\,i}(t) \quad (2.\ 2)$$

Après avoir établi la liste de priorité, on engage les unités une par une dans l'ordre de la liste et on calcule les puissances maximales et minimales cumulatives. Pour un système de N unités, on obtient donc N cas possibles. À chaque heure, on sélectionne les cas qui satisfont la demande et on applique la répartition économique afin de déterminer le coût de production totale, le cas étant le moins coûteux est maintenu [21].

Cette méthode, quoique très rapide, est loin d'être optimale. En effet, les seules contraintes prises en considération sont : contraintes de puissance, de réserve et limite de production. La négligence des contraintes de temps minimum d'arrêt et de redémarrage peut compromettre la faisabilité de la solution. Cela dit, cette méthode peut être utilisée comme solution initiale pour plusieurs autres méthodes afin de mieux se situer aux alentours de la solution générale [22].

### II.1.3. Exemple d'application :

Dans ce qui suit, la méthode de liste de priorité est appliquée à l'exemple du système de 3 unités (Annexe A). La première étape consiste à établir la liste de priorité. Le calcul du FLC (Full Load Coast) est fait à partir de l'équation (2.2), comme montré dans le tableau III.

Tableau III. Coût du fonctionnement en pleine charge de chaque unité (système 3 unités).

| Unité | A | B | C |
|---|---|---|---|
| FLC ($) | 2500 | 1200 | 1000 |

On engage les unités l'une après l'autre selon la liste de priorité de façon à avoir autant de combinaisons que d'unités. On calcul les puissances minimales et maximales cumulatives pour chaque combinaison. Les calculs effectués à cette étape sont exposés dans le tableau IV.

Tableau IV. Combinaisons possibles et puissances cumulatives du système 3 unités (liste de priorité).

| État | Combinaisons | | | Pmin | Pmax |
|---|---|---|---|---|---|
| | C | B | A | | |
| 1 | 1 | 0 | 0 | 10 | 50 |
| 2 | 1 | 1 | 0 | 60 | 150 |
| 3 | 1 | 1 | 1 | 210 | 400 |

Ensuite, on détermine les combinaisons capables de satisfaire la demande (tableau V). À chaque heure, on maintient la combinaison la moins coûteuse sans prendre en considération l'infraction des contraintes de temps minimal d'arrêt et de redémarrage. Dans notre cas, à l'heure 2, deux combinaisons peuvent satisfaire la demande cependant d'après la répartition économique, la combinaison la moins coûteuse est la combinaison 2 (figure 2.1).

Tableau V. Combinaisons faisables du système 3 unités (liste de priorité).

| | Heure 1 : 150 MW | Heure 2 : 300 MW | Heure 3 : 200 MW |
|---|---|---|---|
| Combinaisons faisables | 2 | 3 | 3 |

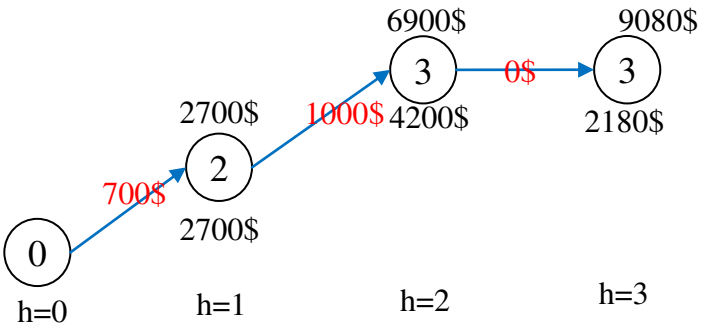



Figure 2. 1 Stratégie d'engagement par liste de priorité.

Ainsi, on a pu rapidement trouver une solution sans trop élargir l'espace de recherche. Cependant, l'optimum n'est pas forcément atteint car on ne prend pas en considération tous les cas possibles. L'extinction de l'unité A à l'heure 1 correspond à une infraction de temps minimum de redémarrage car le nombre d'heures de mise en démarrage ne dépasse pas le temps minimum d'allumage. Ainsi, cette méthode ne respecte pas les contraintes de temps minimum d'extinction et d'allumage.

## II.2.Programmation Dynamique :

### II.2.1. Introduction :

La base de la programmation dynamique est la théorie d'optimalité élucidée par Bellman en 1957. Le mot « programmation » ne fait pas référence à sa définition moderne mais plutôt à un ensemble de règles qu'on peut suivre afin de résoudre un problème. C'est une technique d'optimisation dont le principe consiste à décomposer le problème en une série de sous problèmes et de développer une solution optimale du problème de départ étape par étape. La solution est donc le résultat d'une séquence de décisions [23].

L'approche de résolution de cette méthode se base sur trois principes [24] :

- Division en sous-problèmes.
- Construction d'une table où les solutions des sous-problèmes sont enregistrées afin d'éviter la répétition des calculs.
- Combinaison des solutions des sous-problèmes pour former la solution du problème original.

Cette méthode, développée pour la première fois dans le domaine de l'engagement des turbines par P. G. Lowery en 1966 [25], a été raffinée d'avantage par Ayoub et Patton en 1971 [26], Pang et Chen en 1976 [27], Sheble en 1981 [28] et pleins d'autres dans les années qui suivent.

### III.2.2. Principe de fonctionnement :

Fondamentalement, la méthode de programmation dynamique pour l'UCP examine chaque état possible à chaque intervalle de temps. En effet, un état correspond à une combinaison d'unités de production en une heure si on parle d'engagement de turbines à court terme. Sachant qu'on peut parcourir le problème à partir de la dernière heure jusqu'à la première heure (programmation dynamique en arrière), on préfère le parcourir de l'heure initiale à l'heure finale (programmation dynamique en avant) car on a l'avantage d'avoir les conditions initiales et l'état initial des unités. Par ailleurs, l'historique précédent des états nous permet de calculer le coût de redémarrage qui est en fonction du temps [1].

La dimensionnalité du problème est un aspect très important qu'il faut prendre en considération. En effet, pour un système de N unités, $(2^N - 1)$ états existent. Sachant que l'optimisation en une période de temps est divisée en plusieurs intervalles T, la solution au problème est un chemin liant les états à chaque intervalle. Le nombre de chemins possibles est donc : $(2^N - 1)^T$.

Par exemple : 5 unités, 24 heures.

$(2^N - 1)^T = (2^5 - 1)^{24} = 6.2\ 10^{35}\ combinaisons.$

On doit également prendre en considération les différentes contraintes à notre disposition. Ces dernières nous permettront de rejeter certains états non faisables et ainsi réduire la dimensionnalité du problème. En ce qui concerne les combinaisons possibles, on a deux possibilités :

- Une stricte liste de priorité. N états possibles pour N unités.
- Une énumération complète des états possibles. $(2^N - 1)$ états pour N unités.

Bien qu'il soit plus rapide d'énumérer les combinaisons selon une liste de priorité stricte, on retrouve une solution plus optimale en énumérant tous les cas possibles. Ceci n'est pas problématique lorsque le système est relativement petit, cependant les réseaux réels sont beaucoup plus volumineux [29].

### II.2.3. Méthodologie :

Les étapes de la programmation dynamique en avant sont :

1. Faire soit une énumération complète des cas possibles ($2^N - 1$ états pour N unités), soit une liste de priorité (N états pour N unités).
2. Trouver pour chaque heure H tous les cas pouvant satisfaire la demande. C'est-à-dire que la somme des puissances maximales/minimales des unités opérationnelles doit être supérieure/inférieure ou égale à la demande et réserve.
3. Trouver à l'heure précédente H-1 les cas réalisables et effectuer des combinaisons entre les cas réalisables de l'heure actuelle et ceux de l'heure précédente.
4. Enregistrer le nombre d'heures pendant lesquelles l'unité a été maintenue éteinte afin de déterminer la nature du coût de redémarrage (à chaud ou à froid).
5. Si les contraintes de temps minimum d'allumage et d'extinction ne sont pas respectées, la combinaison est rejetée. Sinon, on calcul la somme des coûts de production de chaque unité opérationnelle après avoir effectué une répartition économique (ED) et on enregistre la stratégie pour chaque heure.
6. Si tous les cas réalisables à l'heure H-1 ont été traités, on passe à l'étape 7 sinon on passe à l'étape 3.
7. Si tous les cas réalisables à l'heure H ont été traités, on passe à l'étape 8 sinon on passe à l'étape 2.
8. On calcul le coût de production total pour chaque stratégie (chemin). La solution correspond à la stratégie la moins coûteuse.

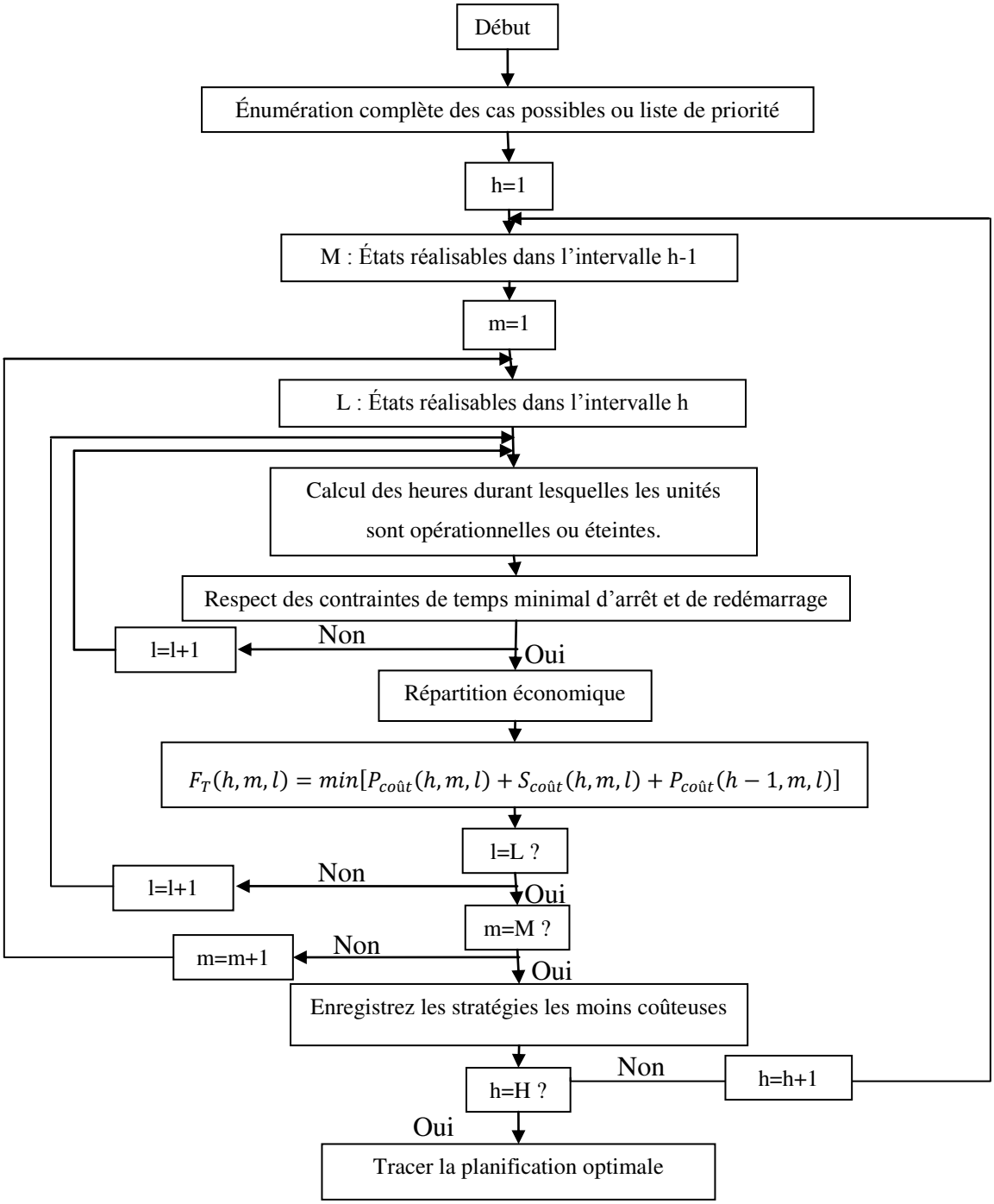



Figure 2. 2 Algorithme de la programmation dynamique en avant.

## II.2.4. Réduction du nombre de stratégies enregistrées :

On définit un état (H, I) comme étant la combinaison I à l'heure H. Si l'état ne satisfait pas la demande et ne garantit pas la réserve, la répartition économique n'est pas effectuée. Si l'état ne respecte pas les contraintes de temps minimum d'arrêt et de redémarrage, il est rejetée et n'est pas considéré pour une éventuelle combinaison avec les états de l'heure suivante. Ainsi, les différentes contraintes permettent de réduire l'espace de recherche. Cependant, ceci n'est pas suffisant lorsque le système a un nombre d'unités trop important.

Il est possible de limiter d'avantage l'espace de recherche en n'enregistrant que les stratégies les moins coûteuses à chaque heure. Ainsi, les stratégies les plus chères sont éliminées. Si on décide de faire une énumération complète, le nombre maximal de stratégies peut atteindre $2^N - 1$. La figure ci-contre illustre ce processus [1].

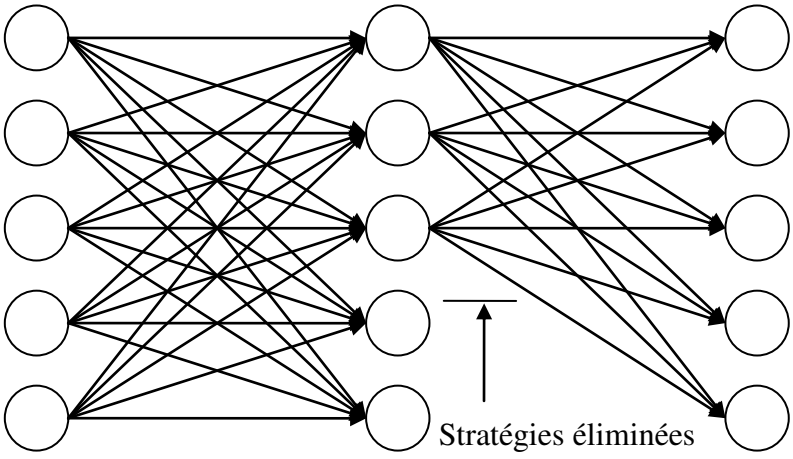



Figure 2. 3 Limitation de l'espace de recherche (programmation dynamique) [1].

Cependant, ceci ne garantit pas une solution optimale, ni même une solution faisable. En effet, il se peut que les stratégies les plus chères éliminées au début de la recherche soient les seules stratégies faisables capables de respecter certaines contraintes.

## II.2.5. Exemple d'application :

Dans le but d'éclaircir la méthode, un simple exemple est étudié : un système de 3 unités (annexe A). On ne prend en considération que quelques contraintes pour simplifier la procédure de recherche de la solution optimale. Sachant que le système contient 3 unités de production et que le besoin horaire est sur 3 heures, la programmation dynamique à énumération complète prend en considération le nombre de chemins suivant :

Nombre total de chemins possibles : $(2^3 - 1)^3 = 343$.

L'énumération de toutes les combinaisons possibles (ou bien l'énumération par liste de priorité) est nécessaire afin de déterminer les cas pouvant satisfaire la demande. La transition entre les combinaisons possibles étant illustrée ci-dessous, le but est de déterminer le chemin permettant d'obtenir un coût de production totale optimal tout en respectant les différentes contraintes.

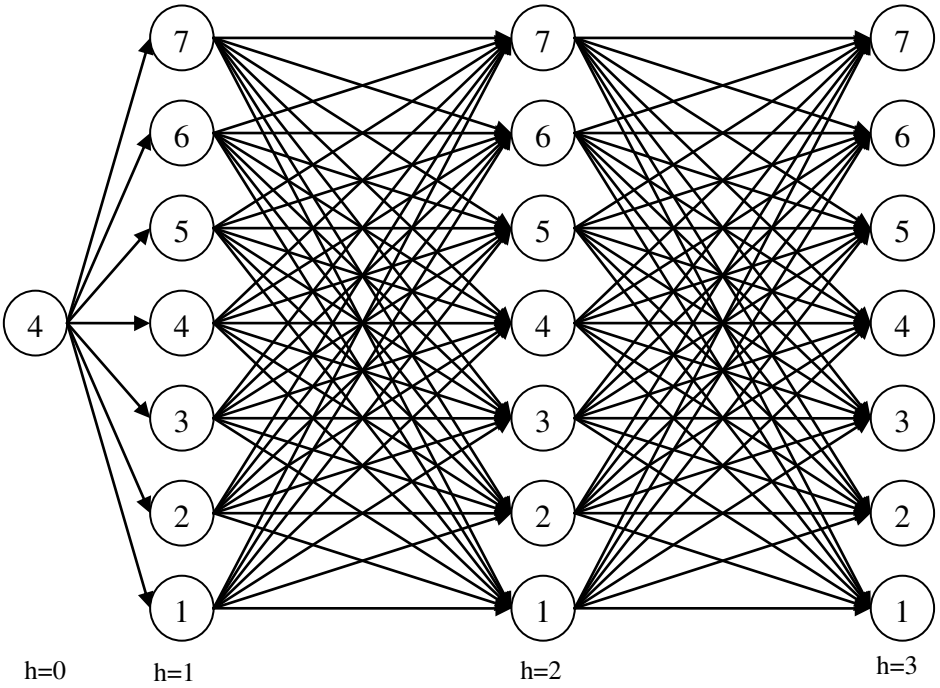



Figure 2. 4 Transition entre les combinaisons possibles.

La deuxième étape à effectuer consiste à déterminer les combinaisons faisables parmi les combinaisons possibles. Ceci se fait en additionnant les puissances de production minimales et maximales des unités opérationnelles. Si la puissance totale demandée est dans l'intervalle des limites de production des unités opérationnelles, la combinaison est donc satisfaisante.

Tableau VI. Combinaisons possibles du système 3 unités (programmation dynamique).

| Pmax (MW) | | 0 | 50 | 100 | 150 | 250 | 300 | 350 | 400 |
|---|---|---|---|---|---|---|---|---|---|
| Pmin (MW) | | 0 | 10 | 50 | 60 | 150 | 160 | 200 | 210 |
| Combinaisons | C | 0 | 1 | 0 | 1 | 0 | 1 | 0 | 1 |
| | B | 0 | 0 | 1 | 1 | 0 | 0 | 1 | 1 |
| | A | 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 |
| État | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |

Tableau VII. Combinaisons faisables pour une demande en charges de 3 heures (programmation dynamique)

| | Heure 1 : 150 MW | Heure 2 : 300 MW | Heure 3 : 200 MW |
|---|---|---|---|
| Combinaisons faisables | 3,4 | 5,6, 7 | 4,5, 6 |

La transition entre les combinaisons faisables est illustrée comme ci-dessous, sachant que l'état initial correspond à l'état 4. Cependant, certaines transitions ne sont pas faisables car les contraintes de temps minimum d'arrêt et de démarrage ne sont pas respectées. La transition 4-3 n'est pas faisable car l'unité A a un temps minimum d'arrêt égal à 3 heures. Ainsi toutes les transitions allant de 3 à 5, 6 et 7 ne sont également pas faisables. L'unité B a un temps minimum de démarrage égal à 1 heure, ainsi les transitions 7-5, 7-4, 6-5 et 6-4 ne sont pas faisables.

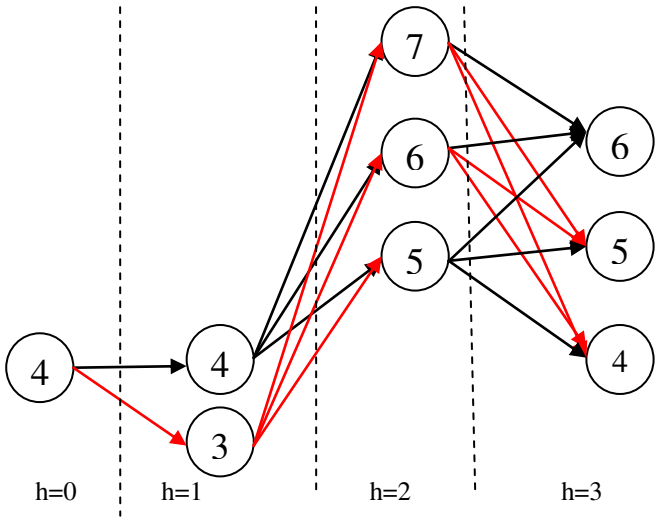



Figure 2. 5 Transitions entre les combinaisons faisables (programmation dynamique).

L'étape suivante consiste à calculer les coûts de production instantanés de chaque combinaison après avoir effectué une distribution économique. On ajoute également les coûts de redémarrage et ainsi on détermine le coût de production total pour chaque chemin. On décide d'éliminer les transitions 7-6 et 5-6 car la transition 6-6 est la moins coûteuse. Ainsi, la stratégie la moins coûteuse est 4-5-4. Évidemment, cet exemple a pour but d'illustrer le principe de la méthode et ne me montre pas sa vraie complexité.

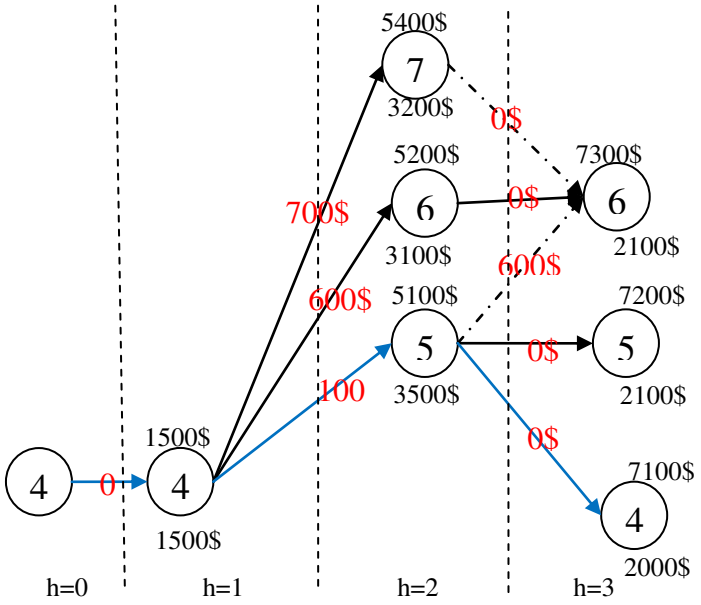



Figure 2. 6 Coût de production total pour chaque chemin (programmation dynamique).

# CHAPITRE III :

# Méthodes méta heuristiques pour la résolution de l'UCP

# III.1. Introduction :

Dans ce chapitre, on traite les méthodes méta heuristiques, plus précisément les techniques d'optimisation basées sur la biologie et la nature et les méthodes de recherche locale. En général, trois concepts sont développés dans le domaine du méta heuristique :

- Algorithmes évolutionnaires.
- Algorithmes d'intelligence distribuée.
- Algorithmes de recherche.

Les algorithmes évolutionnaires sont des méthodes d'optimisation basées sur le principe de « la survie du plus fort » de Darwin. Ce principe stipule que dans la nature, les individus les plus aptes ont une meilleure chance de survivre. Plusieurs disciplines existent telles que [33] : Les algorithmes génétiques, les stratégies d'évolution, la programmation génétique, l'évolution différentielle ...etc.

L'intelligence distribuée ou intelligence en essaims, introduite par Beni en 1989 [34], est le comportement collectif des systèmes auto-organisés et décentralisés. Plusieurs algorithmes basés sur l'intelligence distribuée ont été proposé : optimisation par essaims particulaires, optimisation par colonies de fourmilles, colonie d'abeilles artificielles, algorithme des chauves-souris… etc.

Les méthodes de recherche locale consistent à améliorer une solution initialement complète de manière répétitive en explorant son voisinage. Ces méthodes sont donc à solution unique et plusieurs stratégies d'amélioration de la méthode de recherche locale standard ont été développées : Recuit simulé, recherche taboue, GRASP (greedy randomized adaptive search algorithm),…etc.

La nécessité d'implémentation de ces méthodes au problème de l'UCP est due à la qualité de la solution non satisfaisante obtenue par liste de priorité et du temps d'exécution important de la programmation dynamique. Dans ce qui suit, on développe les méthodologies des algorithmes génétiques, l'optimisation par essaims particulaires et le recuit simulé. Une hybridation entre la recherche locale et l'optimisation par essaims particulaires est effectuée dans le but d'améliorer cette dernière.

