PolyPublie
Polytechnique Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Titre: Title:	Développement d'un algorithme génétique pour l'optimisation de réseaux d'échangeurs de chaleur
Auteur: Author:	Jean Dipama
Date:	2003
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Dipama, J. (2003). Développement d'un algorithme génétique pour l'optimisation de réseaux d'échangeurs de chaleur [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/7245/

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/7245/
Directeurs de recherche: Advisors:	Alberto Teyssedou, & Mikhaïl Sorin
Programme: Program:	Unspecified

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal

https://publications.polymtl.ca

UNIVERSITÉ DE MONTRÉAL

DÉVELOPPEMENT D'UN ALGORITHME GÉNÉTIQUE POUR L'OPTIMISATION DE RÉSEAUX D'ÉCHANGEURS DE CHALEUR

DIPAMA JEAN
DÉPARTEMENT DE GÉNIE PHYSIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE ÉNERGÉTIQUE)
DÉCEMBRE 2003

© Dipama Jean, 2003

National Library of Canada
Bibliothèque nationale du Canada

Acquisitions and Bibliographic Services
Acquisisitons et services bibliographiques

395 Wellington Street
Ottawa ON K1A 0N4
Canada

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 0-612-89196-8
Our file Notre référence
ISBN: 0-612-89196-8

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of this thesis in microform, paper or electronic formats.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de cette thèse sous la forme de microfiche/film, de reproduction sur papier ou sur format électronique.

The author retains ownership of the copyright in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

L'auteur conserve la propriété du droit d'auteur qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou aturement reproduits sans son autorisation.

In compliance with the Canadian Privacy Act some supporting forms may have been removed from this dissertation.

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de ce manuscrit.

While these forms may be included in the document page count, their removal does not represent any loss of content from the dissertation.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

DÉVELOPPEMENT D'UN ALGORITHME GÉNÉTIQUE POUR L'OPTIMISATION DE RÉSEAUX D'ÉCHANGEURS DE CHALEUR

Présenté par : DIPAMA Jean

en vue de l'obtention du diplôme de : Maîtrise ès Sciences appliquées

a été dûment accepté par le jury d'examen constitué de :

M. ZIKOVSKY Lubomir, Ph.D., président

M. TEYSSEDOU Alberto, Ph.D., membre et directeur de recherche

M. SORIN Mikhaïl, Ph.D., membre et codirecteur de recherche

M. BERNIER Michel, Ph.D., membre

REMERCIEMENTS

Je voudrais exprimer mes sincères remerciements à mon directeur de recherche, M. Alberto TEYSSEDOU et à mon codirecteur de recherche, M. Mikhaïl SORIN, pour leur encadrement sans faille qui a permis la réalisation de ce travail. Qu'ils trouvent ici, l'expression de ma profonde gratitude.

C'est l'occasion pour moi également de remercier le personnel du centre de la technologie de l'énergie du CANMET, pour les échanges fructueux que nous avons eus ensemble dans le cadre de ce projet.

Je voudrais témoigner ma reconnaissance à l'Agence Canadienne de Développement International (ACDI), pour avoir financé mes études de Maîtrise.

Au Ministère des Enseignements Secondaire, Supérieur et de la Recherche Scientifique du Burkina Faso, je dis merci, pour l'opportunité qui m'a été offerte de bénéficier de la bourse de l'ACDI pour entreprendre ces études.

Pour terminer, je tiens à remercier mes collègues étudiants, le personnel de l'institut de génie nucléaire, et tous ceux qui, d'une manière ou d'une autre m'ont apporté leur soutien tout au long de mes études.

RÉSUMÉ

La conception des réseaux d'échangeurs de chaleur pour la récupération d'énergie dans les procédés industriels est un problème combinatoire complexe qui requiert des choix judicieux des courants à apparier pour obtenir un réseau optimal. Différentes approches, notamment thermodynamiques et mathématiques, ont été développées pour résoudre ce type de problème. Récemment, des chercheurs ont combiné l'approche des algorithmes génétiques et la programmation mathématique, afin de synthétiser des réseaux d'échangeurs de chaleur pour la récupération d'énergie.

Un algorithme génétique est une technique stochastique d'optimisation s'inspirant des mécanismes de sélection naturelle. C'est une procédure itérative qui simule l'évolution d'une population à travers plusieurs générations. À chaque génération, les individus de la population sont évalués et les meilleurs sont sélectionnés pour constituer la prochaine génération. Ainsi les individus sélectionnés sont soumis à un croisement et à une mutation avec l'espoir que la génération suivante sera meilleure que la précédente. À la fin des générations, le meilleur individu de la population est retenu et représente la solution au problème posé.

L'objectif de notre projet est d'explorer la possibilité de l'utilisation exclusive des algorithmes génétiques pour optimiser aussi bien la topologie du réseau que la distribution des charges des échangeurs de chaleur. Pour cela, un modèle de simulation de la récupération de chaleur a été élaboré et un algorithme d'optimisation basé sur les procédures des algorithmes génétiques a été écrit et codifié en FORTRAN 90.