# III.2. Algorithmes génétiques :

## III.2.1. Introduction :

Les algorithmes génétiques ont été développés par John Holland à l'université de Michigan. Ce dernier a écrit le premier livre traitant ce sujet intitulé *Adaptation in Natural and Artificial Systems* et publié en 1975. Ces algorithmes sont conçus pour imiter l'évolution biologique dont le processus est une source d'idées pour la résolution de problèmes d'optimisation pratiques et théoriques. En effet, la reproduction mise en conjonction à la sélection naturelle peut résulter au développement d'espèces hautement adaptées. Ceci est intéressant lorsqu'on fait face à un problème d'optimisation parcourant un espace de recherche assez grand et dont la solution ne peut être atteinte qu'à un temps loin d'être raisonnable [35].

Les algorithmes génétiques ont suscité beaucoup d'intérêt après la publication du livre *Genetic Algorithms in Search, Optimization and Machine Learning* en 1989 par David Goldberg. Depuis, cette méthode d'optimisation a été introduite dans plusieurs domaines. Ces algorithmes comprennent trois opérations de bases : la reproduction, le croisement et la mutation. Le pseudo-code des algorithmes génétiques est présenté ci-dessous :

**Algorithmes génétiques :**

Initialisation de la population de taille L : $x = (x_1, x_2 \dots, x_L)$ et du nombre d'itérations N.

Codage binaire de la population (représentation sous forme de chromosomes).

Initialisation des probabilités de croisement $p_m$et de mutation $p_c$.

$i$ :=1

Tant que $i$<$N$

Évaluation $f(x) = \big(f(x_1), f(x_2), \dots, f(x_L)\big)$.

Sélection des parents $(P_1, P_2, \dots, P_L) = Sélection(x_1, x_2 \dots, x_L)$

Si $p_c > r$

Génération des enfants $(E_1, E_2, \dots, E_L) = Croisement(x_1, x_2 \dots, x_L)$

Si $p_m > r$

Génération des mutants $(M_1, M_2, \dots, M_L) = Mutation(x_1, x_2 \dots, x_L)$

Sélection des meilleurs individus de la nouvelle génération.

$i$ :=$i$+1

## III.2.2. L'approche génétique pour la résolution de l'UCP:

Les premières tentatives de résolution de l'UCP par les algorithmes génétiques ont été faites par *S.A.Kazarlis* [36] et *G.B.Sheble* en 1996 [37]. Les multiples variantes diffèrent dans le codage, l'évaluation et la représentation du problème. Cela dit, l'UCP est un problème qui dépend de plusieurs contraintes et les différents processus qui caractérisent l'algorithme génétique peuvent enfreindre ces contraintes. On peut contourner cela en ajoutant un coût de pénalité à la fonction objective en cas de violation d'une contrainte. Cependant, l'implémentation d'une telle formulation n'est pas évidente lorsqu'on fait face à un problème aussi complexe que l'UCP et il est possible qu'à un certain moment, tous les individus de la population ne respectent pas les contraintes. Une autre alternative est d'effectuer un mécanisme de réparation qui consiste à modifier les solutions de façon à satisfaire les contraintes. Ainsi, le temps de calcul est réduit considérablement car on reste toujours dans l'espace de faisabilité. Les différents mécanismes de réparation peuvent également être utilisés dans d'autres méthodes évolutionnaires telles que l'optimisation par essaims particulaires ou le recuit simulé.

## III.2.3. Méthodologie de l'algorithme génétique pour la résolution de l'UCP :

### a. Codage du problème :

Pour les algorithmes génétiques, l'une des étapes les plus importantes est le codage du problème. Chaque membre de la population est représenté par une séquence de bits. En effet, si l'unité est en état OFF, elle est symbolisée par « 0 », tandis que si l'unité est en état ON, elle l'est par « 1 ».

| *Unité* | – – – – – – – – – – – – – – – – – – – – – – – – |
|---|---|
| [1] | [000000100000010001000000] |
| [2] | [000000110000000000001000] |
| [3] | [000000111000000001000000] |
| [4] | [000001010110000001111000] |
| [5] | [000000101001000000011111] |
| [6] | [000111111111110000000000] |
| [7] | [001010111000000010000000] |
| [8] | [111100111111111110000000] |
| [9] | [110000011111111111000000] |
| [10] | [001111111100000010100000] |

Figure 3. 1 Exemple d'une représentation binaire d'un planning UCP.

**b. Initialisation :**

Avant l'initialisation de la recherche, la population initiale doit être générée d'une façon aléatoire. C'est-à-dire qu'une distribution uniforme et aléatoire des « 1 » et « 0 » est appliquée à chaque membre de la population.

**c. Contrainte de puissance :**

La 1ère modification à effectuer sur les membres de la population consiste à engager les unités par ordre de priorité lorsque la demande n'est pas satisfaite [38]. Les unités les plus économiques sont engagées en premier.

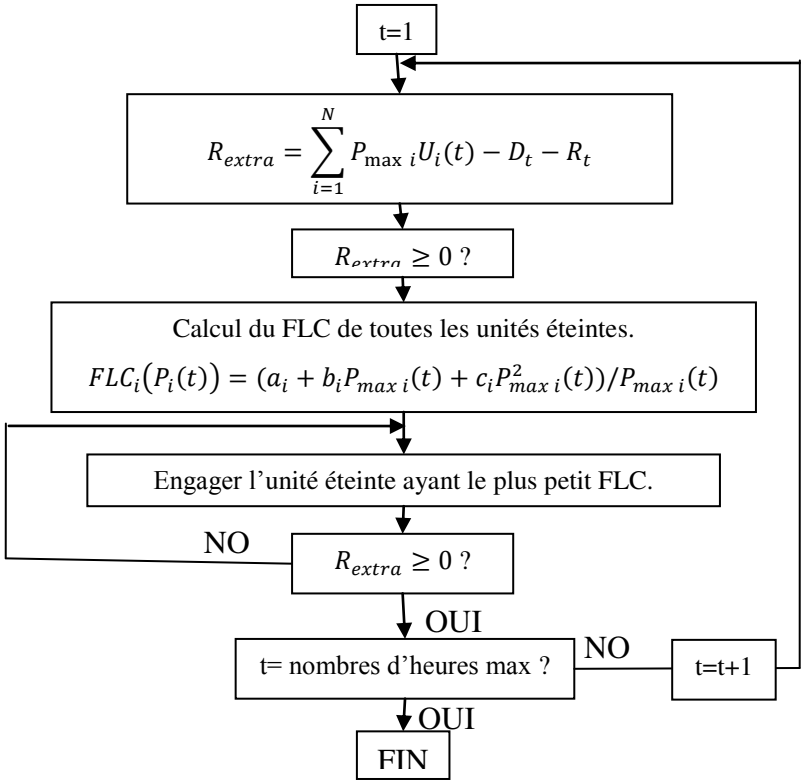



Figure 3. 2 Réparation de violation de la contrainte de puissance.

**d. Extinction des unités excessives :**

La 2ème modification à effectuer est d'éteindre les unités excessives tout en respectant la contrainte de puissance [38]. Les unités les moins économiques sont éteintes en premier tout en s'assurant que la demande et la réserve sont satisfaites. Le désavantage d'un tel mécanisme est la possibilité d'infraction des contraintes de temps minimum d'arrêt et de redémarrage.

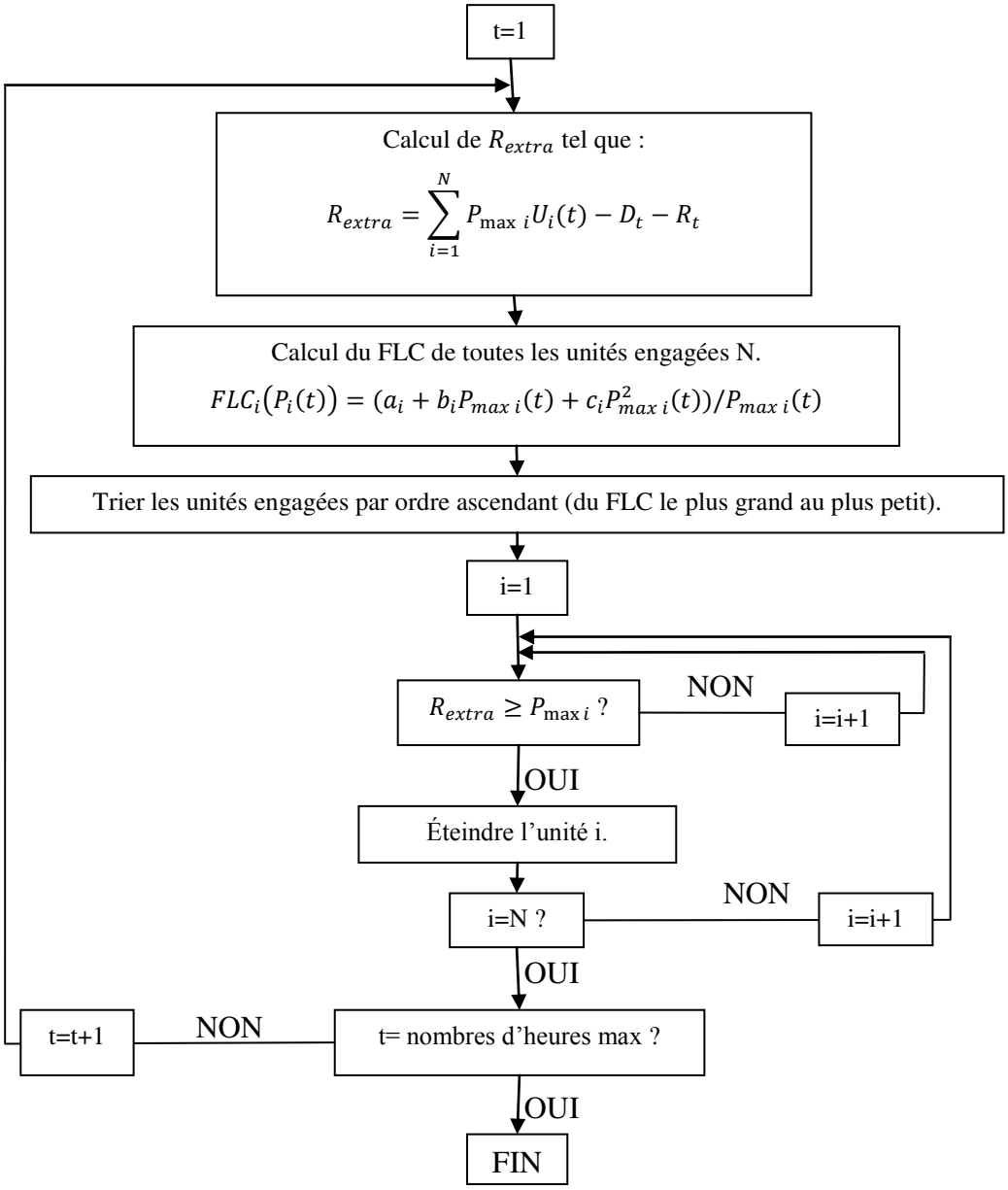



Figure 3. 3 Mécanisme d'extinction des unités excessives.

### e. Calcul des durées d'arrêt et d'allumage des unités :

Avant d'appliquer les contraintes de temps minimum d'arrêt et d'allumage, il est nécessaire de calculer le nombre d'heures durant lesquelles les unités sont opérationnelles ou éteintes. Ce calcul est fait à partir des équations suivantes :

$$ON_{i,t} = \begin{cases} ON_{i,t} + 1, & si\ U_{i,t} = 1 \\ 0, & si\ U_{i,t} = 0 \end{cases} \quad (3.\ 1)$$

$$OFF_{i,t} = \begin{cases} OFF_{i,t} + 1, & si\ U_{i,t} = 0 \\ 0, & si\ U_{i,t} = 1 \end{cases} \quad (3.\ 2)$$

$$Tel\ que\ i = 1, \dots, N\ et\ t = 1, \dots, T$$

**f. Contraintes de temps minimum d'arrêt et d'allumage :**

Cette étape consiste à réparer les infractions des contraintes de temps minimum d'arrêt et de redémarrage. Ce mécanisme est basé sur le fait que pendant les heures creuses, les contraintes de temps minimum d'arrêt et de redémarrage sont susceptibles d'être enfreintes [38]. Ce qui suit détaille ce mécanisme :

$$Si\ U_{i,t} = 0, U_{i,t-1} = 1\ et\ ON_{i,t} < MUT_i : \quad U_{i,t} = 1. \quad (3.\ 3)$$

$$Si\ U_{i,t} = 0, U_{i,t-1} = 1, OFF_{i,t} < MDT_i\ et\ (t - 1 + MDT_i) \leq T: \quad U_{i,t} = 1. \quad (3.\ 4)$$

$$Tel\ que\ i = 1, \dots, N\ et\ t = 1, \dots, T$$

À chaque modification, il est important de mettre à jour les durées d'arrêt et d'allumage des unités.

**g. Évaluation des membres de la population :**

À ce stade, il s'agit d'effectuer une répartition économique de tous les membres de la population afin de déterminer le coût de production total de chaque membre. La valeur d'aptitude ou *fitness value* correspond au coût total de toutes les unités à une heure précise.

**h. La sélection :**

La sélection consiste à créer de nouveaux individus (parents) à partir des individus actuels. Plusieurs méthodes de sélection existent : la roulette, la sélection par rang, la sélection par tournoi et l'élitisme. Cependant, la méthode utilisée dans la présente méthodologie est celle de sélection par tournoi. Cette dernière consiste à prendre au hasard des individus dans la population et de comparer les valeurs d'aptitudes. Ceux qui ont les valeurs d'aptitudes de meilleure qualité sont vainqueurs du tournoi et leurs données sont conservées.

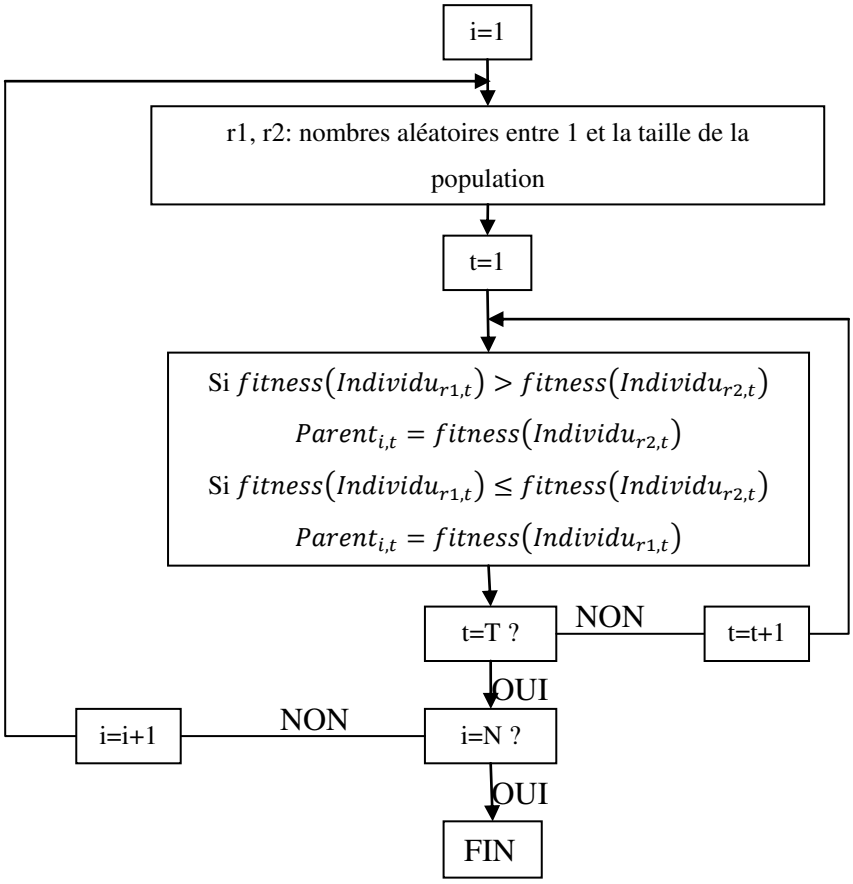



Figure 3. 4 Sélection par tournoi.

**i. Croisement :**

L'opérateur de croisement a pour but de recomposer les gènes des individus existant dans la population. Cette manipulation de la structure des chromosomes permet de diversifier l'espace de recherche.

Le principe de base est de générer deux enfants à partir de deux parents. Ce processus consiste à tirer aléatoirement une position dans les chaînes de bits des deux parents. Un échange des deux sous-chaînes produit deux enfants. Le découpage des chaînes ne peut s'effectuer que si une probabilité de croisement n'est pas dépassée (entre 0.5 et 1).

Plusieurs types de croisement existent : croisement à un point, croisement à multiples points, croisement en anneau…etc. L'exemple suivant explique le croisement à deux points sur deux chaînes de bits :

Exemple : Croisement aux points 3 et 7.

Parent 1 : 1111111111 Enfant 1 : 1110000111

Parent 2 : 0000000000 Enfant 2 : 0001111000

Ce processus, appliqué à l'UCP, est d'avantage expliqué dans la figure (3.5).

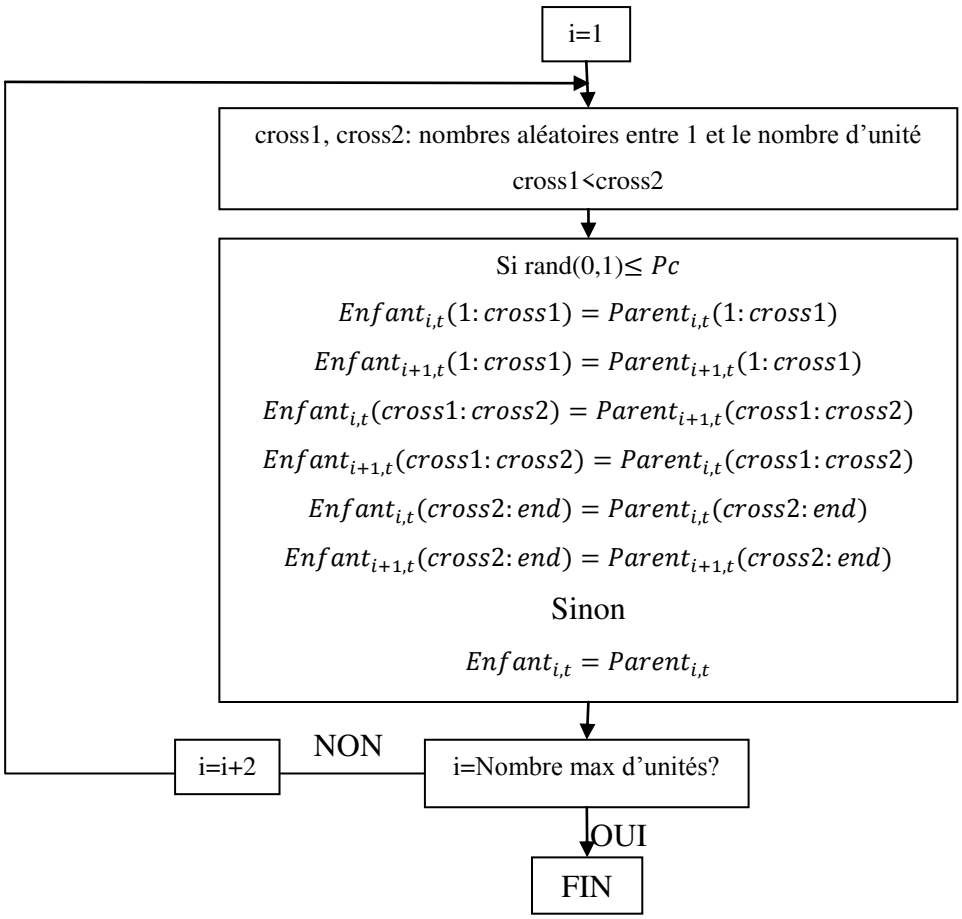



Figure 3. 5 Croisement à deux points.

**j. Mutation :**

L'opérateur de mutation a pour but de garantir l'exploration de l'espace de recherche. En utilisant la probabilité de mutation (entre 0.01 et 0.1), on tire aléatoirement un ou plusieurs bits du chromosome et on bascule leur valeur. L'exemple suivant explique ce processus :

Exemple : Mutation aux points 4 et 8.

Enfant : 1111111111 Mutant : 1110111011.

Cet opérateur permet donc de s'assurer que chaque solution dans l'espace de recherche a une chance d'être traitée. La mutation à deux points appliquée à l'UCP est d'avantage expliquée dans la figure (3.6).

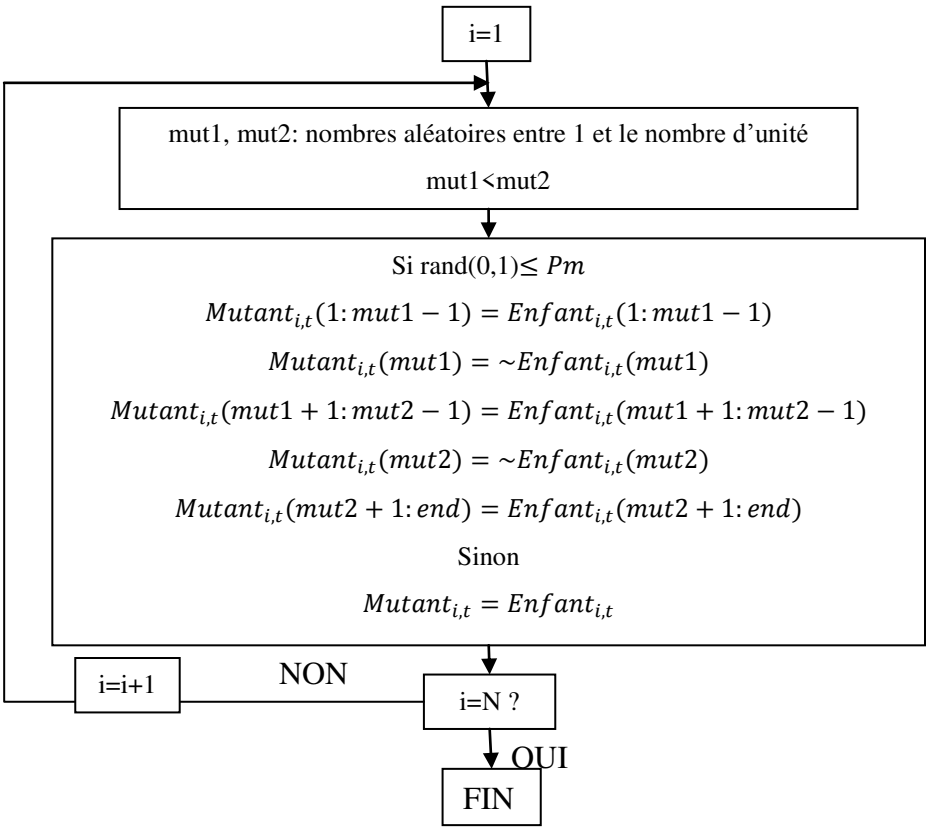



Figure 3. 6 Mutation à deux points.

**k. Critère d'arrêt :**

L'algorithme génétique est sous forme d'une boucle itérative. À chaque itération, une nouvelle population est générée à partir de l'ancienne population. Il est difficile d'estimer le nombre d'itérations sachant que les algorithmes génétiques ne garantissent pas une solution optimale à chaque essai. La nature aléatoire des opérateurs de croisement et de mutation ne permet pas de déterminer un critère d'arrêt efficace. Il se peut qu'il y ait une redondance de solution optimale sans pour autant atteindre l'optimum global. Effectuer plusieurs essais afin de déterminer le nombre d'itérations qui convient à la dimensionnalité du problème est une bonne initiative. Si le nombre d'itérations maximales est atteint, l'algorithme est arrêté, sinon on passe à l'étape **c**.

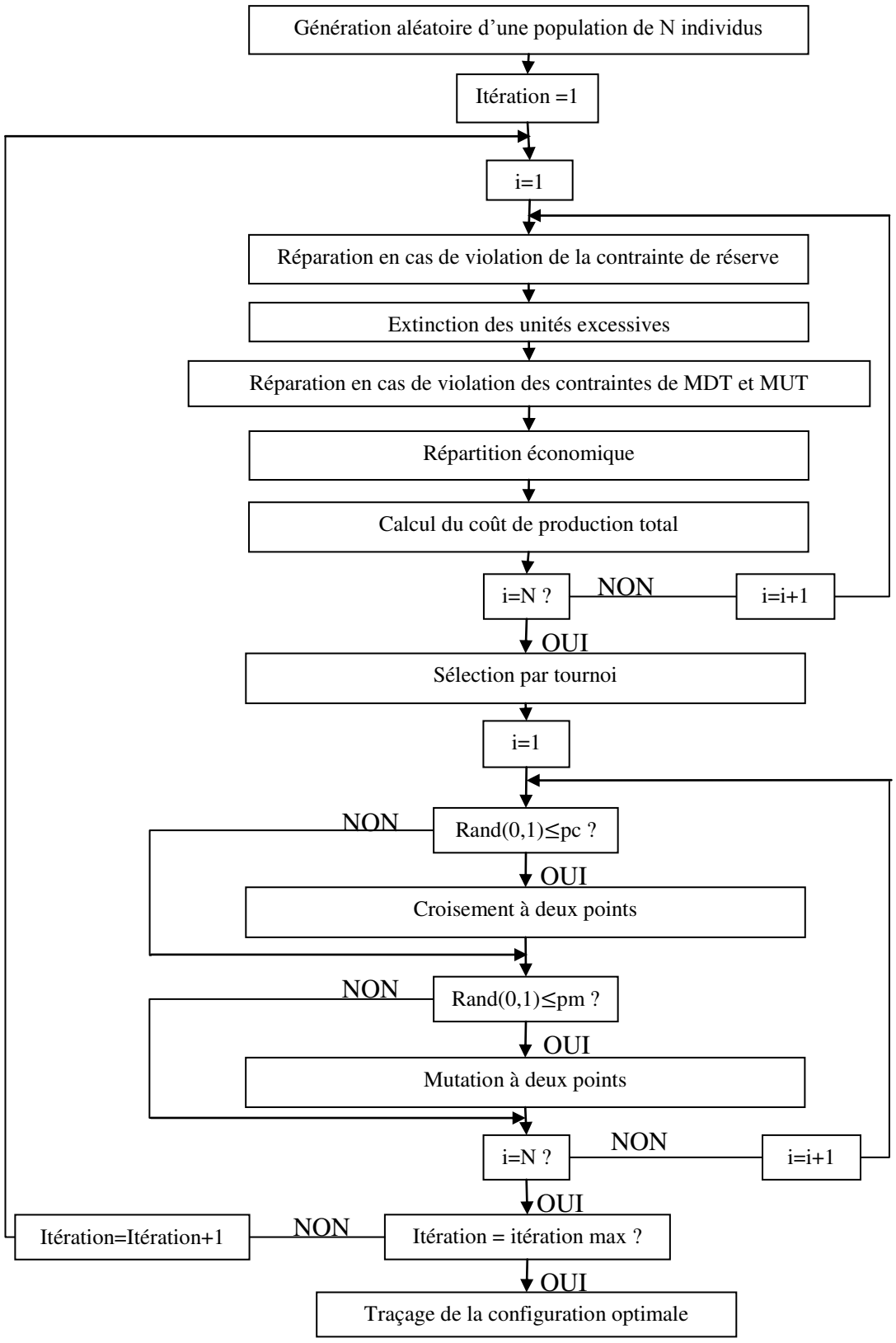



Figure 3. 7 Algorithme génétique pour la résolution de l'UCP.

# III.3. Optimisation par essaims particulaires :

## III.3.1. Introduction :

L'optimisation par essaims particulaires, introduite par James Kennedy et Russel Eberhart en 1995 [39], est une approche basée sur le comportement collectif des oiseaux à l'intérieur d'une nuée. Ainsi, son but est de simuler les mouvements synchrones et brusques d'un essaim d'oiseaux qui est capable de rester dans une formation optimale.

Le prédécesseur de l'optimisation par essaims particulaires est le programme informatique *Boids*, développé par Craig W.Reynolds en 1986, qui simule le comportement d'une nuée d'oiseaux. Chaque individu « *boid* » doit respecter des lois simples : éviter la collision avec un autre *boid*, se déplacer selon la vitesse moyenne du voisinage et ne pas quitter la formation en suivant le même chemin [40]. Kennedey et Eberhart ont ensuite modifié ces lois et ont proposé l'algorithme PSO (particle swarm optimization). Au fil des années, PSO a été d'avantage modifié, développé et implémenté avec succès dans le domaine de l'optimisation.

## III.3.2. L'algorithme de base de l'optimisation par essaims particulaires :

Dans un problème d'optimisation, un individu ou une particule i est représentée par une chaîne de données de dimension D. En premier lieu, une population (« essaim ») de N particules est initialisée avec des positions $x_i$ et des vitesses $v_i$ aléatoires. La fonction fitness $f(x_i)$ est évaluée pour chaque particule ce qui détermine la meilleure position trouvée par l'essaim (Gbest) et la meilleure position propre à chaque particule (Pbest). À chaque itération (« mouvement »), les positions et les vitesses de chaque particule sont mis à jours.

### III.3.2.1. L'approche classique :

L'algorithme classique proposé par Kennedy et Eberhart [39] a été conçu pour résoudre les problèmes d'optimisation continue et peut être implémenté en quelques lignes. Les équations mathématiques qui la décrivent sont les suivantes :

$$v_i^{k+1} = \overbrace{w v_i^k}^{\text{Mouvement actuel}} + \underbrace{C_1 rand\left(P_{best\,i}^k - x_i^k\right)}_{\text{Influence personnelle}} + \underbrace{C_2 rand\left(G_{best}^k - x_i^k\right)}_{\text{Influence sociale}} \quad (3.5)$$

$$\underbrace{x_i^{k+1} = x_i^k + v_i^{k+1}}_{\text{Nouvelle position}} \quad (3.6)$$

$$k = 1, \dots, itération\ \max \qquad i = 1, \dots, nombre\ de\ particules$$

L'équation (3.5) consiste à mettre à jour les vitesses des particules et inclut des paramètres aléatoires représentés par *rand* (tiré aléatoirement entre 0 et 1) et de trois accélérations pondérées : *w* (coefficient d'inertie), $C_1$ (paramètre cognitif) et $C_2$ (paramètre social). L'étape suivante consiste à mettre à jour les positions des particules à partir de l'équation (3.6).

Le déplacement d'une particule est influencé par les trois composantes suivantes (figure 3.8) [41]:

- Une composante physique : la particule tend à suivre sa direction courante de déplacement.
- Une composante cognitive : la particule tend à se diriger vers le meilleur site par lequel elle est déjà passée.
- Une composante sociale : la particule tend à se diriger vers le meilleur site déjà atteint par ses voisins.

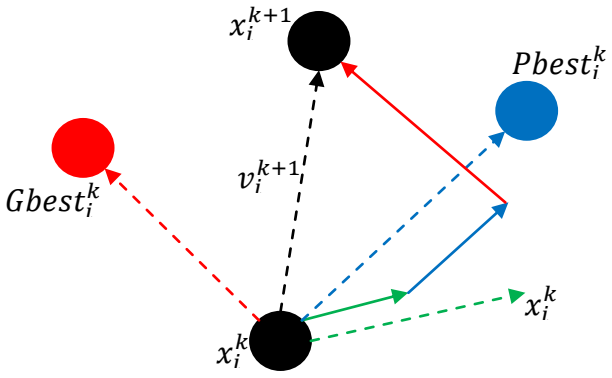



Figure 3. 8 Déplacement d'une particule [41].

### III.3.2.2. L'approche binaire :

L'approche de base du PSO est intrinsèquement continue et par conséquent n'est pas convenable pour résoudre des problèmes d'optimisation dont les variables de contrôle sont discrètes ou binaires. Cependant plusieurs problèmes d'optimisation sont binaires ce qui a poussé Kennedy et Eberhart à proposer une autre variante de l'algorithme [42]. Afin de transformer les variables continues fournies par PSO à l'intervalle [0,1], la fonction sigmoïde est utilisée :

$$S\left(v_i^{k+1}\right) = \frac{1}{1+e^{-v_i^{k+1}}} \tag{3. 7}$$

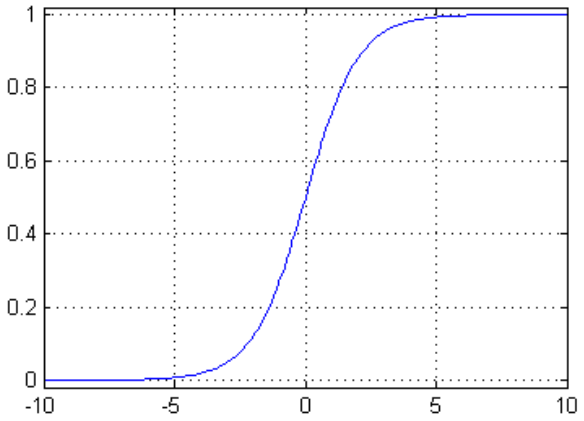



Figure 3. 9 Fonction sigmoïde.

La mise à jour des vitesses des particules est faite à partir de l'équation (3.5), tandis que l'équation de mise à jour des positions (3.6) est remplacée par l'opération suivante :

$$x_i^{k+1} = \begin{cases} 1 & si\ S\left(v_i^{k+1}\right) > rand(0,1) \\ 0 & si\ S\left(v_i^{k+1}\right) \leq rand(0,1) \end{cases} \tag{3. 8}$$

Les valeurs attribuées aux paramètres $C_1$, $C_2$ et $w$ dépendent du problème et de sa dimension. Cependant, il est possible d'attribuer des limites aux coefficients d'inertie et d'accélération et de les ajuster au cours de la recherche de la solution optimale [43]. Le coefficient d'inertie est linéairement réduit à chaque itération par le biais de l'équation suivante :

$$w^k = w_{max} - \frac{w_{max} - w_{min}}{kmax} k \tag{3. 9}$$

De même, les coefficients d'accélération varient en fonction des itérations afin de contrôler le processus de recherche avec efficacité. Un grand composant cognitif et un petit

composant social au début de la recherche permet de bien explorer l'espace de recherche et d'éviter la convergence vers le meilleur candidat prématurément. Un petit composant cognitif et un grand composant social à la fin de la recherche permet de converger vers l'optimum global.

$$C_1^k = \left(C_1^{fin} - C_1^{ini}\right)\left(\frac{k}{kmax}\right) + C_1^{ini} \quad , C_1^{fin} < C_1^{ini} \tag{3. 10}$$

$$C_2^k = \left(C_2^{fin} - C_2^{ini}\right)\left(\frac{k}{kmax}\right) + C_2^{ini} \quad , C_2^{fin} > C_2^{ini} \tag{3. 11}$$

Le pseudo-code du PSO binaire est présenté ci-dessous :

**Algorithme de l'optimisation binaire par essaims particulaires :**

Initialisation de la population de taille L : $p = (p_1, p_2 \dots, p_L)$ et du nombre d'itérations N.

Codage binaire de la population (représentation sous forme de chromosomes).

Initialisation des vitesses et des positions $(v, x) = (v_1, x_1; v_2, x_2; \dots; v_L, x_L)$

Initialisation des coefficients d'inertie *w* et d'accélération $c_1$ et $c_2$.

Calcul de la fonction coût *f(p) :=(* $f(p_1), f(p_2) \dots, f(p_L)$*).*

Pour chaque particule j : $P_{best\ j} = f(p_j)$ et déterminer la meilleure particule $G_{best}$.

*i* :=1

Tant que *i<N*

{

Mettre à jour des vitesses et positions à partir des équations (3. 5) et (3. 8).

Calcul des fonctions coût *f(p).*

Mettre à jour $P_{best\ j}$ pour chaque particule j et $G_{best}$.

}

Fin

*i :=i+1*

Fin

### III.3.3. Méthodologie de l'optimisation binaire par essaims particulaires pour la résolution de l'UCP :

La version classique de l'optimisation par essaims particulaires n'est pas convenable pour la résolution du problème d'engagement des turbines car les particules de la population sont représentées par des matrices de bits de 1 et 0. Ainsi, la version binaire de l'algorithme est beaucoup plus adaptée à ce type de problème. La structure de chaque particule $P_i$ est définie par les différents éléments suivants :

- Matrice binaire $X_i$ de dimension *NxH* qui représente la matrice des positions courantes.
- Matrice réel $V_i$ de dimension *NxH* qui représente la matrice des vitesses de déplacement.
- Un coût de production total *fitness* $F_T$.
- $P_{best}^k$ qui correspond à la meilleure configuration obtenue par la particule à l'itération k.
- $G_{best}^k$ qui correspond à la meilleure configuration trouvée par l'essaim, c'est-à-dire l'ensemble des particules.

Les mêmes mécanismes de réparation utilisés précédemment pour les algorithmes génétiques sont appliqués aux particules en cas de violations des contraintes. L'algorithme PSO binaire pour l'UCP est présenté dans la figure ci-dessous.

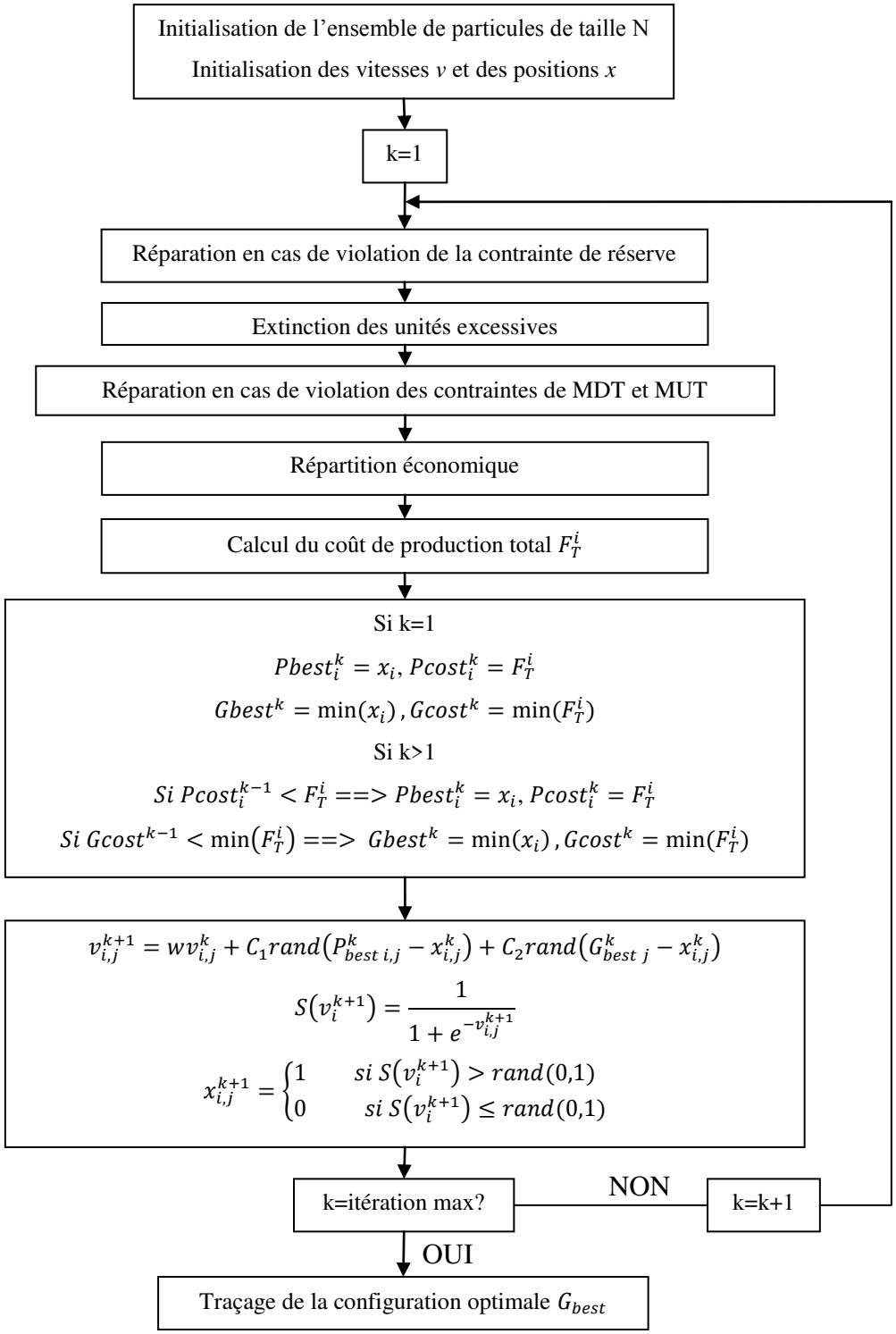



Figure 3. 10 Méthodologie de l'optimisation par essaims particulaires binaire

## III.3.4. L'approche hybride de l'optimisation binaire par essaims particulaires pour la résolution de l'UCP :

L'optimisation binaire par essaims particulaires, bien que très utilisée, n'est pas forcément satisfaisante, et cela dépend de la complexité du problème. En effet, le concept de base du PSO qui consiste à diriger les particules vers la meilleure particule n'est pas complètement présent dans le PSO binaire. De plus, l'algorithme tend à explorer inutilement tous points de l'espace de recherche, ce qui fait augmenter le nombre d'itérations nécessaire pour arriver à la solution optimale [44].

Plusieurs variantes ont été proposées afin d'améliorer la méthode PSO binaire, notamment une hybridation avec d'autres méthodes d'optimisation. L'une des hybridations possibles consiste à combiner les deux concepts de PSO et GA. Tandis qu'une autre hybridation envisageable est d'appliquer une recherche locale sur le meilleur élément de la population.

### III.3.3.1. Hybridation avec les algorithmes génétiques :

L'hybridation du PSO avec les algorithmes génétiques proposée par *Menhas* [44] est une hybridation de haut niveau co-évolutionnaire, c'est-à-dire que deux métaheuristiques coopèrent pour trouver la solution optimale du problème. Afin de mieux exploiter les concepts sociaux et cognitifs du PSO classique, un croisement stochastique et intelligent est incorporé à l'algorithme de base. Le croisement consiste à partager les données entre la meilleure particule et l'ensemble des particules selon une probabilité.

$$x_i^{k+1} = \begin{cases} x_i^{k+1} & si\ 0 \le R \le \alpha \\ Pbest_i^k & si\ \alpha < R \le 2\alpha \\ Gbest_i^k & si\ 2\alpha < R \le 1 \end{cases} \quad (3.\ 12)$$

Dans l'équation (3. 12), R représente une probabilité uniforme entre 0 et 1, tandis que $\alpha$ représente un pourcentage prédéfini de probabilité de croisement. La possibilité d'un bit d'une particule de rester dans son état est de 16.66%, tandis que la possibilité d'être remplacé par le bit du record personnel de la particule est de 33,33%. Enfin, la possibilité d'être remplacé par le bit de la meilleure particule de l'essaim est de 50%.

Quant à l'opérateur de mutation, il permet de maintenir la diversité de la population. Dans l'équation (3. 13), un nombre aléatoire, générée entre 10 et 1, est comparée à une probabilité de mutation prédéfinie afin d'effectuer la mutation d'un bit.

$$x_i^{k+1} = \begin{cases} \overline{x_i^{k+1}} \; si \; R \leq p_m \\ x_i^{k+1} \quad si \; R > p_m \end{cases} \tag{3. 13}$$

**III.3.3.2. Hybridation avec la recherche locale élitiste 2-opt :**

L'hybridation entre les algorithmes à population et les algorithmes de recherche locale est connue sous le nom d'algorithmes mémétiques [45]. Afin d'améliorer la convergence et diminuer le nombre d'itérations nécessaires pour arriver à la solution optimale, une hybridation entre l'optimisation par essaims particulaires et un algorithme basée sur la recherche locale 2-opt est proposée.

L'algorithme 2-opt, proposé par George A. Croes en 1958, est un algorithme de recherche locale qui consiste à améliorer la solution initiale dans le but de résoudre le problème du voyageur de commerce. Le principe étant applicable au problème de l'UCP, une approche élitiste et binaire du 2-opt est combinée au PSO binaire.

À chaque itération, l'opérateur 2-opt est appliqué à la matrice de positions $X_i$ de la meilleure particule. La recherche consiste à commuter les états de deux unités sélectionnées aléatoirement à chaque heure. Ce processus est répété jusqu'à ce que le coût de production total soit inférieur à celui de la meilleure particule où bien jusqu'à ce qu'un critère d'arrêt soit satisfait. Étant donné que la sélection des unités est aléatoire, il est possible de retomber sur une particule modifiée déjà visitée. Afin d'éviter cela, on enregistre les stratégies de commutation et on évite l'évaluation d'une particule soumise à une commutation déjà effectuée auparavant. Ceci permettra à l'algorithme PSO de converger en moins d'itérations et de visiter plus de solutions possibles dans l'espace de recherche.

La recherche locale 2-opt peut avoir comme critère d'arrêt une limite en durée. Cependant, une autre alternative envisageable est d'arrêter la recherche lorsque la meilleure solution trouvée par PSO n'a pas encore été améliorée depuis un nombre prédéterminé d'itérations.

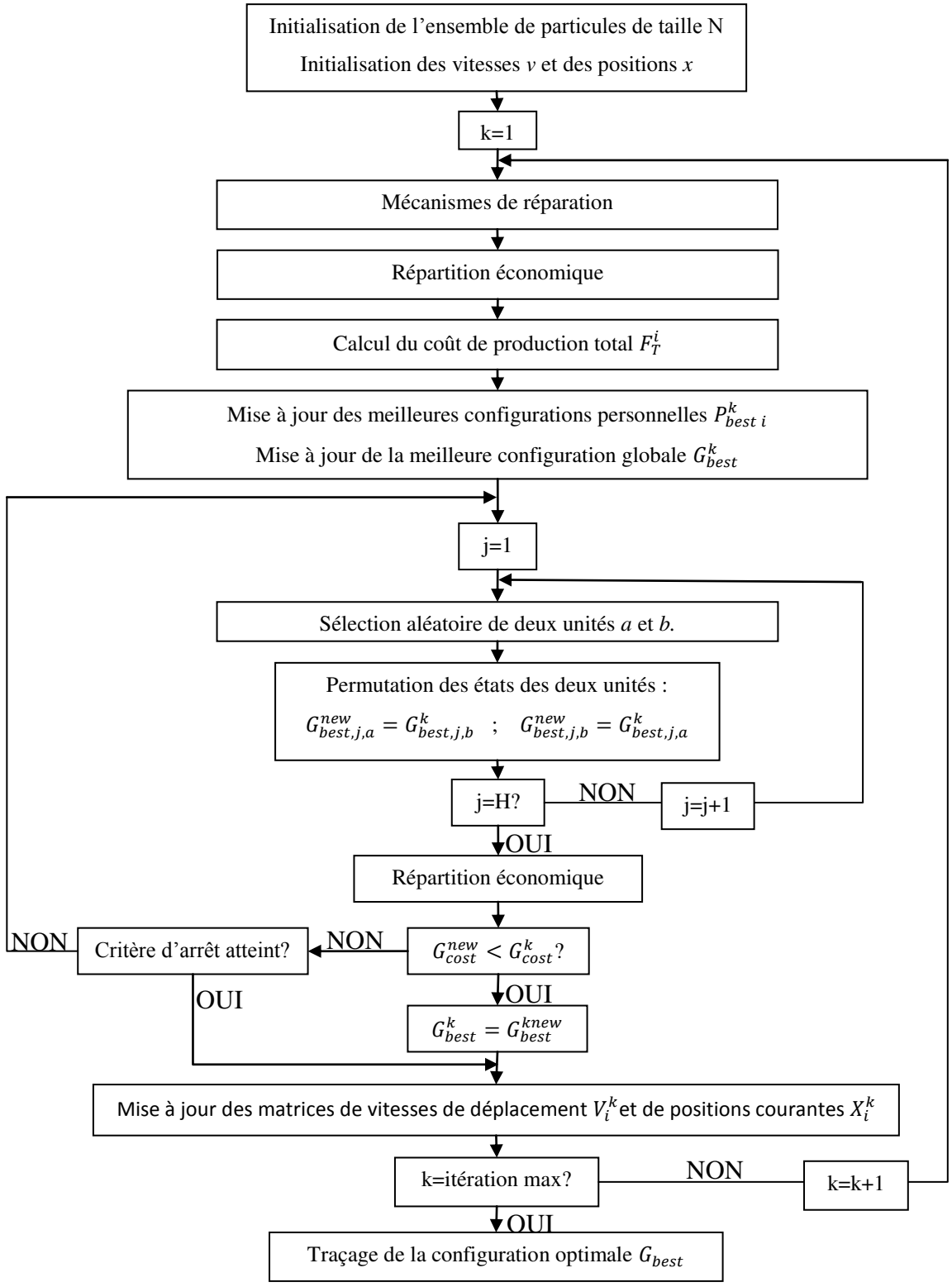



Figure 3. 11 Méthodologie de l'algorithme de l'optimisation binaire par essaims particulaires et recherche locale.

# III.4. Recuit simulé :

## III.4.1. Introduction :

L'algorithme du recuit simulé s'appuie sur l'algorithme de Metropolis-Hastings [46] qui simule le processus d'aboutir à un équilibre thermique à une température fixe. Cet algorithme a été généralisé par trois chercheurs de la société IBM, S. Kirkpatrick, C.D. Gelatt et M.P. Vecchi en 1983 [11] afin de décrire l'évolution d'un système thermodynamique. Cependant, plusieurs autres chercheurs ont contribué indépendamment au développement du recuit simulé tels que M. Pincus en 1970 [47] et V. Cerny en 1982 [48].

Le principe de cette méthode est basé sur le phénomène thermodynamique du recuit qui consiste à refroidir un matériau solide préalablement chauffé le plus lentement possible pour obtenir une structure moléculaire optimale. En faisant une analogie entre le processus du recuit et l'optimisation, un grand nombre de problèmes d'optimisation combinatoires peuvent être résolu en suivant la même procédure de transition d'un point d'équilibre à un autre. Cette analogie est indiquée par :

- Les solutions du problème d'optimisation sont équivalentes aux configurations du métal.
- Un paramètre de contrôle $T$ est équivalent à la température du processus du recuit.
- La fonction coût d'une solution est équivalente à l'énergie d'une configuration.

Le recuit simulé est donc une méthode de recherche locale à solution unique similaire au *hill climbing* cependant cette méthode a la particularité d'accepter des solutions de moindre qualité et cela selon une probabilité. Cette technique d'optimisation est particulièrement utile lorsqu'on fait face à des problèmes combinatoires complexes binaires car le principe de perturbation de la solution est facilement implémentable.

## III.4.2. Méthodologie générale du recuit simulé:

L'algorithme du recuit simulé a l'avantage d'être à solution unique, c'est-à-dire qu'à l'instar des méthodes d'optimisation à population, l'espace de recherche est parcouru en perturbant la solution initiale et en la remplaçant en cas d'amélioration, ce qui permettra de réduire le temps de calcul. Les différents paramètres décrivant l'algorithme sont les suivants :

- La température initiale $T_0$.
- La constante de refroidissement *C*.
- Tolérance $\varepsilon$.

La température $T$, étant un paramètre de contrôle fixé au début de l'algorithme, décroît linéairement au cours de l'exécution de l'algorithme. La détermination des paramètres n'est pas évidente car cela dépend du type et de la dimension du problème. De plus, un mauvais choix des paramètres peut influencer considérablement la convergence et la qualité de la solution, ce qui n'est pas forcément le cas pour d'autres méthodes métaheuristiques. Ainsi, il est important de prédéterminer les paramètres par essai et erreur.

Au début, une solution initiale est générée aléatoirement et à chaque itération, des perturbations sont effectuées sur la solution actuelle. La solution modifiée est acceptée si elle est de meilleure qualité que la solution actuelle. Cependant, il se peut qu'elle soit acceptée quelque soit sa qualité si la condition suivante est satisfaite :

$$e^{-\Delta/T} > r \qquad (3.\ 14)$$

Tel que *r* est un nombre aléatoire dans l'intervalle [0,1] et $\Delta$ est la différence entre la valeur de la fonction coût de la solution actuelle et celle de la solution modifiée. Le pseudo-code du recuit simulé est comme suit :

**Algorithme recuit simulé :**

Initialisation de la température $T := T_0$ et choix aléatoire de la solution initiale $x := x_0$

Calcul de la fonction coût initiale $f(x) := f(x_0)$

Initialisation du taux de refroidissement *C,* de la tolérance $\varepsilon$ et du nombre d'itérations *N*.

*i* :=1

Tant que $T > \varepsilon$ et $i < N$

*v* :=voisin*(x)*

Calcul de $\Delta = f(v) - f(x)$

Si $\Delta < 0$

Accepter la nouvelle solution *x :=v ; f(x)=f(v)*

Sinon

{Génération d'un nombre aléatoire *r* entre 0 et 1.

Si $e^{-\Delta/T} > r$

Accepter la nouvelle solution *x :=v ; f(x)=f(v)* }

Décrémentation *T :=T*C*

*i :=i+1*

## III.4.3. Méthodologie du recuit simulé pour la résolution de l'UCP :

La résolution du problème d'engagement des turbines par recherche locale peut s'avérer être difficile car la recherche d'un minimum global situé dans un large espace de recherche en un temps de calcul assez grand n'est pas pratique. Cependant, la méthode du recuit simulé proposée permet de non seulement d'enregistrer les stratégies de perturbations sur la solution en cours afin d'éviter la redondance, mais d'également bien explorer l'espace de recherche à chaque itération. Les différentes étapes qui constituent cette méthode sont :

**Étape 1** : Générer une solution initiale $U_i(h,n)$ faisable tel que *h* et *n* représentent respectivement le nombre d'heures et le nombre d'unités. La solution initiale peut être générée par d'autres méthodes déterministes telles que la liste de priorité. Bien que la méthode de liste de priorité ne respecte pas les contraintes de temps minimal d'arrêt et de redémarrage, elle donne une solution initiale qui correspond au profil de la demande prévisionnelle, ce qui n'est pas le cas lors d'une génération aléatoire. Ainsi, on se situe

mieux dans l'espace de faisabilité. Un exemple d'une solution initiale générée par liste de priorité est illustré ci-dessous :

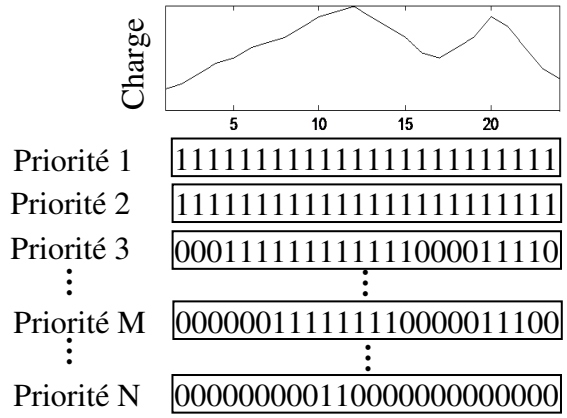



Figure 3. 12 Solution initiale $U_i$ par liste de priorité [49].

**Étape 2** : Initialiser le compteur des itérations à K=0, la température initiale $T_i$.

**Étape 3** : Si le critère d'équilibre, $T < \varepsilon$ tel que $T = T_i$, n'est pas satisfait, passer à l'étape 4, sinon passer à l'étape 12.

**Étape 4** : Effectuer une perturbation de la solution actuelle qui consiste à permuter l'état de deux unités sélectionnées aléatoirement à chaque heure. L'algorithme de permutation est décrit comme suit :

```
Tant que t<nombre d'heures
Sélectionner aléatoirement deux unités a et b :
a=rand(1:n) ; b=rand(1:n) ;
Permuter les états des deux unités :
temp=U(t,a) ;
U(t,a)=U(t,b) ;
U(t,b)=temp ;
t=t+1 ;
Fin.
```

**Étape 5** : Enregistrer la stratégie de permutation. Si cette stratégie a déjà été effectuée auparavant, revenir à l'étape 4, sinon passez à l'étape 6.

**Étape 6**: Engager des unités en cas de non respect de la contrainte de puissance et de réserve.

**Étape 7** : Extinction des unités excessives.

**Étape 8** : Réparation en cas d'infraction des contraintes de temps minimal d'arrêt et de redémarrage.

**Étape 9** : Effectuer une répartition économique de la solution actuelle et déterminer le coût de production total de la solution actuelle $F_T^K$.

**Étape 10** : Effectuer le test suivant :

Si $F_T^K < F_T^{K-1}$ : accepter la solution et passer à l'étape.

Sinon si $e^{\frac{F_T^{K-1} - F_T^K}{T}} < rand(0,1)$ : accepter la solution actuelle et passer à l'étape.

Sinon rejeter la solution actuelle et passer à l'étape 3.

**Étape 11 :** Diminuer la valeur du paramètre de contrôle *T=Cool*T,* sachant que *Cool* est un paramètre fixe déterminé au préalable.

**Étape 12 :** Diminuer la valeur du paramètre de contrôle initial $T_i = T_i - \frac{T_i}{K}$.

**Étape 13 :** Mise à jour du nombre d'itérations : K=K+1. Si le nombre d'itérations maximale est atteint, passer à l'étape 14 sinon passer à l'étape 3.

**Étape 14 :** Traçage de la planification de la meilleure solution trouvée.

# CHAPITRE IV :

# Simulations, résultats et interprétations

# IV.1. Introduction :

Dans une première partie, une description globale du logiciel UC-Plan.0.0.1 est présentée à travers une visualisation des différents éléments le constituant. Ce logiciel, qui synthétise toutes les méthodes expliquées dans ce mémoire, est un environnement dans lequel les simulations sont effectuées sur une multitude de systèmes. Ces tests ont été faits sur un PC Intel® Core™ i5-4210U CPU @ 1.70GHz 2.40 Ghz et 4,00 Go de RAM.

Dans une deuxième partie, des programmes de résolution du problème d'engagement des turbines ont été élaborés et implémentés dans le logiciel en se basant sur les formulations des méthodes déterministes (liste de priorité et programmation dynamique) et des méthodes avancées (algorithme génétique, optimisation par essaims particulaires et recuit simulé). L'optimalité, l'efficacité et la faisabilité des méthodes ont été constatés à partir des validations effectuées sur différents systèmes de différentes tailles. Les résultats de simulation ont été comparés afin de voir l'apport de chaque méthode.

# IV.2. Logiciel de simulation UC-Plan.0.0.1:

Le logiciel présenté dans ce mémoire a été développé afin de faciliter l'interaction entre les différents modules qui le constituent, notamment : entrée/sortie, interface graphique, engagement des turbines et répartition économique.

L'intérêt de développer un tel logiciel est de faciliter l'étude du problème de l'UCP. En effet, chaque méthode de résolution est différente de l'autre et il est difficile d'évaluer leur performance d'une manière cohérente. Grace au logiciel, il est possible de tester n'importe quelle méthode sur une multitude de systèmes d'essai tout en visualisant en temps réel la procédure de recherche. Le fait de pouvoir enregistrer les résultats de simulation facilite leur exploitation.

L'organigramme du logiciel est illustré dans la figure ci-contre.

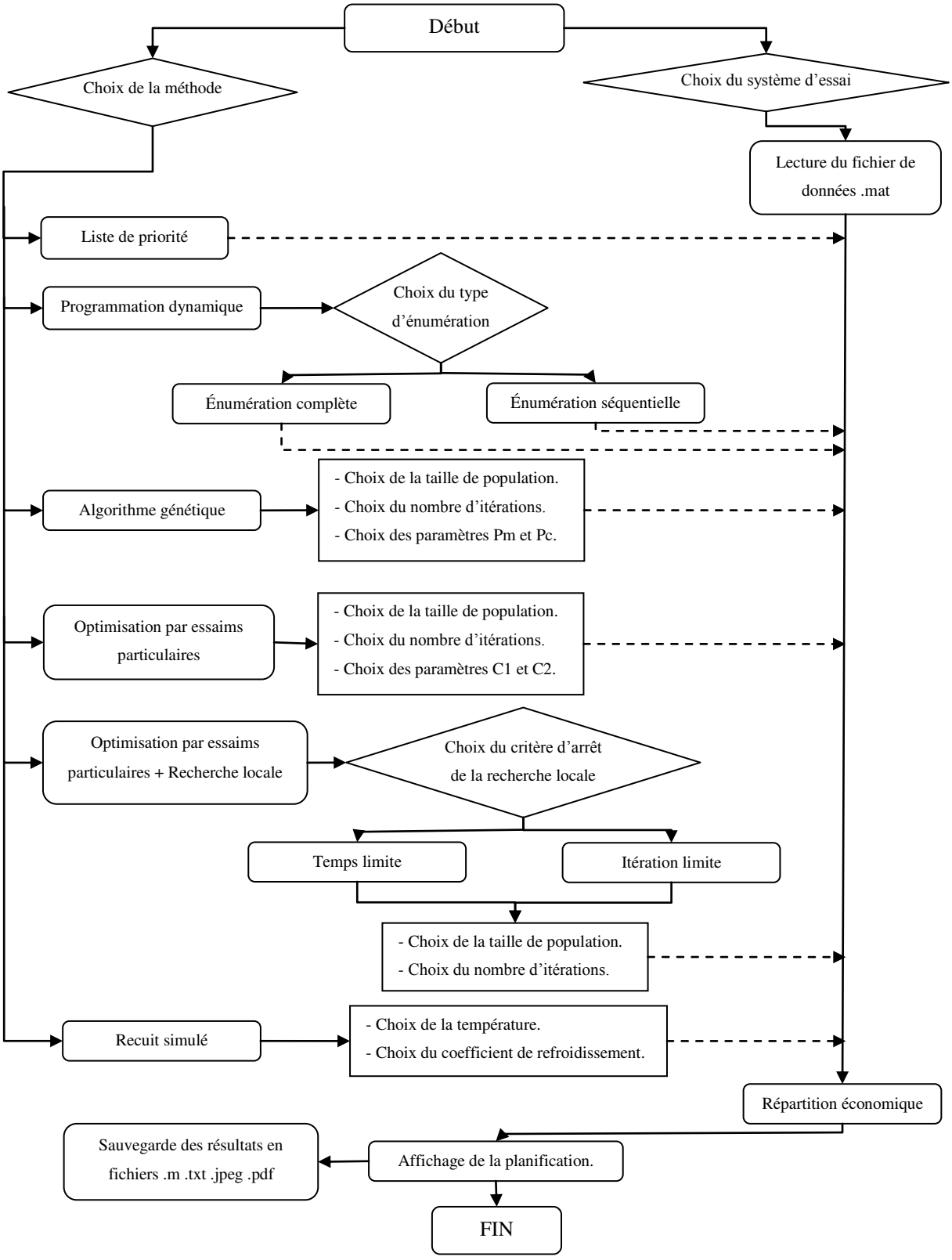
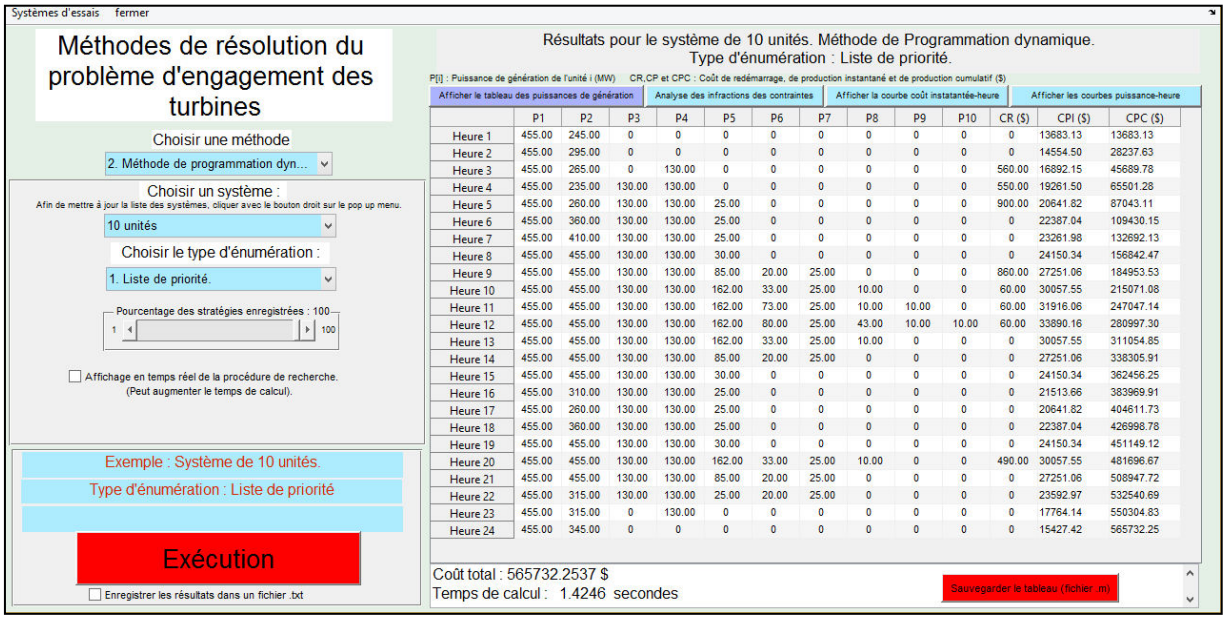



Figure 4. 1 Organigramme général du logiciel

## IV.2.1. Interface graphique :

L'interface graphique a été conçue afin de donner à l'utilisateur la possibilité d'interagir avec les différents éléments du logiciel. Ainsi, il est possible de modifier et d'adapter les paramètres d'entrée à différentes simulations, d'enregistrer les données de sortie sous plusieurs types de formats, d'ajouter des données d'entrée et de visualiser les résultats.




| | P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 | P9 | P10 | CR ($) | CPI ($) | CPC ($) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Heure 1 | 455.00 | 245.00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13683.13 | 13683.13 |
| Heure 2 | 455.00 | 295.00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 14554.50 | 28237.63 |
| Heure 3 | 455.00 | 265.00 | 0 | 130.00 | 0 | 0 | 0 | 0 | 0 | 0 | 560.00 | 16892.15 | 45689.78 |
| Heure 4 | 455.00 | 235.00 | 130.00 | 130.00 | 0 | 0 | 0 | 0 | 0 | 0 | 550.00 | 19261.50 | 65501.28 |
| Heure 5 | 455.00 | 260.00 | 130.00 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 900.00 | 20641.82 | 87043.11 |
| Heure 6 | 455.00 | 360.00 | 130.00 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 0 | 22387.04 | 109430.15 |
| Heure 7 | 455.00 | 410.00 | 130.00 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 0 | 23261.98 | 132692.13 |
| Heure 8 | 455.00 | 455.00 | 130.00 | 130.00 | 30.00 | 0 | 0 | 0 | 0 | 0 | 0 | 24150.34 | 156842.47 |
| Heure 9 | 455.00 | 455.00 | 130.00 | 130.00 | 85.00 | 20.00 | 25.00 | 0 | 0 | 0 | 860.00 | 27251.06 | 184953.53 |
| Heure 10 | 455.00 | 455.00 | 130.00 | 130.00 | 162.00 | 33.00 | 25.00 | 10.00 | 0 | 0 | 60.00 | 30057.55 | 215071.08 |
| Heure 11 | 455.00 | 455.00 | 130.00 | 130.00 | 162.00 | 73.00 | 25.00 | 10.00 | 10.00 | 0 | 60.00 | 31916.06 | 247047.14 |
| Heure 12 | 455.00 | 455.00 | 130.00 | 130.00 | 162.00 | 80.00 | 25.00 | 43.00 | 10.00 | 10.00 | 60.00 | 33890.16 | 280997.30 |
| Heure 13 | 455.00 | 455.00 | 130.00 | 130.00 | 162.00 | 33.00 | 25.00 | 10.00 | 0 | 0 | 0 | 30057.55 | 311054.85 |
| Heure 14 | 455.00 | 455.00 | 130.00 | 130.00 | 85.00 | 20.00 | 25.00 | 0 | 0 | 0 | 0 | 27251.06 | 338305.91 |
| Heure 15 | 455.00 | 455.00 | 130.00 | 130.00 | 30.00 | 0 | 0 | 0 | 0 | 0 | 0 | 24150.34 | 362456.25 |
| Heure 16 | 455.00 | 310.00 | 130.00 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 0 | 21513.66 | 383969.91 |
| Heure 17 | 455.00 | 260.00 | 130.00 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 0 | 20641.82 | 404611.73 |
| Heure 18 | 455.00 | 360.00 | 130.00 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 0 | 22387.04 | 426998.78 |
| Heure 19 | 455.00 | 455.00 | 130.00 | 130.00 | 30.00 | 0 | 0 | 0 | 0 | 0 | 0 | 24150.34 | 451149.12 |
| Heure 20 | 455.00 | 455.00 | 130.00 | 130.00 | 162.00 | 33.00 | 25.00 | 10.00 | 0 | 0 | 490.00 | 30057.55 | 481696.67 |
| Heure 21 | 455.00 | 455.00 | 130.00 | 130.00 | 85.00 | 20.00 | 25.00 | 0 | 0 | 0 | 0 | 27251.06 | 508947.72 |
| Heure 22 | 455.00 | 315.00 | 130.00 | 130.00 | 25.00 | 20.00 | 25.00 | 0 | 0 | 0 | 0 | 23592.97 | 532540.69 |
| Heure 23 | 455.00 | 315.00 | 0 | 130.00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 17764.14 | 550304.83 |
| Heure 24 | 455.00 | 345.00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15427.42 | 565732.25 |

Figure 4. 2 Interface graphique du logiciel.

À partir de l'interface graphique, l'utilisateur a la possibilité de :

- Sélectionner les systèmes d'essai et les méthodes de résolution afin d'effectuer les simulations.
- Visualiser la planification optimale obtenue par répartition économique dans un tableau.
- Visualiser les différentes courbes de charge, de convergence, d'infraction des contraintes …etc.
- Modifier, ajouter, supprimer ou réinitialiser les systèmes d'essai (fichiers .mat).
- Sauvegarder les résultats de simulation sous les formats suivants : pdf, matlab (.m), texte (.txt) et image (.jpeg et .tiff).

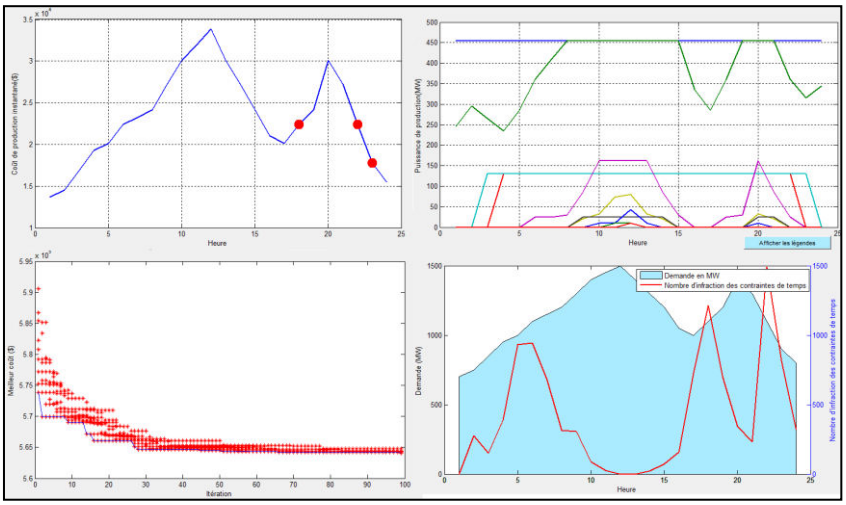



Figure 4. 3 Différentes courbes d'affichage des résultats.

## IV.2.2. Entrée/Sortie (I/O) :

Le module entrée/sortie (input/output) interagit avec les fichiers externes des données d'entrée et de sortie. L'onglet des systèmes d'essai permet de visualiser les données des générateurs, la demande prévisionnelle et la réserve tournante de chaque système, tout comme il est possible d'ajouter ou supprimer des systèmes. Les données sont enregistrées sous forme d'un fichier .mat dans un sous dossier, tandis que les résultats le sont dans un autre sous dossier. Ces fichiers peuvent être sélectionnés à partir d'un menu déroulant pour une éventuelle simulation.

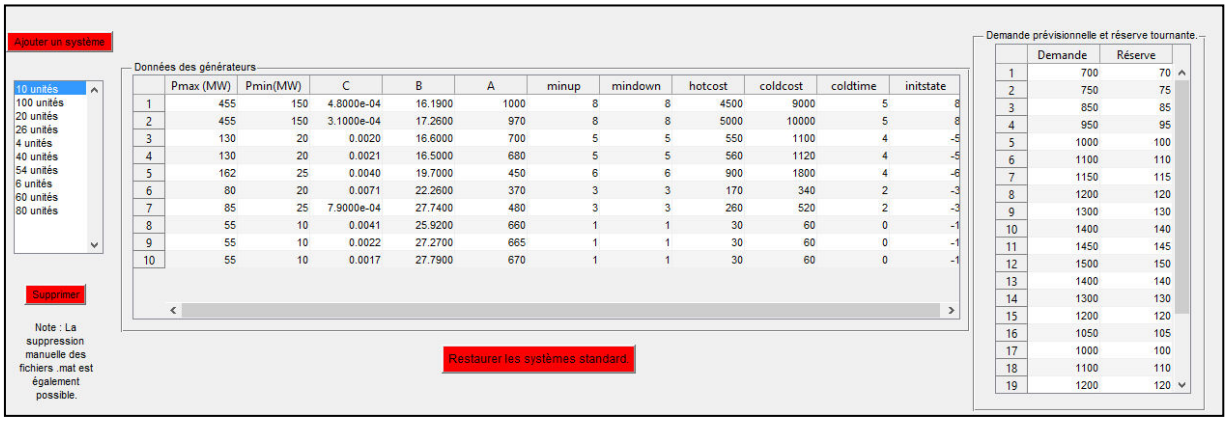



Figure 4. 4 Onglet des systèmes d'essai.

### IV.2.3. Interface de saisie des paramètres de simulation :

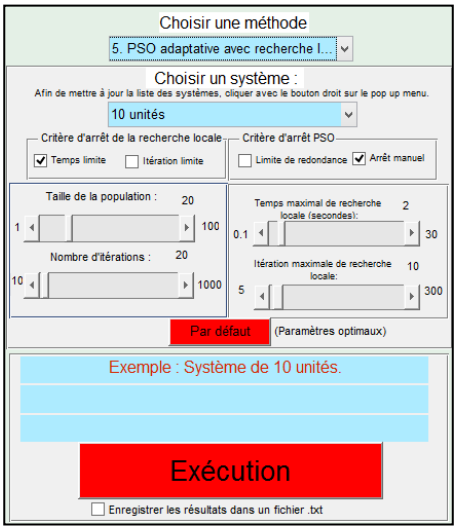



Figure 4. 5 Interface de saisie.

Plusieurs menus pop-up, curseurs et autres types d'éléments de sélection participent à la flexibilité et à la facilité d'utilisation du logiciel. Ces éléments constituants cette interface sont :

- Le menu pop-up du choix des méthodes permet d'afficher les différents paramètres modifiables caractérisant la méthode. La sélection d'une méthode implique l'affichage du reste des composants de saisie.

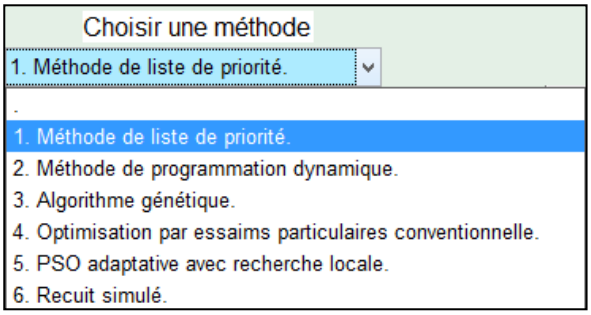



Figure 4. 6 Choix d'une méthode.

- Le menu pop-up du choix des systèmes d'essai permet de charger les données des générateurs et la demande prévisionnelle du système sélectionné. Ces données sont enregistrées sous formes de fichiers .mat dans un sous-dossier.

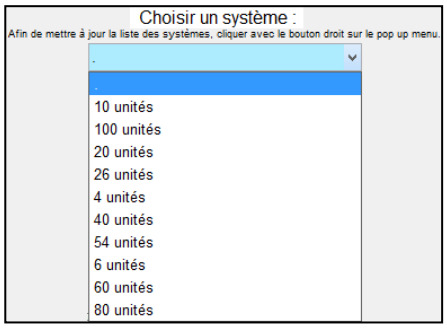
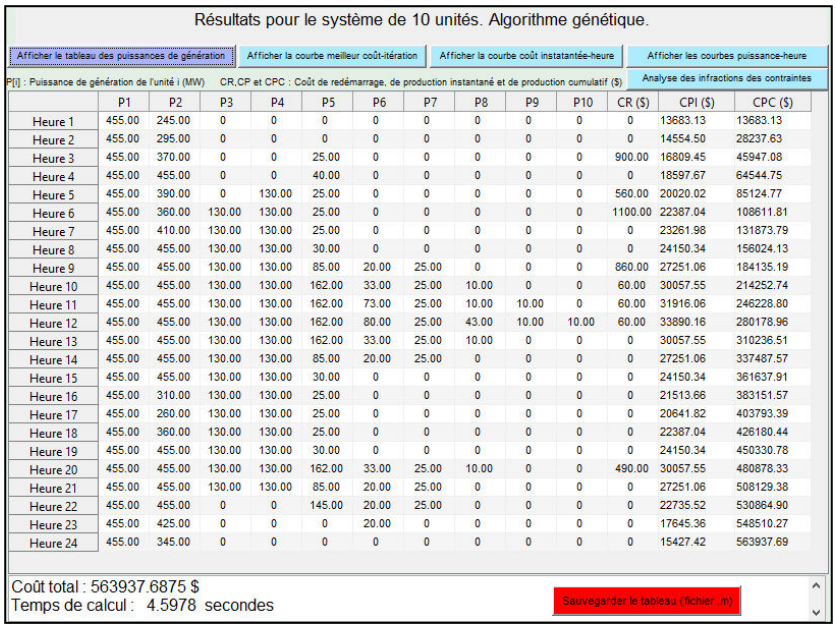



Figure 4. 7 Choix d'un système d'essai.

- Les boutons poussoirs permettent à l'utilisateur d'entamer une simulation et de générer les paramètres optimaux pour chaque système.
- Les curseurs permettent de fixer les valeurs des paramètres relatifs aux méthodes telles que les probabilités, le nombre d'itérations…etc.
- Les check-box permettent de choisir le type de critère d'arrêt de la méthode, le type de génération de la solution initiale, les opérations additionnelles, …etc.

## IV.2.4. Interface d'affichage des résultats de simulation :




| | P1 | P2 | P3 | P4 | P5 | P6 | P7 | P8 | P9 | P10 | CR ($) | CPI ($) | CPC ($) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Heure 1 | 455.00 | 245.00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13683.13 | 13683.13 |
| Heure 2 | 455.00 | 295.00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 14554.50 | 28237.63 |
| Heure 3 | 455.00 | 370.00 | 0 | 0 | 25.00 | 0 | 0 | 0 | 0 | 0 | 900.00 | 16809.45 | 45947.08 |
| Heure 4 | 455.00 | 455.00 | 0 | 0 | 40.00 | 0 | 0 | 0 | 0 | 0 | 0 | 18597.67 | 64544.75 |
| Heure 5 | 455.00 | 390.00 | 0 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 560.00 | 20020.02 | 85124.77 |
| Heure 6 | 455.00 | 360.00 | 130.00 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 1100.00 | 22387.04 | 108611.81 |
| Heure 7 | 455.00 | 410.00 | 130.00 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 0 | 23261.98 | 131873.79 |
| Heure 8 | 455.00 | 455.00 | 130.00 | 130.00 | 30.00 | 0 | 0 | 0 | 0 | 0 | 0 | 24150.34 | 156024.13 |
| Heure 9 | 455.00 | 455.00 | 130.00 | 130.00 | 85.00 | 20.00 | 25.00 | 0 | 0 | 0 | 860.00 | 27251.06 | 184135.19 |
| Heure 10 | 455.00 | 455.00 | 130.00 | 130.00 | 162.00 | 33.00 | 25.00 | 10.00 | 0 | 0 | 60.00 | 30057.55 | 214252.74 |
| Heure 11 | 455.00 | 455.00 | 130.00 | 130.00 | 162.00 | 73.00 | 25.00 | 10.00 | 10.00 | 0 | 60.00 | 31916.06 | 246228.80 |
| Heure 12 | 455.00 | 455.00 | 130.00 | 130.00 | 162.00 | 80.00 | 25.00 | 43.00 | 10.00 | 10.00 | 60.00 | 33890.16 | 280178.96 |
| Heure 13 | 455.00 | 455.00 | 130.00 | 130.00 | 162.00 | 33.00 | 25.00 | 10.00 | 0 | 0 | 0 | 30057.55 | 310236.51 |
| Heure 14 | 455.00 | 455.00 | 130.00 | 130.00 | 85.00 | 20.00 | 25.00 | 0 | 0 | 0 | 0 | 27251.06 | 337487.57 |
| Heure 15 | 455.00 | 455.00 | 130.00 | 130.00 | 30.00 | 0 | 0 | 0 | 0 | 0 | 0 | 24150.34 | 361637.91 |
| Heure 16 | 455.00 | 310.00 | 130.00 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 0 | 21513.66 | 383151.57 |
| Heure 17 | 455.00 | 260.00 | 130.00 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 0 | 20641.82 | 403793.39 |
| Heure 18 | 455.00 | 360.00 | 130.00 | 130.00 | 25.00 | 0 | 0 | 0 | 0 | 0 | 0 | 22387.04 | 426180.44 |
| Heure 19 | 455.00 | 455.00 | 130.00 | 130.00 | 30.00 | 0 | 0 | 0 | 0 | 0 | 0 | 24150.34 | 450330.78 |
| Heure 20 | 455.00 | 455.00 | 130.00 | 130.00 | 162.00 | 33.00 | 25.00 | 10.00 | 0 | 0 | 490.00 | 30057.55 | 480878.33 |
| Heure 21 | 455.00 | 455.00 | 130.00 | 130.00 | 85.00 | 20.00 | 25.00 | 0 | 0 | 0 | 0 | 27251.06 | 508129.38 |
| Heure 22 | 455.00 | 455.00 | 0 | 0 | 145.00 | 20.00 | 25.00 | 0 | 0 | 0 | 0 | 22735.52 | 530864.90 |
| Heure 23 | 455.00 | 425.00 | 0 | 0 | 0 | 20.00 | 0 | 0 | 0 | 0 | 0 | 17645.36 | 548510.27 |
| Heure 24 | 455.00 | 345.00 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15427.42 | 563937.69 |

Figure 4. 8 Interface d'affichage des résultats.

Les différents éléments relatifs à l'interface d'affichage des résultats sont :

- Des barres de progression apparaissent lors du déroulement d'une simulation. La progression est en fonction du temps (heures) pour les méthodes

déterministes tandis qu'elle l'est en fonction des itérations pour les méthodes avancées.

- Des boutons poussoirs qui permettent d'interrompre la simulation, de sauvegarder les données de sorties en une multitude de formats et d'afficher les courbes de charges, de coûts, tableau de planification …etc.

# IV.3. Application des méthodes déterministes à l'UCP:

## IV.3.1. Liste de priorité :

En se basant sur la méthodologie décrite dans le chapitre II, nous avons élaboré un programme permettant d'appliquer la méthode de liste de priorité sur des systèmes de différentes tailles. Le sous-problème de répartition économique a été traité par la méthode de lambda logique décrite dans le chapitre I. La planification du système de 10 unités établie par liste de priorité est donnée par le tableau VIII.

Tableau VIII. Résultats pour le système à 10 unités par méthode de liste de priorité.

| Heures | Unités | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| 1 | 455 | 245 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 2 | 455 | 295 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 455 | 265 | 0 | 130 | 0 | 0 | 0 | 0 | 0 | 0 |
| 4 | 455 | 235 | 130 | 130 | 0 | 0 | 0 | 0 | 0 | 0 |
| 5 | 455 | 260 | 130 | 130 | 25 | 0 | 0 | 0 | 0 | 0 |
| 6 | 455 | 360 | 130 | 130 | 25 | 0 | 0 | 0 | 0 | 0 |
| 7 | 455 | 410 | 130 | 130 | 25 | 0 | 0 | 0 | 0 | 0 |
| 8 | 455 | 455 | 130 | 130 | 30 | 0 | 0 | 0 | 0 | 0 |
| 9 | 455 | 455 | 130 | 130 | 85 | 20 | 25 | 0 | 0 | 0 |
| 10 | 455 | 455 | 130 | 130 | 162 | 33 | 25 | 10 | 0 | 0 |
| 11 | 455 | 455 | 130 | 130 | 162 | 73 | 25 | 10 | 10 | 0 |
| 12 | 455 | 455 | 130 | 130 | 162 | 80 | 25 | 43 | 10 | 10 |
| 13 | 455 | 455 | 130 | 130 | 162 | 33 | 25 | 10 | 0 | 0 |
| 14 | 455 | 455 | 130 | 130 | 85 | 20 | 25 | 0 | 0 | 0 |
| 15 | 455 | 455 | 130 | 130 | 30 | 0 | 0 | 0 | 0 | 0 |
| 16 | 455 | 310 | 130 | 130 | 0 | 0 | 0 | 0 | 0 | 0 |
| 17 | 455 | 260 | 130 | 130 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 455 | 360 | 130 | 130 | 25 | 0 | 0 | 0 | 0 | 0 |
| 19 | 455 | 455 | 130 | 130 | 30 | 0 | 0 | 0 | 0 | 0 |
| 20 | 455 | 455 | 130 | 130 | 162 | 33 | 25 | 10 | 0 | 0 |
| 21 | 455 | 455 | 130 | 130 | 85 | 20 | 25 | 0 | 0 | 0 |
| 22 | 455 | 360 | 130 | 130 | 25 | 0 | 0 | 0 | 0 | 0 |
| 23 | 455 | 315 | 0 | 130 | 0 | 0 | 0 | 0 | 0 | 0 |
| 24 | 455 | 345 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| Coût de production total | | | | | | | | | | 564799.3120 $ |
| Temps de calcul | | | | | | | | | | 0.0797 secondes |

Cette méthode, quoique très rapide, présente des infractions des contraintes de temps minimum d'arrêt et de redémarrage, notamment :

- L'unité 5 à l'heure 18 : l'unité a redémarré après 2 heures d'arrêt, alors que son temps minimum d'arrêt est de 6 heures.
- L'unité 6 à l'heure 22 : l'unité s'est arrêtée après 2 heures de fonctionnement, alors que son temps minimum de redémarrage est de 3 heures.
- L'unité 7 à l'heure 22 : l'unité s'est arrêtée après 2 heures de fonctionnement, alors que son temps minimum de redémarrage est de 3 heures.
- L'unité 5 à l'heure 23 : l'unité s'est arrêtée après 5 heures de fonctionnement, alors que son temps minimum de redémarrage est de 6 heures.

Ainsi, la méthode de liste de priorité est susceptible d'enfreindre les contraintes de temps, ce qui peut compromettre la faisabilité de la solution. Bien que pour des systèmes relativement petits, l'infraction des contraintes de temps est moins probable, ceci n'est pas le cas pour des systèmes de grandes tailles, comme la figure (4.9) le montre. Des tests ont été effectués sur les systèmes de 10, 20, 40, 60, 80 et 100 unités, et on remarque que plus le nombre d'unités augmente, plus le nombre d'infraction des contraintes de temps augmente.

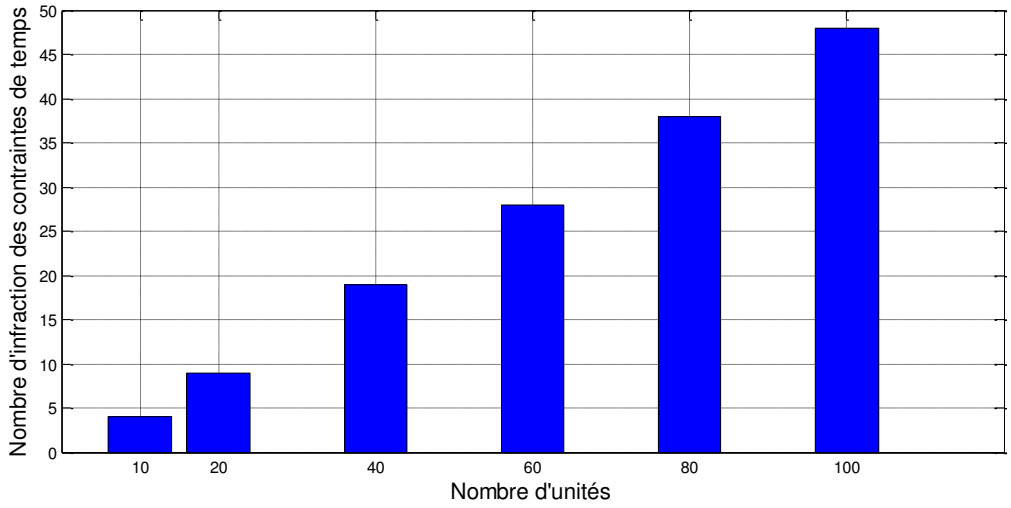



Figure 4. 9 Nombre d'infraction des contraintes de temps en fonction du nombre d'unités.

La figure ci-contre montre les infractions des contraintes de temps situées sur la courbe de charge du système 118 bus (54 unités). On constate que pendant les heures qui suivent les heures de pointe, les états pris en considération ont tendance à enfreindre la

contrainte de temps minimum de redémarrage car la charge prévisionnelle diminue ainsi moins d'unité sont nécessaires pour satisfaire la demande. Tandis que pendant les heures menant aux heures de pointe où les unités ont tendances à redémarrer, la contrainte de temps minimum d'arrêt est enfreinte. Ainsi, ces infractions sont beaucoups plus fréquences pendant les heures creuses où les unités ont été soit arrêtées, soit redémarrées.

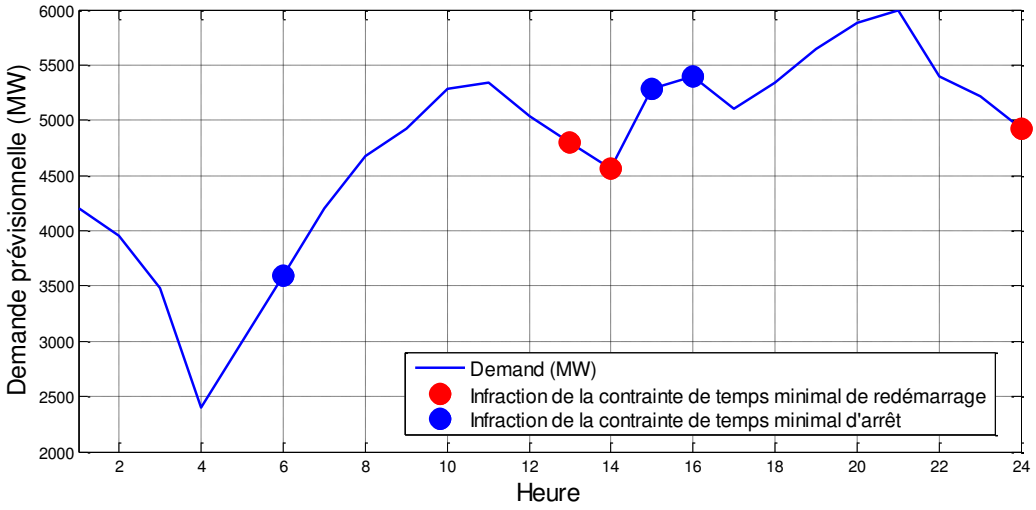



Figure 4. 10 Les infractions des contraintes de temps situées sur la courbe de charge.

On constate que la méthode de liste de priorité n'est valable que si l'on considère les contraintes de demande à satisfaire, de réserve à garantir et de puissance bornée. C'est à partir de ces résultats qu'on arrive à déterminer la vraie difficulté du problème de l'UCP. En effet, il est simple de satisfaire la demande car ce n'est qu'un choix des unités les moins coûteuses, cependant la difficulté du problème combinatoire prend beaucoup plus d'ampleur lorsqu'on prend en considération les contraintes de temps minimum d'arrêt et de redémarrage.

## IV.3.2. Programmation dynamique :

En se basant sur la méthodologie décrite dans le chapitre II, un programme a été élaboré permettant d'appliquer la programmation dynamique pour la résolution de l'UCP. D'après les résultats obtenus pour le système de 4 unités, le processus de programmation dynamique en avant par énumération complète est illustré dans la figure ci-contre.

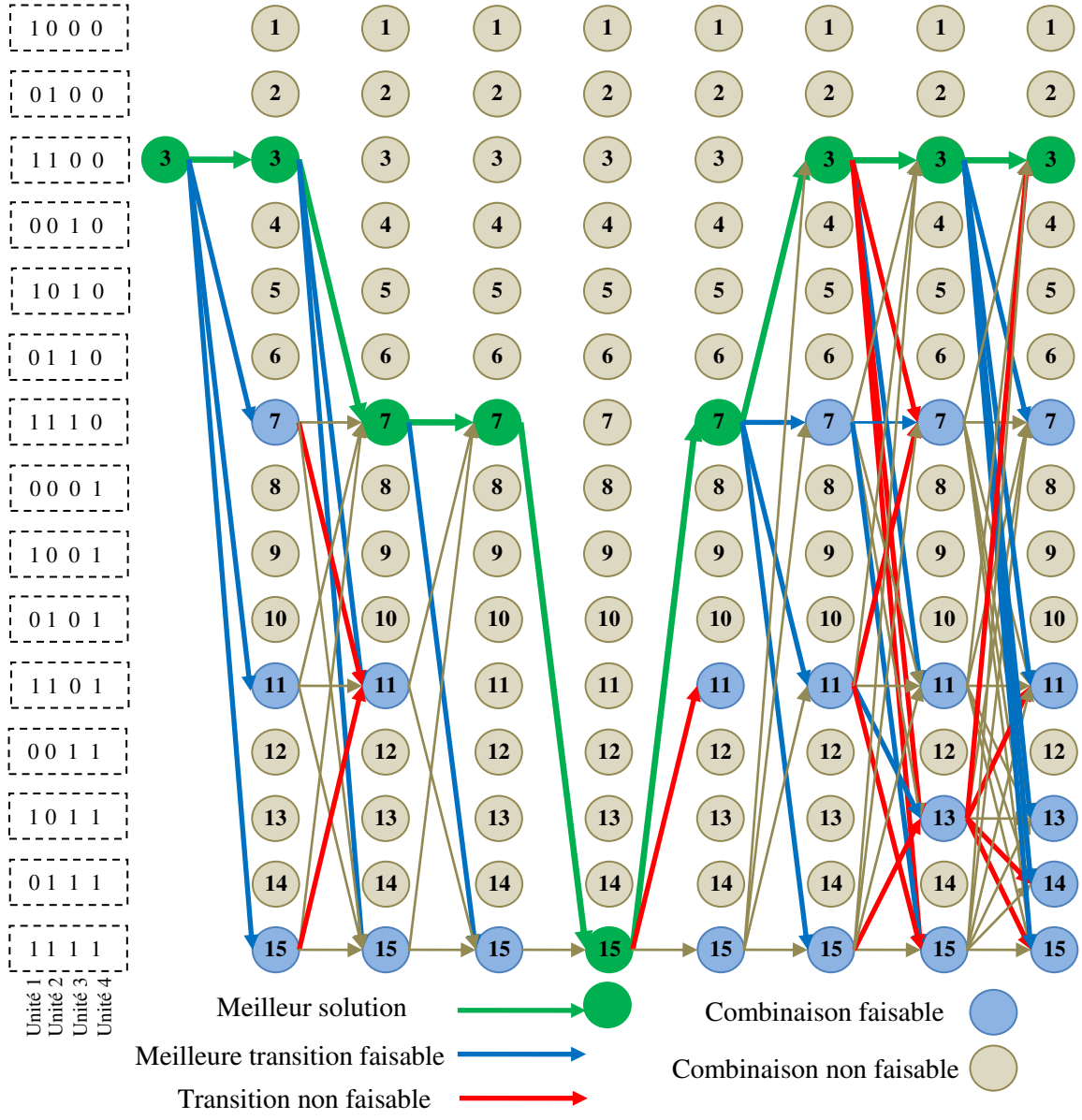



Figure 4. 11 Programmation dynamique (énumération complète) appliquée au système de 4 unités.

Le tableau IX représente la planification optimale du système de 10 unités par programmation dynamique. Les cas possibles ont été énumérés par énumération complète et l'espace de recherche n'a pas été limité, c'est-à-dire qu'à chaque heure toutes les stratégies possibles ont été enregistrées. Ce qui en résulte est un temps de calcul beaucoup trop important, malgré l'optimalité de la solution.

Tableau IX. Résultats pour le système de 10 unités par programmation dynamique (énumération complète).

| Heures | Unités | | | | | | | | | | Coût de redémarrage ($) | Coût de production instantané ($) | Coût de production cumulatif ($) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | | | |
| 1 | 455 | 245 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 13683.1 | 13683.1 |
| 2 | 455 | 295 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 14554.5 | 28237.6 |
| 3 | 455 | 370 | 0 | 0 | 25 | 0 | 0 | 0 | 0 | 0 | 900 | 16809.4 | 45947.1 |
| 4 | 455 | 455 | 0 | 0 | 40 | 0 | 0 | 0 | 0 | 0 | 0 | 18597.7 | 64544.7 |
| 5 | 455 | 390 | 0 | 130 | 25 | 0 | 0 | 0 | 0 | 0 | 560 | 20020 | 85124.8 |
| 6 | 455 | 360 | 130 | 130 | 25 | 0 | 0 | 0 | 0 | 0 | 1100 | 22387 | 108612 |
| 7 | 455 | 410 | 130 | 130 | 25 | 0 | 0 | 0 | 0 | 0 | 0 | 23262 | 131874 |
| 8 | 455 | 455 | 130 | 130 | 30 | 0 | 0 | 0 | 0 | 0 | 0 | 24150.3 | 156024 |
| 9 | 455 | 455 | 130 | 130 | 85 | 20 | 25 | 0 | 0 | 0 | 860 | 27251.1 | 184135 |
| 10 | 455 | 455 | 130 | 130 | 162 | 33 | 25 | 10 | 0 | 0 | 60 | 30057.6 | 214253 |
| 11 | 455 | 455 | 130 | 130 | 162 | 73 | 25 | 10 | 10 | 0 | 60 | 31916.1 | 246229 |
| 12 | 455 | 455 | 130 | 130 | 162 | 80 | 25 | 43 | 10 | 10 | 60 | 33890.2 | 280179 |
| 13 | 455 | 455 | 130 | 130 | 162 | 33 | 25 | 10 | 0 | 0 | 0 | 30057.6 | 310237 |
| 14 | 455 | 455 | 130 | 130 | 85 | 20 | 25 | 0 | 0 | 0 | 0 | 27251.1 | 337488 |
| 15 | 455 | 455 | 130 | 130 | 30 | 0 | 0 | 0 | 0 | 0 | 0 | 24150.3 | 361638 |
| 16 | 455 | 310 | 130 | 130 | 25 | 0 | 0 | 0 | 0 | 0 | 0 | 21513.7 | 383152 |
| 17 | 455 | 260 | 130 | 130 | 25 | 0 | 0 | 0 | 0 | 0 | 0 | 20641.8 | 403793 |
| 18 | 455 | 360 | 130 | 130 | 25 | 0 | 0 | 0 | 0 | 0 | 0 | 22387 | 426180 |
| 19 | 455 | 455 | 130 | 130 | 30 | 0 | 0 | 0 | 0 | 0 | 0 | 24150.3 | 450331 |
| 20 | 455 | 455 | 130 | 130 | 162 | 33 | 25 | 10 | 0 | 0 | 490 | 30057.6 | 480878 |
| 21 | 455 | 455 | 130 | 130 | 85 | 20 | 25 | 0 | 0 | 0 | 0 | 27251.1 | 508129 |
| 22 | 455 | 455 | 0 | 0 | 145 | 20 | 25 | 0 | 0 | 0 | 0 | 22735.5 | 530865 |
| 23 | 455 | 425 | 0 | 0 | 0 | 20 | 0 | 0 | 0 | 0 | 0 | 17645.4 | 548510 |
| 24 | 455 | 345 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 15427.4 | 563938 |
| Coût de production total | 563937.6875 $ | | | | | | | | | | | | |
| Temps de calcul | 161.7429 secondes | | | | | | | | | | | | |

En analysant la figure, on remarque que le temps de calcul de chaque heure est semblable au profil de la charge. En effet, les temps de calcul pendant les heures creuses sont beaucoup plus élevés que ceux des heures de pointes. Ceci est dû au fait que pendant

les heures de pointes, le nombre de cas possibles de satisfaire la demande prévisionnelle est considérablement réduit. Dans notre cas, il n'ya qu'un seul cas possible satisfaire la demande de l'heure 12. Ainsi, toutes les stratégies enregistrées auparavant vont converger vers un seul cas, ce qui diminue le temps d'exécution du programme pendant cette heure.

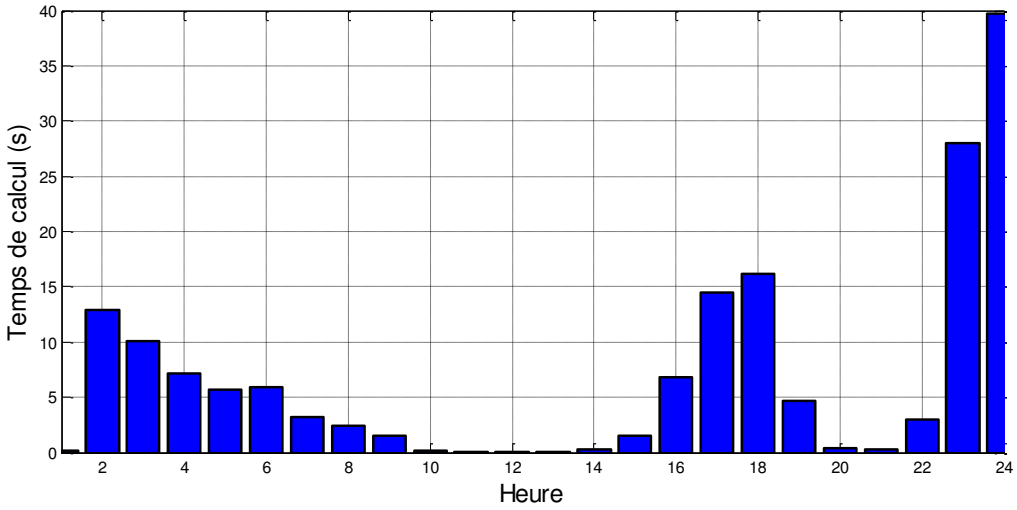



Figure 4. 12 Temps de calcul écoulé à chaque heure - système de 10 unités (programmation dynamique énumération complète).

Une des alternatives envisageables nous permettant de réduire le temps de calcul est l'énumération par liste de priorité. Cela dit, ce type d'énumération ne donne pas la solution optimale car on va de $2^N - 1$ cas possibles à $N$ cas possibles. L'espace de recherche n'est pas assez vaste pour trouver l'optimum et bien qu'avec ce type d'énumération il est possible d'appliquer la programmation dynamique aux systèmes de grandes tailles (chose non faisable avec l'énumération complète), le temps de calcul reste assez important.

Une autre alternative consiste à limiter le nombre de stratégies enregistrées à chaque heure. Les meilleures stratégies en termes de coûts sont enregistrées pour la prochaine heure, tandis que les pires stratégies sont éliminées. Ceci est très efficace car on n'acceptant que 30% des stratégies par heure, on obtient le même coût de production total que si l'on faisait le test avec 100% des stratégies enregistrées. Cependant, si le test est fait avec moins de 30%, à une certaine heure aucune stratégie ne respecte les contraintes de temps minimum d'arrêt et de redémarrage et par conséquent toutes les stratégies sont éliminées.

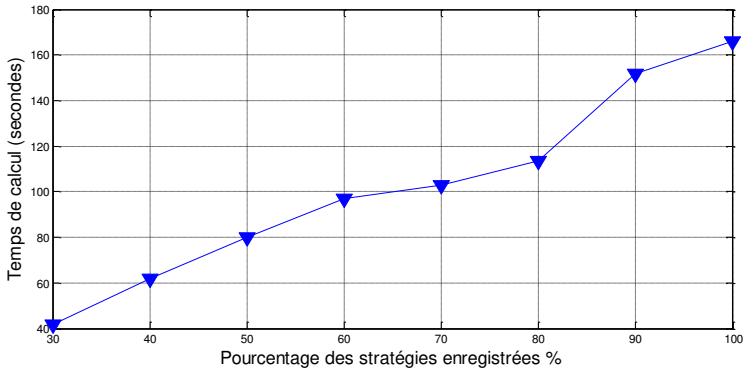



Figure 4. 13 Pourcentage des stratégies enregistrées en fonction du temps d'exécution (système 10 unités).

## IV.3.3. Récapitulatif :

À partir des résultats obtenus, on constate que l'énumération complète des combinaisons possibles donne un meilleur coût de production. Bien que la solution obtenue en énumérant à partir d'une liste de priorité stricte soit correcte, elle n'est pas optimale. Cela dit, il y a un compromis entre optimalité et temps de calcul. Certes, l'énumération complète des combinaisons possibles donne de bons résultats mais le temps de calculs devient important pour des systèmes de grande taille.

La liste de priorité donne de meilleurs résultats en termes de coût par rapport à la programmation dynamique séquentielle (énumération par liste de priorité stricte) et en termes de temps de calculs par rapport à la programmation dynamique (énumération complète). Cependant, cette méthode peut donner des planifications non faisables car elle ne prend pas en considération les contraintes de temps minimum d'arrêt et de redémarrage.

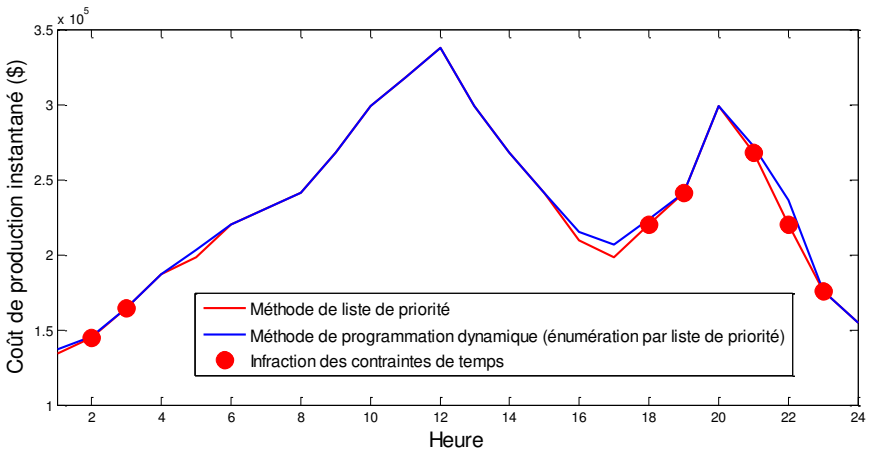



Figure 4. 14 Coûts de production instantanés - méthodes déterministes (système de 100 unités).

La figure (4. 14), qui correspond aux coûts de production instantanés obtenus par liste de priorité et par programmation dynamique séquentielle appliqués au système de 100 unités, illustre ce compromis entre faisabilité et optimalité. En effet, bien que la liste de priorité présente de meilleurs coûts de production pendant les heures creuses par rapport à la programmation dynamique séquentielle, elle présente également des infractions des contraintes de temps pendant ces heures là.

Comme on peut le voir dans le tableau X, le temps de calcul augmente exponentiellement avec le nombre de générateurs. Par exemple, si on songeait faire une énumération complète du système de 100 unités, on aurait à faire à 1.2677e+30 combinaisons possibles. L'ordinateur n'étant pas capable de traiter autant de données, il s'est avéré préférable d'uniquement faire le test pour le cas d'énumération par liste de priorité.

Tableau X. Récapitulatif des résultats de simulation des méthodes déterministes.

| Nombre d'unités de production | Coût de production total ($) | | | Temps de calcul (secondes) | | |
|---|---|---|---|---|---|---|
| | Liste de priorité | Programmation dynamique | | Liste de priorité | Programmation dynamique | |
| | | Énumération par liste de priorité | Énumération complète | | Énumération par liste de priorité | Énumération complète |
| 4 | 84869.4771 | 84869.4771 | 84869.4771 | 0.0411 | 0.1115 | 0.1115 |
| 10 | 564799.3120 | 565732.2537 | 563937.6875 | 0.0702 | 0.5801 | 161.6456 |
| 20 | 1124981.0481 | 1129061.346 | ---------- | 0.0805 | 2.3943 | ---------- |
| 40 | 2248460.7967 | 2254634.167 | ---------- | 0.1529 | 17.396 | ---------- |
| 60 | 3367196.0083 | 3378390.008 | ---------- | 0.3185 | 70.720 | ---------- |
| 80 | 4490679.8906 | 4503965.800 | ---------- | 0.6207 | 216.95 | ---------- |
| 100 | 5613794.9209 | 5627722.601 | ---------- | 1.1257 | 482.31 | ---------- |

# IV.4. Application des méthodes métaheuristiques au problème de l'Unit Commitment:

## IV.4.1. Algorithmes génétiques :

La méthodologie de l'algorithme génétique décrite dans le chapitre III est la base du programme élaboré pour la résolution de l'UCP. Plusieurs tests ont été effectués afin de déterminer l'influence des différents paramètres sur la performance de la méthode.

### IV.4.1.1. Application sur le système de 10 unités :

Cette méthode a été appliquée sur le système de 10 unités tel que les valeurs des paramètres utilisés sont :

- Taille de la population : 10.
- Nombre d'itérations : 100.
- Probabilité de croisement : $P_c$.
- Probabilité de mutation : $P_m$.

Après simulation, on obtient la courbe du coût de production total en fonction des itérations illustré dans la figure ci-contre.

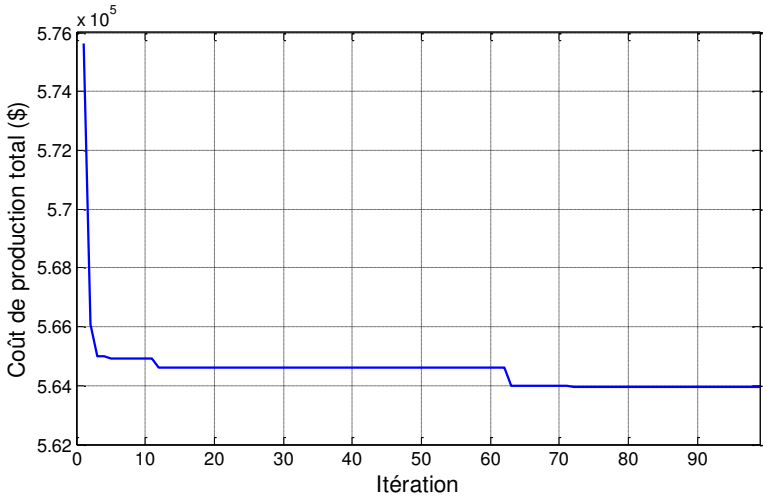



Figure 4. 15 Courbe de convergence de l'algorithme génétique appliqué au système de 10 unités.

La meilleure solution trouvée à la fin des itérations correspond au coût de production total $F_T = 563937.6875$ \$, qui est égal au coût calculé par programmation

dynamique. Cela dit, le temps de calcul est considérablement réduit par rapport à la programmation dynamique.

### IV.4.1.2. Influence du nombre de population :

La figure (4.16) illustre l'influence de la taille de la population sur la qualité de la solution trouvée pour trois systèmes de différentes tailles (40, 60 et 80 unités). Sachant qu'une population correspond à un ensemble de solutions qui se partagent des données, plus le nombre de solutions envisageables augmente, plus on diversifie l'espace de recherche et les chances de trouver l'optimum augmente. Cela dit, le nombre d'itérations nécessaires pour arriver à cet optimum augmente également ce qui implique un temps de calcul relativement grand. Par ailleurs, l'algorithme a tendance à converger prématurément vers un optimum local lorsque la population est petite.

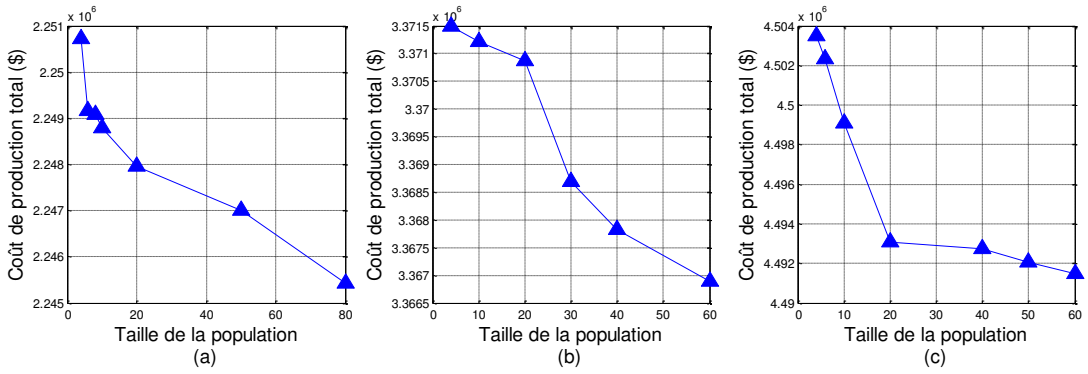



Figure 4. 16 Influence de la taille de population sur le coût de production total pour les systèmes de : a. 40 unités b. 60 unités c. 80 unités.

### IV.4.1.3. Influence des nombres de points de croisement et de mutation:

Il faut rappeler que l'opérateur de croisement permet de combiner les informations génétiques de deux solutions afin de créer deux nouvelles solutions. Dans notre cas, deux chaînes de bits représentent chacune l'état ON/OFF de toutes les unités pendant une certaine heure. Un échange de données entre ces deux chaînes peut se faire en un seul point ou bien en plusieurs points. Pour les systèmes de petites tailles, tel que le système de 10 unités, un point de croisement est suffisant. Cependant, lorsque l'algorithme est appliquée à un système de grande taille, tel que le système de 100 unités, un seul point de croisement n'est pas suffisant. Ceci est dû au fait que les bits rapprochés dans la chaîne de données ont tendance à rester rapprochés pendant plusieurs itérations, ce qui en résulte une lente

convergence. Par ailleurs, augmenter le nombre de points de croisement peut garantir une meilleure uniformité d'échange de données.

L'opérateur de mutation, quant à lui, assure la diversification de l'espace de recherche. Ainsi, une mutation à deux points permet de maintenir une meilleure diversité de l'espace de recherche par rapport à la mutation à un seul point, particulièrement pour les systèmes de grandes tailles.

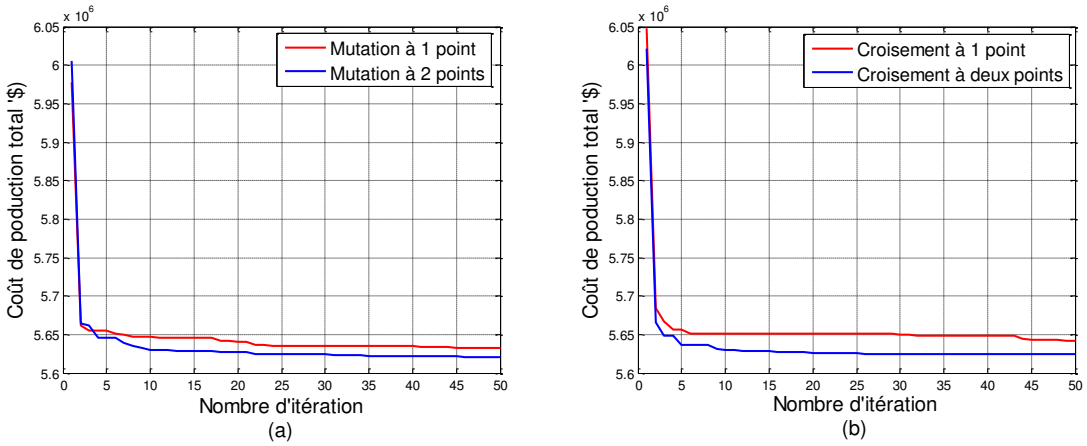



Figure 4. 17 Influence du nombre de points de croisement (b) et du nombre de points de mutation (a) sur la convergence de l'algorithme génétique (système de 100 unités).

### IV.4.1.4. Influence des probabilités de mutation et de croisement :

Les probabilités de mutation et de croisement ne sont que des paramètres qui permettent d'ajuster le comportement de l'algorithme génétique. Une diminution de la probabilité de croisement $P_C$ va permettre aux solutions actuelles d'être transférées vers l'itération suivante sans modification. Dans notre cas, ceci n'est pas forcément bénéfique, comme illustré dans la figure (4.18).

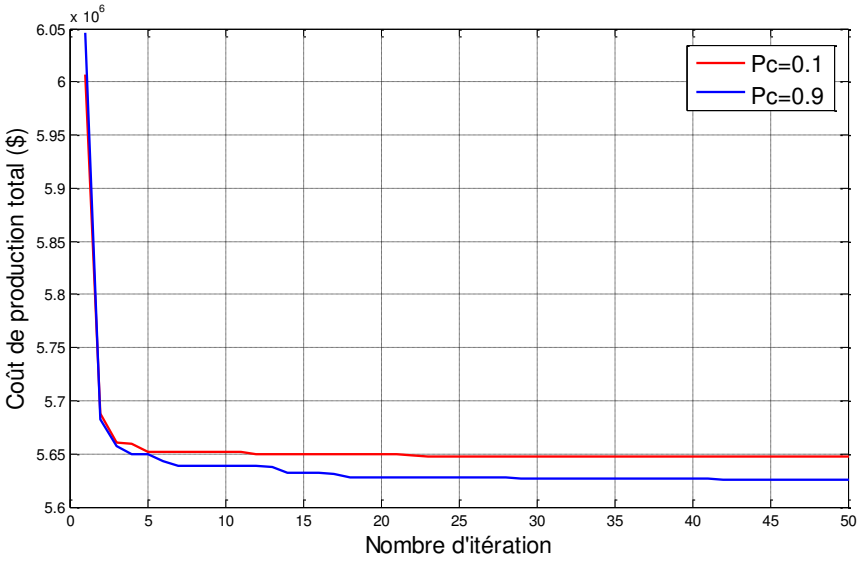



Figure 4. 18 Influence de la probabilité de croisement sur la convergence.

La probabilité de mutation est généralement faible (entre 0.001 et 0.1) car il est préférable d'exploiter que de diversifier. En effet la mutation a pour but de diversifier l'exploration tandis que le croisement conduit la population à converger vers les meilleures solutions trouvées (exploitation). Une probabilité de mutation trop importante empêche la convergence vers une solution optimale, comme illustré dans la figure (4.19).

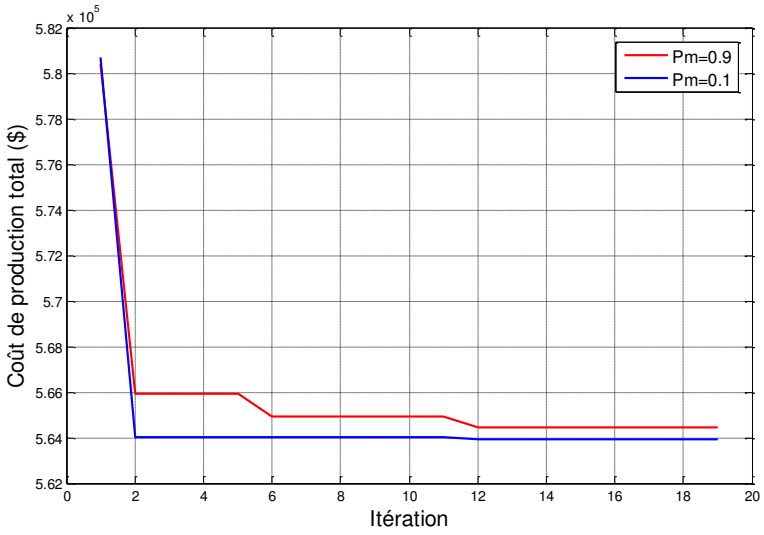



Figure 4. 19 Influence de la probabilité de mutation sur la convergence.

## IV.4.2. Optimisation par essaims particulaires :

Les différentes méthodologies de l'optimisation par essaims particulaires décrites dans le chapitre III on été appliquées au problème de l'UCP, tandis que la répartition économique a été résolu par la méthode de lambda logique.

### IV.4.3.1. Application sur le système de 10 unités :

Afin de mieux évaluer la performance de chaque méthodologie, on a effectué des simulations sur le système de 10 unités. La figure (4.20) représente la courbe de convergence des différentes méthodes.

On remarque que les méthodes hybrides donnent de meilleurs résultats par rapport à l'approche binaire. En effet, cette approche a le désavantage de rechercher tous points de l'espace de recherche ce qui augmente le nombre d'itérations nécessaires pour arriver à la solution optimale. En outre, cette méthode a tendance à être piégée dans les minima locaux et ceci est plus fréquent pour des systèmes de grandes tailles. Tandis que les deux approches hybrides donnent de bons résultats et convergent vers la solution optimale en un temps raisonnable. Cela dit l'hybridation avec la recherche locale requiert moins d'itérations que l'hybridation avec l'algorithme génétique.

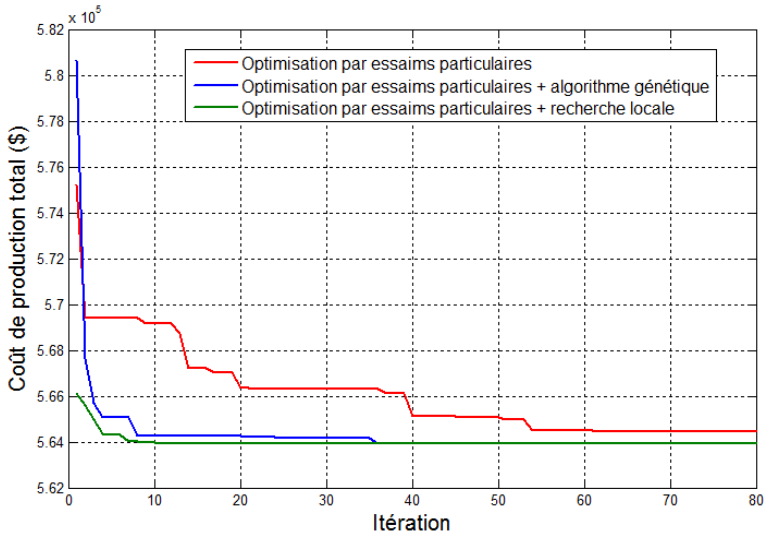



Figure 4. 20 Courbe de convergence des trois approches de l'optimisation par essaims particulaires.

### IV.4.2.2. Application sur le système de 100 unités :

En appliquant la méthode hybride (optimisation par essaims particulaires et recherche locale), on obtient la courbe de convergence illustrée dans la figure (4.21) tel que les paramètres utilisés sont les suivants :

- Taille de la population : 5.
- Nombre d'itérations : 160.
- Coefficients d'accélérations adaptatifs.
- Limite de temps de la recherche locale : 2 secondes.

Il faut rappeler que cette hybridation consiste à améliorer la solution de la meilleure particule en perturbant cette dernière. La convergence de la meilleure particule est notamment meilleure que celle des autres particules. Ainsi, ce processus de perturbation a permit d'intensifier la recherche localement. Vers la fin des itérations, toutes les particules convergent vers la meilleure solution. Parmi les méthodes avancées citées dans ce chapitre, on obtient le meilleur coût de production total pour le système de 100 unités : 5608096.97 $.

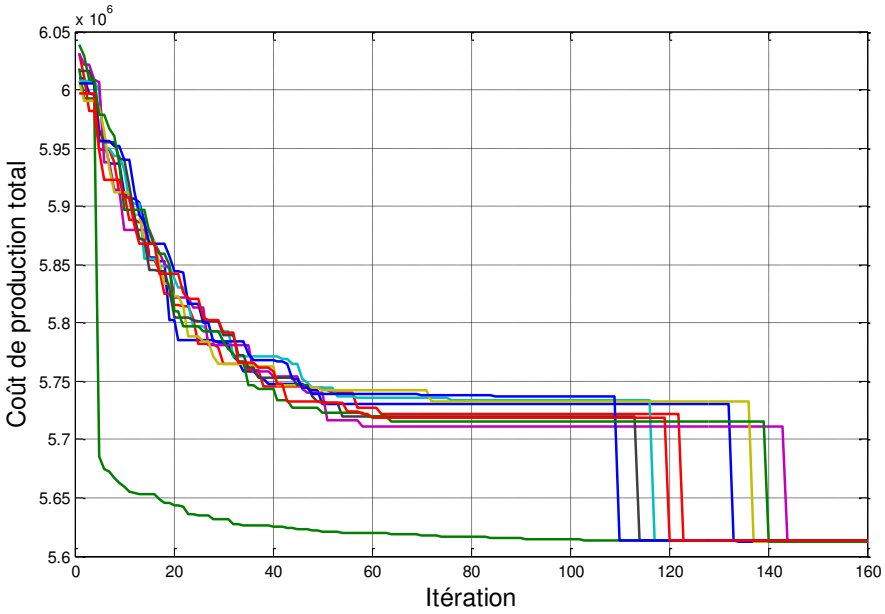



Figure 4. 21 Convergence de la méthode hybride (optimisation par essaims particulaires et recherche locale).

### IV.4.2.3. Influence des coefficients C1 et C2 (application sur le système de 26 unités):

D'après la figure (4.22), qui illustre le meilleur coût trouvée par chaque particule au fil des itérations, on constate que les paramètres C1 et C2 ont un effet sur la convergence de la méthode. Une grande valeur du paramètre cognitif et une petite valeur du paramètre social permet aux particules d'explorer l'espace de recherche, tandis qu'une petite valeur du paramètre cognitif et une grande valeur du paramètre social permet aux particules de converger vers le meilleur élément de l'essaim.

Afin d'encourager l'exploration au début des itérations et l'exploitation à la fin des itérations, il est préférable d'utiliser l'équation (3.10) qui varie les coefficients en fonction des itérations.

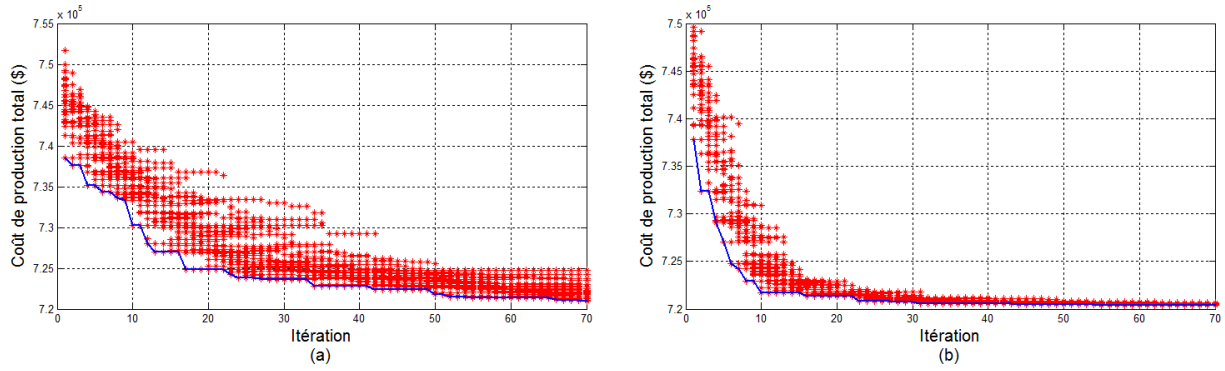



Figure 4. 22 Influence de c1 et c2 sur la convergence : a. c1=3 et c2=0.5 b. c1=0.5 et c2=3.

## IV.4.3. Recuit simulé :

Un programme basé sur la méthodologie du recuit simulé (chapitre III) a été appliqué sur des systèmes de différentes tailles. Le choix des paramètres et la méthode de génération de la solution initiale sont discutés.

### IV.4.3.1. Application sur le système de 10 unités :

La figure (4.23) illustre les courbes de convergence de la méthode appliquée au système de 10 unités. Le choix d'une valeur initiale du paramètre de contrôle *T* est important car si cette valeur initiale est trop élevée, la probabilité d'acceptation d'une solution de moindre qualité est élevée tout au long des itérations. Il est préférable d'accepter tous types de solutions au début de la recherche pour encourager l'exploration,

cependant à la fin des itérations on cherche plutôt à maintenir une bonne convergence de la solution. Ainsi, le choix du paramètre de contrôle est crucial.

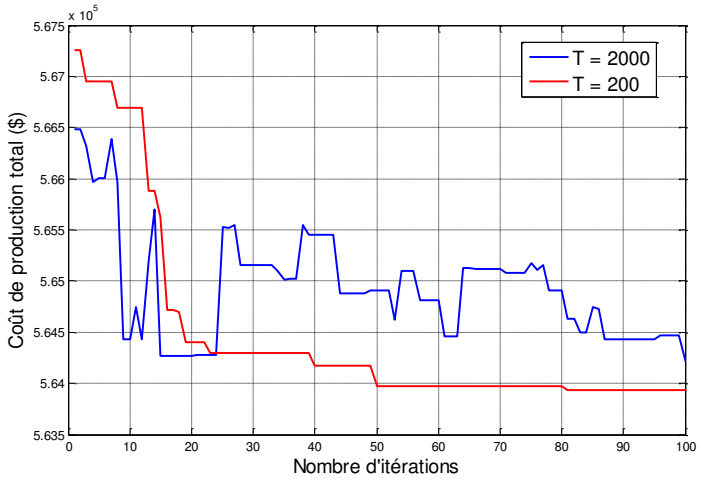



Figure 4. 23 Courbe de convergence de la méthode du recuit simulé pour le système de 10 unités.

### IV.4.3.2. Application sur le système de 54 unités (118 bus) :

La figure (4.24) illustre les courbes de convergence du recuit simulé appliqué sur le réseau 118 bus ayant 54 unités. Deux méthodes de génération de la solution initiale ont étés prises en considération. La solution initiale générée par liste de priorité présente une meilleure convergence par rapport à la solution initiale générée aléatoirement.

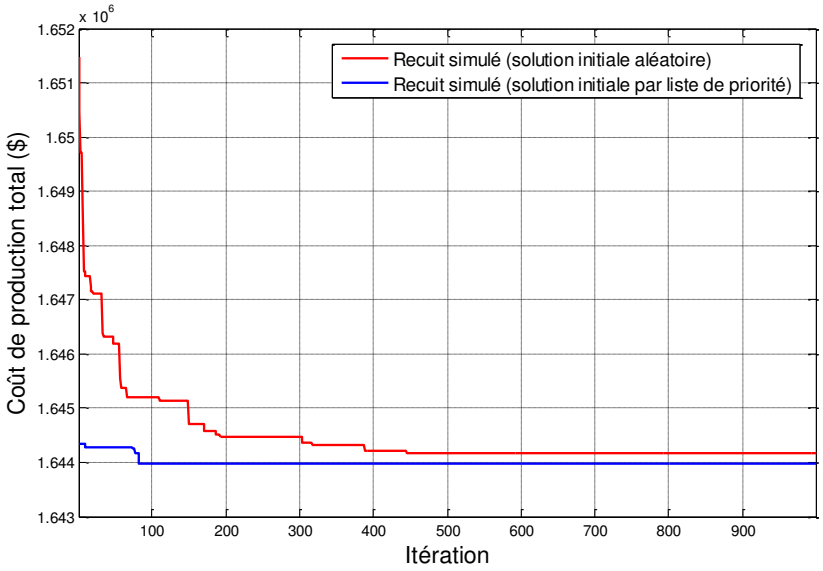



Figure 4. 24 Convergence de la méthode du recuit simulé pour deux différentes approches de la solution initiale (système 54 unités).

### IV.4.4. Récapitulatif :

Les meilleurs et pires coûts de production de chaque méthode avancée sont donnés par le tableau XI. Tandis que les figures (4.25) et (4.26) nous illustrent les courbes de temps d'exécution et de nombre d'itérations en fonction du nombre d'unités de production.

Sur le plan ressources informatiques, les algorithmes génétiques sont les plus coûteuses car la transition d'un ensemble de solution à un autre requiert plusieurs opérations effectuées sur chaque solution. En effet, l'exécution d'un tel algorithme en une itération se fait en plusieurs étapes : élitisme, sélection, croisement et mutation. Étant donné que la convergence de cette méthode est lente, le nombre d'itérations nécessaires pour atteindre l'optimum est assez grand. Par conséquent, le temps d'exécution et le nombre d'itérations augmentent exponentiellement en fonction du nombre d'unités.

Quant au recuit simulé, le grand nombre d'itérations est dû au fait que cette méthode est à solution unique. Cependant, puisque l'action de perturbation de la solution est facilement implémentable au problème de l'UCP, cette méthode converge rapidement à condition de bien choisir les paramètres de contrôle.

L'optimisation binaire par essaims particulaires est la méthode la moins optimale parmi les méthodes avancées. Ainsi, nous avons proposé deux hybridations : hybridation avec les algorithmes génétiques et hybridation avec la recherche locale. La première améliore la qualité de la solution pour des systèmes relativement petits, cependant lorsque le nombre d'unités augmente, la méthode converge vers un optimum local. La deuxième hybridation résout le problème de convergence prématurée grâce à la diversification de la recherche locale.

En comparant les résultats obtenus par les méthodes avancées, il est clair que l'optimisation par essaims particulaires avec recherche locale donne les meilleurs résultats en termes de coût de production total, de temps de calcul et de nombre d'itérations requises pour atteindre l'optimum. L'intensification de la recherche à chaque itération permet d'améliorer l'approche de base de l'optimisation par essaims particulaires.

Tableau XI. Récapitulatif des résultats obtenus par les méthodes avancées.

| Unité | Algorithme génétique | | Recuit simulé | | Optimisation par essaims particulaires | | PSO + recherche locale | |
|---|---|---|---|---|---|---|---|---|
| | Meilleur coût | Pire coût | Meilleur coût | Pire coût | Meilleur coût | Pire coût | Meilleur coût | Pire coût |
| 10 | 563937.69 | 564296.54 | 563937.69 | 564296.541 | 563937.68 | 565567.90 | 563937.68 | 563947.83 |
| 20 | 1124295.63 | 1124791.27 | 1123875.75 | 1124535.98 | 1124703.67 | 1126540.49 | 1123785.42 | 1124295.63 |
| 40 | 2247126.67 | 2248896.20 | 2245531.15 | 2246910.53 | 2253308.94 | 2269767.51 | 2245293.68 | 2246007.52 |
| 60 | 3367706.43 | 3369723.35 | 3365984.12 | 3367091.60 | 3391734.20 | 3420629.85 | 3363717.86 | 3366369.02 |
| 80 | 4490830.85 | 4498440.49 | 4489499.98 | 4491728.53 | 4527696.38 | 4574244.38 | 4489642.45 | 4492982.82 |
| 100 | 5613359.68 | 5619201.89 | 5609922.94 | 5613451.03 | 5691047.46 | 5701076.00 | 5608096.97 | 5613344.44 |

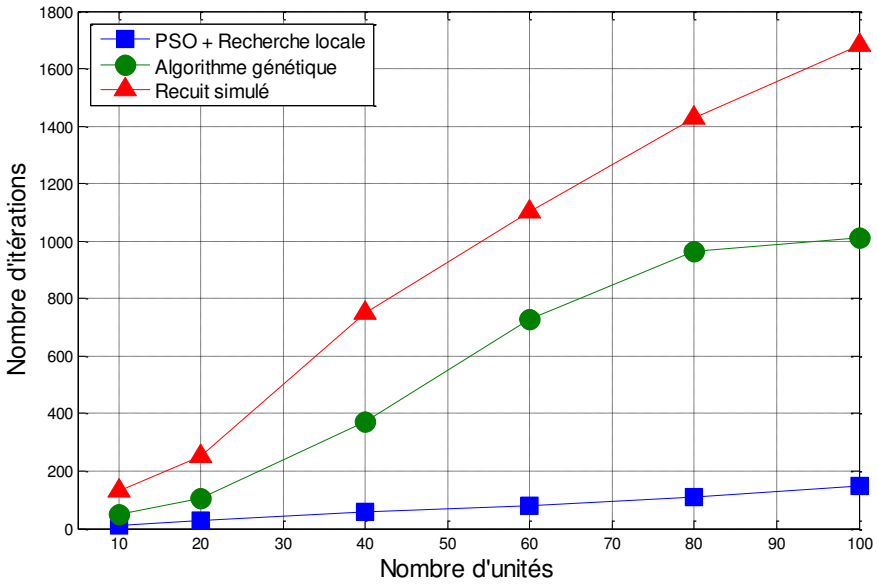



Figure 4. 25 Nombre d'itérations nécessaire pour différentes méthodes avancées.

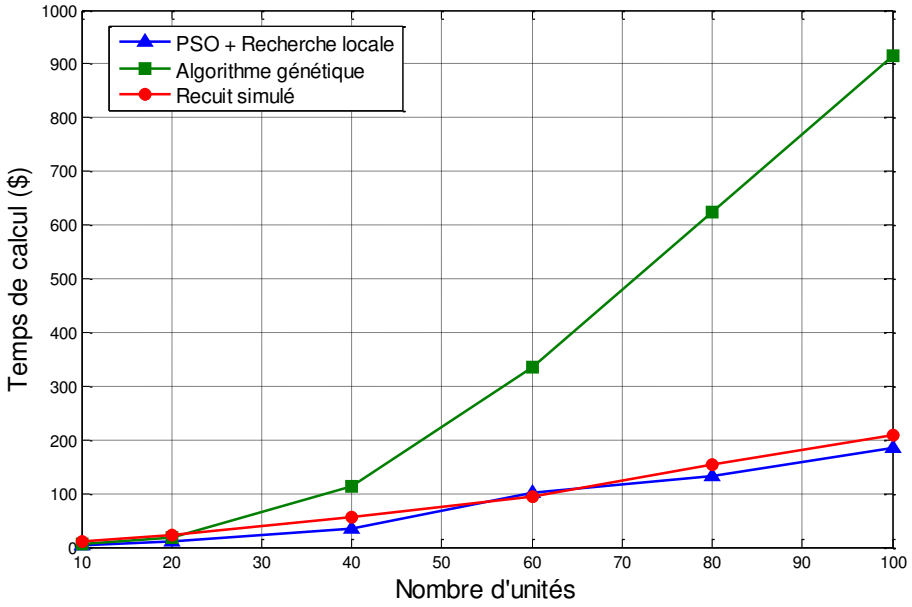



Figure 4. 26 Temps de calcul en fonction du nombre d'unités pour différentes méthodes avancées.

# Conclusion générale:

Dans ce mémoire, le problème fondamental a été présenté suivi d'une formulation mathématique. Les différentes contraintes qu'on peut faire subir au problème sont nombreuses, cependant afin d'éviter toute complexité excessive, les plus communes ont été citées. Afin de mieux se familiariser au problème d'UCP, une brève synthèse des méthodes de résolution a été développée. Un logiciel, développé afin de réaliser nos simulations, a été détaillé ainsi que les différents éléments le constituant. Ce logiciel recueille toutes les méthodes classiques et avancées ainsi que les bases de données et nous a permit de faciliter l'étude de la performance des méthodes suivantes :

En premier lieu, on a exposé et appliqué deux méthodes classiques : la méthode de liste de priorité et la programmation dynamique. La première n'est pas toujours faisable car elle ne respecte pas certaines contraintes, notamment les contraintes de temps minimum d'arrêt et de redémarrage. En ce qui concerne la programmation dynamique, on constate que malgré la résolution satisfaisante du problème d'un point de vue théorique, elle n'est pas réalisable lorsqu'on fait face aux réseaux de nos jours qui comptent des centaines d'unités de production. Bien que la méthode donne des résultats optimaux, le temps de calcul devient contraignant lorsqu'on l'applique à des systèmes de grandes tailles. On a proposé donc de réduire le nombre de cas possibles en énumérant selon une liste de priorité. Cela dit, la solution n'est pas optimale.

En second lieu, nous avons proposé trois méthodes avancées : algorithmes génétiques, recuit simulé et optimisation par essaims particulaires. Toutes ces méthodes ont pu résoudre le problème en un temps raisonnable et en obtenant des solutions de bonne qualité. Cela dit, chaque méthode a ses propres avantages et inconvénients.

L'optimisation binaire par essaims particulaires n'étant pas efficace pour la résolution de l'UCP, nous avons proposé une hybridation entre l'optimisation par essaims particulaires et la recherche locale 2-opt. Ce dernier non seulement présente les meilleurs résultats parmi les méthodes avancées, mais son utilisation est une bonne alternative lorsque les méthodes déterministes ne sont pas efficaces.

# Perspectives :

Au-delà des travaux effectués dans ce mémoire, il serait intéressant d'approfondir l'étude de l'engagement des turbines en traitant les points suivants :

- Prendre en considération d'autres contraintes telles que les contraintes environnementales, les contraintes d'élévation et de chute maximales de la puissance, la contrainte des puissances initiales délivrées par les unités, contraintes de zones interdites…etc.
- Envisager l'utilisation d'autres fonctions objectives qui visent à minimiser les émissions de gaz, les pertes de puissance…etc.
- Envisager d'autres hybridations, notamment des hybridations entre des méthodes déterministes et des méthodes avancées telles que : logique floue et programmation dynamique, relaxation lagrangienne et optimisation par essaims particulaires,…etc.

# Bibliographie

[1] Wood, A.J. and Wollenberg, B.F. Power Generation, Operation & Control. second ed. New York : John Wiley & Sons Ltd., 1996, 592p.

[2] Cohen, A.I. and Sherkat, V.R. Optimization-based methods for operations scheduling. *Proceedings of the IEEE*, 1987. vol. 75, n° 12, p. 1574-1591.

[3] Sydulu, M. A very fast and effective noniterative "λ-logic based" algorithm for economic dispatch of thermal units. *TENCON 99. Proceedings of the IEEE, Region 10 Conference,* December, 1999, vol. 2, n°. 10, p. 1434-1437.

[4] Muckstadt, J.A. and Koenig, S.A. An application of Lagrangian relaxation to scheduling in power-generation systems . *Operations Research.* 1977. vol. 25, n° 3, p. 387-403.

[5] Sayeed, Salam. "Unit Commitment Solution Methods''. Proceedings of world academy of science, engineering and technology. *International Journal of Electrical, Computer, Energetic, Electronic and Communication Engineering,* December, 2007, vol. 1, n° 11, p. 1638-1643.

[6] Sheble, G.B. and Fahd, G.N. "Unit Commitment Literature Synopsis", *IEEE Trans. Power Syst.* 1994, vol. 9, n° 1, p. 128-135.

[7] Juste, K.A., Kita, H., Tanaka, E. and Hasegawa, J. An evolutionary programming solution to the unit commitment problem. *IEEE Transactions on Power Systems*. 1999, vol. 4, n° 4, p. 1452-1459.

[8] Talbi, El-Ghazali. A Unified Taxonomy of Hybrid Metaheuristics with Mathematical Programming, Constraint Programming and Machine Learning.In: Hybrid Metaheuristics. *Studies in Computational Intelligence.* 2013, vol. 434, p. 3-76.

[9] Shivani, Mehta and Gourav, Kumar. Unit Commitment Using Hybrid Gravitational Search Algorithm, *International Journal of Research in Advent Technology*, April, 2016, vol. 4, n° 4, p. 12-20.

[10] Usha Rani, S. and Padmanabha Raju, C. H. A Solution to Unit Commitment Problem via Dynamic Programming and Particle Swarm Optimization. *International Journal of Current Engineering and Technology,* October, 2013, vol. 3, n° 4, p. 1495-1503.

[11] Kirkpatrick, S., Gelatt, C.D.Jr. and Vecchi, M. P. "Optimization by Simulated Annealing. *Science*, May, 1983, vol. 220, p. 671-680.

[12] Mantawy, A. H. A unit commitment by tabu search. *IEE Proc-Gener. Transm. Distrib,* January, 1998, vol. 145, n° 1, p. 56-64.

[13] Valenzuela, J. and Smith, A.E. A seeded memetic algorithm for large unit commitment problems. *Journal of Heuristics,* 2002, vol. 8, n° 2, p. 173-195.

[14] Kazarlis, S.A., Bakirtzis, A.G. and Petridis, V. A Genetic Algorithm Solution to the Unit Commitment Problem. *IEEE Transactions on Power Systems,* 1996, vol. 11, p. 83-92.

[15] Sihaj, P. Simon. An ant colony system approach for unit commitment problem. *Electrical Power and Energy Systems*, June, 2006, vol. 28, n° 5, p. 315-323.

[16] Cheng, Chuan-Ping. Unit Commitment by Lagrangian relaxation and genetic algorithms. *IEEE Transactions on Power Systems,* May, 2000, vol. 15, n°. 2, p. 707-714.

[17] Attaviriyanupap, Pathom. A hybrid LR–EP for solving new profit-based UC problem under competitive environment. *IEEE Transactions on Power Systems,* February, 2003, vol. 18, n°. 1, p. 229-237.

[18] Mohammedi, Ridha Djamel. *Étude du Problème d'Engagement de Turbines (Unit Commitment) par la Programmation Dynamique et autres Techniques Avancées.* 110 p. mémoire de magister : Electrotechnique : Laghouat,Université Amar Telidji : 2008.

[19] Su, Chung-Ching. and Hsu, Yuan-Yih. Fuzzy dynamic programming: an application to unit commitment. *Transactions on Power Systems,* August, 1991, vol. 6, n° 3, p. 1231-1237.

[20] Christober, C., Rajan, A. and Mohan, M. R. An evolutionary programming-based tabu search method for solving the unit commitment problem. *IEEE Transactions on Power Systems*, February, 2004, vol. 19, n° 1, p. 577-585.

[21] Govardhan, M. Solution to Unit Commitment with Priority List Approach. *IJAREEIE*, June, 2016, vol. 5, n° 6, p. 5114-5121.

[22] Govardhan, M. and Roy, R. Evolutionary computation based unit commitment using Hybrid Priority List approach. *IEEE International Conference on Power and Energy (PECon),* 2012, p. 245-250.

[23] Bellman, R. E. and Dreyfus, S. E. Applied dynamic programming. New Jersey : *Princeton University Press*, 1962. 335p. Prineton Legacy Library.

[24] Bhowmik, B. Dynamic programming– its principles, applications, strenghts, and limitations. *International Journal of Engineering Science and Technology*, 2010, vol. 2, n° 9, p. 4822-4826.

[25] Lowery, P. G. Generating unit commitment by dynamic programming. *IEEE Trans. on Power Apparatus and Systems,* 1966, vol. 85, n° 5, pp. 422-426.

[26] Ayoub, A. K. and Patton, A. D. Optimal thermal generating unit commitment. *IEEE Trans. on Power Apparatus and Systems,* 1971, vol. 90, n° 4, pp. 1752-1756.

[27] Pang, C. K. and Chen, H. C. Optimal short term thermal unit commitment. *IEEE Trans. on Power Apparatus and Systems,* 1976, vol. 95, n° 4, p. 1336-1346.

[28] Pang, C. K., Sheble, G. B. and Albuyeh, F. Evaluation of dynamic programming based methods and multiple area representation for thermal unit commitment. *IEEE Trans. on Power Apparatus and Systems,* 1981, vol. 100, n° 3, p. 1212-1218.

[29] Kirschen, Daniel. and the University of Washington. Unit Commitment. [en ligne]. [consulté le 05 mars 2018].
Disponible sur http://wwcw2.ee.washington.edu/research/real/Library/Teaching/05a-Unit_Commitment.pptx.

[30] Kumar Kamboj, V. and Bath, S.K. Proc. Single Area Unit Commitment using Dynamic Programming. *Emerging Trends in Engineering and Technology*, 2013, p. 930-936.

[31] Wang, C. and Shahidehpour, S.M. Effects of ramp-rate limits on unit commitment and economic dispatch. *IEEE Transactions on Power Systems,* 1993, vol. 8, n° 3, p. 1341-1350.

[32] IEEE. 118-bus, 54-unit, 24-hour system .[ en ligne]. [consulté le 20 février 2018].
Disponible sur : <motor.ece.iit.edu/Data/JEAS_IEEE118.doc>

[33] Sima Uyar, A. and Turkay, B. Evolutionary Algorithms for the Unit Commitment Problem. *Turk J Elec Engin,* 2008, vol. 16, n° 3, p. 239-255.

[34] Beni, G. and Wang, J. Swarm Intelligence in Cellular Robotic Systems: Robots and biological systems: towards a new bionics. *Proceedings of the NATO Advanced Workshop,* Tuscany, Italy, June 26-30, 1989, vol.102, p. 703-712. ISBN 10.1007/978-3-642-58069-7.

[35] Maifeld, Timothy Trent. *Genetic-based unit commitment algorithm*. Retrospective Theses and Dissertations, n° 10932. 110p .
Thèse de doctorat: Génie électrique : Ames ,Iowa State University :1995.

[36] Kazarlis, S., Bakirtzis, A. A. G. and Petridis, V. A Genetic Algorithm Solution to the Unit Commitment Problem. *IEEE Transactions on Power Systems*, 1996, vol. 11, n° 1, p. 83-92.

[37] Sheblé, G.B., Maifeld, T.T., Brittig, K. G., Fahd, S. and Fukurozaki-Coppinger. Unit commitment by genetic algorithm with penalty methods and a comparison of Lagrangian search and genetic algorithm—economic dispatch example. *International Journal of Electrical Power & Energy Systems*, 1996, vol. 18, n° 6, p. 339-346.

[38] Bukhari, S.B.A., Ahmad, A., Raza, S.A. and Siddique, M.N. A ring crossover genetic algorithm for the unit commitment problem. *Turk. J. Electr. Eng. Comput. Sc*, 2016, vol. 24, p. 3862-3876.

[39] Kennedy, J. and Eberhart, R. Particle swarm optimization. *Proceedings IEEE International Conference on Neural Networks (ICNN '95)*, Perth, WA, Australia, November–December, 1995, vol. 4, pp. 1942-1948.

[40] García-Gonzalo, E. and Fernández-Martínez, J. L. A Brief Historical Review of Particle Swarm Optimization (PSO). *Journal of Bioinformatics and Intelligent Control,* 2012, vol. 1, n°1, p. 3-16.

[41] Hassan, Rania and Cohanim, Babak and de Weck, Olivier. *A Comparison of Particle Swarm Optimization and the Genetic Algorithm.* 46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics and Materials Co-located Conferences, Texas, 18-21 April 2005.

[42] Kennedy, J. and Eberhart, R. A discrete binary version of the particle swarm algorithm, Systems, Man, and Cybernetics, *IEEE International Conference on Computational Cybernetics and Simulation,* October, 1997, vol. 5, p. 4104-4108.

[43] Santhi, R. K. and Subramanian, S. Adaptive Binary PSO based Unit Commitment. *International Journal of Computer Applications (0975 – 8887),* February, 2011, vol. 15, n° 4. p. 01-06.

[44] Menhas M.I, Fei M., Wang L. and Fu X. A Novel Hybrid Binary PSO Algorithm. In: Tan Y., Shi Y., Chai Y., Wang G. (eds) *Advances in Swarm Intelligence*. ICSI 2011. *Lecture Notes in Computer Science*, 2011, vol. 6728, p. 93-100. ISBN 978-3-642-21514-8.

[45] Valenzuela, J. and Smith, A. Seeded Memetic Algorithm for Large Unit Commitment Problems, A.E. Journal of *Heuristics*, 2002, vol. 8, n° 2, p. 173-195.

[46] Metropolis N., Rosenbluth A., Rosenbluth M., Teller A. and Teller E. Equations of state calculations by fast computing machines. *J Chem. Phys,* 1953, vol.21, p. 1087-1982.

[47] Pincus, M. A Monte Carlo method for the approximate solution of certain types of constrained optimization problems, *Operation Researhes*, 1970, vol. 18, p. 1225-1228.

[48] Cemy, V. Thermodynnamical Approach to the Traveling Salesman Problem An Efficient Simulation Algorithm, *Journal of Optimization Theory and Applications*, 1985, vol. 45, n° 1, p. 45-51.

[49] Saber, A. Y., Senjyu T., Miyagi T., N. Urasaki, T. Funabashi. Unit commitment by heuristics and absolutely stochastic simulated annealing. *IET Generation, Transmission & Distribution*, March 2007, vol. 1, no. 2, pp. 234-243.

# Annexe A : Données du système de 3 générateurs.

Tableau XII. Paramètres des générateurs du système de 3 unités [29].

| Unité | Pmin (MW) | Pmax (MW) | Temps minimum de démarrage (heures) | Temps minimum d'arrêt (heures) | Coût sans charge (£) | Coût marginal (£/MWh) | Coût de redémarrage | État initial |
|---|---|---|---|---|---|---|---|---|
| A | 150 | 250 | 3 | 3 | 0 | 10 | 1000 | ON |
| B | 50 | 100 | 1 | 2 | 0 | 12 | 600 | OFF |
| C | 10 | 50 | 1 | 1 | 0 | 20 | 100 | OFF |

# Annexe B : Données du système de 4 unités.

Tableau XIII. Paramètres des générateurs du système de 4 unités [30].

| Unité | Pmax MW | Pmin MW | C ($/MW^2) | B ($/MW) | A ($) | MUT (h) | MDT (h) | HSC (MW) | CSC (MW) | CS (h) | IS (h) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 300 | 75 | 0.0021 | 16.8300 | 684.7400 | 5 | 4 | 500 | 1100 | 5 | 8 |
| 2 | 250 | 60 | 0.0042 | 16.9500 | 585.6200 | 5 | 3 | 170 | 400 | 5 | 8 |
| 3 | 80 | 25 | 0.0018 | 20.7400 | 213.0000 | 4 | 2 | 150 | 350 | 4 | -5 |
| 4 | 60 | 20 | 0.0034 | 23.6000 | 252.0000 | 1 | 1 | 0 | 0 | 0 | -6 |

Tableau XIV. Charges du système de 4 unités [30].

| Heure | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Charge (MW) | 460 | 570 | 620 | 650 | 560 | 450 | 430 | 390 | 460 | 570 | 620 | 650 |

# Annexe C : Données du système de 10 unités.

Tableau XV. Paramètres des générateurs du système de 10 unités [30].

| Unité | Pmax MW | Pmin MW | C ($/MW^2) | B ($/MW) | A ($) | MUT (h) | MDT (h) | HSC (MW) | CSC (MW) | CS (h) | IS (h) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 455 | 150 | 0.0005 | 16.1900 | 1000 | 8 | 8 | 4500 | 9000 | 5 | 8 |
| 2 | 455 | 150 | 0.0003 | 17.2600 | 970 | 8 | 8 | 5000 | 10000 | 5 | 8 |
| 3 | 130 | 20 | 0.0020 | 16.6000 | 700 | 5 | 5 | 550 | 1100 | 4 | -5 |
| 4 | 130 | 20 | 0.0021 | 16.5000 | 680 | 5 | 5 | 560 | 1120 | 4 | -5 |
| 5 | 162 | 25 | 0.0040 | 19.7000 | 450 | 6 | 6 | 900 | 1800 | 4 | -6 |
| 6 | 80 | 20 | 0.0071 | 22.2600 | 370 | 3 | 3 | 170 | 340 | 2 | -3 |
| 7 | 85 | 25 | 0.0008 | 27.7400 | 480 | 3 | 3 | 260 | 520 | 2 | -3 |
| 8 | 55 | 10 | 0.0041 | 25.9200 | 660 | 1 | 1 | 30 | 60 | 0 | -1 |
| 9 | 55 | 10 | 0.0022 | 27.2700 | 665 | 1 | 1 | 30 | 60 | 0 | -1 |
| 10 | 55 | 10 | 0.0017 | 27.7900 | 670 | 1 | 1 | 30 | 60 | 0 | -1 |

Tableau XVI. Charges et réserves du système de 10 unités [30].

| Heure | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Charge (MW) | 700 | 750 | 850 | 950 | 1000 | 1100 | 1150 | 1200 | 1300 | 1400 | 1450 | 1500 |
| Réserve (MW) | 70 | 75 | 85 | 95 | 100 | 110 | 115 | 120 | 130 | 140 | 145 | 150 |
| Heure | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| Charge (MW) | 1400 | 1300 | 1200 | 1050 | 1000 | 1100 | 1200 | 1400 | 1300 | 1100 | 900 | 800 |
| Réserve (MW) | 140 | 130 | 120 | 105 | 100 | 110 | 120 | 140 | 130 | 110 | 90 | 80 |

Les systèmes de 20, 40, 60, 80 et 100 unités sont obtenus en multipliant les données des générateurs du système de 10 unités ainsi que les charges et les réserves.

# Annexe D : Données du système de 26 générateurs.

Tableau XVII. Données des générateurs du système de 26 unités [31].

| Units | Pmin | Pmax | C | B | A | IS | Startcost | Min up | Min down |
|---|---|---|---|---|---|---|---|---|---|
| 1 | 2,4 | 12 | 0,02533 | 25,5472 | 24,3891 | -1 | 0,01 | 1 | 1 |
| 2 | 2,4 | 12 | 0,02649 | 25,6753 | 24,411 | -1 | 0,01 | 1 | 2 |
| 3 | 2,4 | 12 | 0,02801 | 25,8027 | 24,6382 | -1 | 0,01 | 1 | 3 |
| 4 | 2,4 | 12 | 0,02842 | 25,9318 | 24,7605 | -1 | 0,01 | 1 | 4 |
| 5 | 2,4 | 12 | 0,02855 | 26,0611 | 24,8882 | -1 | 0,01 | 1 | 5 |
| 6 | 4 | 20 | 0,01199 | 37,551 | 117,7551 | -1 | 20 | 2 | 6 |
| 7 | 4 | 20 | 0,01261 | 37,6637 | 118,1083 | -1 | 20 | 2 | 7 |
| 8 | 4 | 20 | 0,01359 | 37,777 | 118,4576 | -1 | 20 | 2 | 8 |
| 9 | 4 | 20 | 0,01433 | 37,8896 | 118,8206 | -1 | 20 | 2 | 9 |
| 10 | 15,2 | 76 | 0,00876 | 13,3272 | 81,1364 | 3 | 50 | 3 | 10 |
| 11 | 15,2 | 76 | 0,00895 | 13,3538 | 81,298 | 3 | 50 | 3 | 11 |
| 12 | 15,2 | 76 | 0,0091 | 13,3805 | 81,4641 | 3 | 50 | 3 | 12 |
| 13 | 15,2 | 76 | 0,00932 | 13,4073 | 81,6259 | 3 | 50 | 3 | 13 |
| 14 | 25 | 100 | 0,00623 | 18 | 217,8952 | -3 | 70 | 4 | 14 |
| 15 | 25 | 100 | 0,00612 | 18,1 | 218,335 | -3 | 70 | 4 | 15 |
| 16 | 25 | 100 | 0,00598 | 18,2 | 218,7752 | -3 | 70 | 4 | 16 |
| 17 | 54,25 | 155 | 0,00463 | 10,694 | 142,7348 | 5 | 150 | 5 | 17 |
| 18 | 54,25 | 155 | 0,00473 | 10,7154 | 143,0288 | 5 | 150 | 5 | 18 |
| 19 | 54,25 | 155 | 0,00481 | 10,7367 | 143,3179 | 5 | 150 | 5 | 19 |
| 20 | 54,25 | 155 | 0,00487 | 10,7583 | 143,5972 | 5 | 150 | 5 | 20 |
| 21 | 68,95 | 197 | 0,00259 | 23 | 259,131 | -4 | 200 | 5 | 21 |
| 22 | 68,95 | 197 | 0,0026 | 23,1 | 259,649 | -4 | 200 | 5 | 22 |
| 23 | 68,95 | 197 | 0,00263 | 23,2 | 260,176 | -4 | 200 | 5 | 23 |
| 24 | 140 | 350 | 0,00153 | 10,8616 | 177,0575 | 10 | 300 | 8 | 24 |
| 25 | 100 | 400 | 0,00194 | 7,4921 | 310,0021 | 10 | 500 | 8 | 25 |
| 26 | 100 | 400 | 0,00195 | 7,5031 | 311,9102 | 10 | 500 | 8 | 26 |

Tableau XVIII. Charges et réserves du système de 26 unités [31].

| Heure | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Charge (MW) | 1700 | 1730 | 1690 | 1700 | 1750 | 1850 | 2000 | 2430 | 2540 | 2600 | 2670 | 2590 |
| Réserve (MW) | 400 | 400 | 400 | 400 | 400 | 400 | 400 | 400 | 400 | 400 | 400 | 400 |
| Heure | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
| Charge (MW) | 2590 | 2550 | 2620 | 2650 | 2550 | 2530 | 2500 | 2550 | 2600 | 2480 | 2200 | 1840 |
| Réserve (MW) | 400 | 400 | 400 | 400 | 400 | 400 | 400 | 400 | 400 | 400 | 400 | 400 |

# Annexe E : Données du système 118 nœuds de 54 unités.

Tableau XIX. Données des générateurs du système 118 nœuds/54 unités [32].

| U | Bus No. | Unit Cost Coefficients | | | Pmax (MW) | Pmin (MW) | Ini. State (h) | Min Off (h) | Min On (h) | Ramp (MW/h) | Start Up (MBtu) |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | a (MBtu) | b (MBtu/ MW) | c (MBtu/ $MW^2$) | | | | | | | |
| 1 | 4 | 31.67 | 26.2438 | 0.069663 | 30 | 5 | 1 | 1 | 1 | 15 | 40 |
| 2 | 6 | 31.67 | 26.2438 | 0.069663 | 30 | 5 | 1 | 1 | 1 | 15 | 40 |
| 3 | 8 | 31.67 | 26.2438 | 0.069663 | 30 | 5 | 1 | 1 | 1 | 15 | 40 |
| 4 | 10 | 6.78 | 12.8875 | 0.010875 | 300 | 150 | 8 | 8 | 8 | 150 | 440 |
| 5 | 12 | 6.78 | 12.8875 | 0.010875 | 300 | 100 | 8 | 8 | 8 | 150 | 110 |
| 6 | 15 | 31.67 | 26.2438 | 0.069663 | 30 | 10 | 1 | 1 | 1 | 15 | 40 |
| 7 | 18 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 8 | 19 | 31.67 | 26.2438 | 0.069663 | 30 | 5 | 1 | 1 | 1 | 15 | 40 |
| 9 | 24 | 31.67 | 26.2438 | 0.069663 | 30 | 5 | 1 | 1 | 1 | 15 | 40 |
| 10 | 25 | 6.78 | 12.8875 | 0.010875 | 300 | 100 | 8 | 8 | 8 | 150 | 100 |
| 11 | 26 | 32.96 | 10.7600 | 0.003000 | 350 | 100 | 8 | 8 | 8 | 175 | 100 |
| 12 | 27 | 31.67 | 26.2438 | 0.069663 | 30 | 8 | 1 | 1 | 1 | 15 | 40 |
| 13 | 31 | 31.67 | 26.2438 | 0.069663 | 30 | 8 | 1 | 1 | 1 | 15 | 40 |
| 14 | 32 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 15 | 34 | 31.67 | 26.2438 | 0.069663 | 30 | 8 | 1 | 1 | 1 | 15 | 40 |
| 16 | 36 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 17 | 40 | 31.67 | 26.2438 | 0.069663 | 30 | 8 | 1 | 1 | 1 | 15 | 40 |
| 18 | 42 | 31.67 | 26.2438 | 0.069663 | 30 | 8 | 1 | 1 | 1 | 15 | 40 |
| 19 | 46 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 59 |
| 20 | 49 | 28 | 12.3299 | 0.002401 | 250 | 50 | 8 | 8 | 8 | 125 | 100 |
| 21 | 54 | 28 | 12.3299 | 0.002401 | 250 | 50 | 8 | 8 | 8 | 125 | 100 |
| 22 | 55 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 23 | 56 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 24 | 59 | 39 | 13.2900 | 0.004400 | 200 | 50 | 10 | 8 | 8 | 100 | 100 |
| 25 | 61 | 39 | 13.2900 | 0.004400 | 200 | 50 | 10 | 8 | 8 | 100 | 100 |
| 26 | 62 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 27 | 65 | 64.16 | 8.3391 | 0.010590 | 420 | 100 | 10 | 10 | 10 | 210 | 250 |
| 28 | 66 | 64.16 | 8.3391 | 0.010590 | 420 | 100 | 10 | 10 | 10 | 210 | 250 |
| 29 | 69 | 6.78 | 12.8875 | 0.010875 | 300 | 80 | 10 | 8 | 8 | 150 | 100 |
| 30 | 70 | 74.33 | 15.4708 | 0.045923 | 80 | 30 | 4 | 4 | 4 | 40 | 45 |
| 31 | 72 | 31.67 | 26.2438 | 0.069663 | 30 | 10 | 1 | 1 | 1 | 15 | 40 |
| 32 | 73 | 31.67 | 26.2438 | 0.069663 | 30 | 5 | 1 | 1 | 1 | 15 | 40 |
| 33 | 74 | 17.95 | 37.6968 | 0.028302 | 20 | 5 | 1 | 1 | 1 | 10 | 30 |
| 34 | 76 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 35 | 77 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 36 | 80 | 6.78 | 12.8875 | 0.010875 | 300 | 150 | 10 | 8 | 8 | 150 | 440 |
| 37 | 82 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 38 | 85 | 31.67 | 26.2438 | 0.069663 | 30 | 10 | 1 | 1 | 1 | 15 | 40 |
| 39 | 87 | 32.96 | 10.7600 | 0.003000 | 300 | 100 | 10 | 8 | 8 | 150 | 440 |
| 40 | 89 | 6.78 | 12.8875 | 0.010875 | 200 | 50 | 10 | 8 | 8 | 100 | 400 |

| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 41 | 90 | 17.95 | 37.6968 | 0.028302 | 20 | 8 | 1 | 1 | 1 | 10 | 30 |
| 42 | 91 | 58.81 | 22.9423 | 0.009774 | 50 | 20 | 1 | 1 | 1 | 25 | 45 |
| 43 | 92 | 6.78 | 12.8875 | 0.010875 | 300 | 100 | 8 | 8 | 8 | 150 | 100 |
| 44 | 99 | 6.78 | 12.8875 | 0.010875 | 300 | 100 | 8 | 8 | 8 | 150 | 100 |
| 45 | 100 | 6.78 | 12.8875 | 0.010875 | 300 | 100 | 8 | 8 | 8 | 150 | 110 |
| 46 | 103 | 17.95 | 37.6968 | 0.028302 | 20 | 8 | 1 | 1 | 1 | 10 | 30 |
| 47 | 104 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 48 | 105 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 49 | 107 | 17.95 | 37.6968 | 0.028302 | 20 | 8 | 1 | 1 | 1 | 10 | 30 |
| 50 | 110 | 58.81 | 22.9423 | 0.009774 | 50 | 25 | 2 | 2 | 2 | 25 | 45 |
| 51 | 111 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 52 | 112 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 53 | 113 | 10.15 | 17.8200 | 0.012800 | 100 | 25 | 5 | 5 | 5 | 50 | 50 |
| 54 | 116 | 58.81 | 22.9423 | 0.009774 | 50 | 25 | 2 | 2 | 2 | 25 | 45 |

Tableau XX. Charges et réserves du système 118 nœuds/54 unités [32].

| H | Real Load (MW) | Spinning Reserve (MW) |
|---|---|---|
| 1 | 4200 | 210 |
| 2 | 3960 | 198 |
| 3 | 3480 | 174 |
| 4 | 2400 | 120 |
| 5 | 3000 | 150 |
| 6 | 3600 | 180 |
| 7 | 4200 | 210 |
| 8 | 4680 | 234 |
| 9 | 4920 | 246 |
| 10 | 5280 | 264 |
| 11 | 5340 | 267 |
| 12 | 5040 | 252 |
| 13 | 4800 | 240 |
| 14 | 4560 | 228 |
| 15 | 5280 | 264 |
| 16 | 5400 | 270 |
| 17 | 5100 | 255 |
| 18 | 5340 | 267 |
| 19 | 5640 | 282 |
| 20 | 5880 | 294 |
| 21 | 6000 | 300 |
| 22 | 5400 | 270 |
| 23 | 5220 | 261 |
| 24 | 4920 | 246 |