L'algorithme que nous avons développé supporte les problèmes de synthèse de réseaux d'échangeurs de chaleurs qui peuvent être résolus sans recours à la division des courants.

Nous avons vérifié la validité de notre programme en résolvant des problèmes de réseaux d'échangeurs de chaleur que l'on retrouve dans la littérature. Les résultats obtenus sont comparables à ceux publiés dans la littérature .

La facilité de mise en œuvre et l'habilité à prendre en compte divers types de contraintes montrent que les algorithmes génétiques peuvent être une alternative pour la résolution des problèmes d'optimisation, surtout pour les problèmes combinatoires qui sont difficilement traités par les procédures mathématiques.

ABSTRACT

The synthesis of heat exchanger networks constitutes a very complex task. In order to improve the overall heat recovery in an existing industrial process, the synthesis must solve a combinatorial problem that must satisfy the best matching between hot and cold streams of the network. For solving such a problem, different approaches, i.e., thermodynamics and mathematical models, have been used with some success in the past. Recently, researchers have combined linear mathematical programming techniques with genetic algorithm methods for optimizing heat exchangers networks having a limited number of flow streams and thermal units.

A genetic algorithm is a stochastic optimization technique that was developed based on the theory of the natural selection of biological species. It's an iterative process that simulates the evolution of a population through several generations. In each generation, individuals of a population are evaluated and the best individuals are selected to reproduce the next generation. Thus, the selected individuals are submitted to crossover and mutation operations in order to improve the population average. After the number of predefined iterations is over and/or a pre established fitness is reached, the best individual in the population is considered as the best solution for the problem.

The aim of this Master's project consists in exploring the possibility of using exclusively genetic algorithms for both, the topological and the parametric optimization of large heat exchanger networks. Thus, we have developed a model capable of simulating the maximum heat recovery of complex industrial process and we have written an optimization algorithm entirely based on genetic algorithms. The software has been developed by using the FORTRAN 90 language. We must mention, however, that the proposed software is limited to the synthesis of heat exchanger networks that do not require the splitting of streams.

The algorithm has been tested by treating several heat exchanger network problems encountered in the open literature. In general, the results obtained with the proposed package are quite similar to those obtained by using more complex and time consuming techniques, i.e., the simplex method. The easy implementation of genetic algorithms and their ability to handle different thermodynamic constraints, have shown that they can be used as an efficient method for optimizing quite complex heat exchanger networks. In general, we have demonstrated that genetic algorithms are a powerful tool for solving combinatorial optimization problems that are difficult to handle by other mathematical methods.

TABLE DES MATIÈRES

REMERCIEMENTS iv

RÉSUMÉ v

ABSTRACT vii

TABLE DES MATIÈRES ix

LISTE DES TABLEAUX xiii

LISTE DES FIGURES xiv

LISTE DES SIGLES ET ABRÉVIATIONS xvii

LISTE DES ANNEXES xxii

CHAPITRE I 1

1 INTRODUCTION 1

1.1 Problématique de la réduction de la consommation d'énergie 1

1.2 La récupération de chaleur 1

1.3 Contribution du présent ouvrage: 3

1.4 Organisation du mémoire 4

CHAPITRE II 6

2 HISTORIQUE 6

2.1 Les méthodes thermodynamiques 6

2.1.1 Détermination de la consommation minimale d'énergie 8

2.1.2 Détermination du nombre minimum d'unités 9

2.1.3 Détermination de la surface minimale de transfert de chaleur 10

2.2 Les méthodes mathématiques 12
2.3 Les méthodes basées sur l'intelligence artificielle 13

CHAPITRE III 15

3 LES ECHANGEURS DE CHALEUR 15

3.1 Introduction 15
3.2 Géométrie de construction des échangeurs de chaleur : 15
3.3 Les principaux modes d'écoulement 16
3.4 Équations fondamentales 17
3.5 Profils de température selon le mode d'écoulement 21
3.6 Conditions spéciales de fonctionnement des échangeurs de chaleur 23
3.7 Concept de température minimale d'approche d'un échangeur de chaleur 24

CHAPITRE IV 26

4 SYNTHÈSE D'UN RÉSEAU D'ÉCHANGEURS DE CHALEUR PAR LA MÉTHODE DU PINCEMENT 26

4.1 Introduction 26
4.2 La collecte des données de l'installation : 27
4.3 Détermination de la puissance maximale récupérable 30
4.3.1 Choix de la température minimale d'approche 30
4.3.2 Construction du diagramme des courbes composites 30
4.3.3 Influence de la température minimale sur la récupération de chaleur 34
4.4 Procédure de conception d'un REC 37
4.4.1 Décomposition du système au point de pincement 37
4.4.2 Conception séparée d'un REC de chaque côté du pincement 40
4.4.3 Fusion des deux réseaux séparés 47
4.4.4 Optimisation du réseau initial 48
4.5 Conclusion 56

CHAPITRE V 57

5 SYNTHÈSE DE RÉSEAUX D'ÉCHANGEURS DE CHALEUR PAR LES ALGORITHMES GÉNÉTIQUES 57

5.1 Introduction 57

5.2 De la sélection naturelle à l'optimisation des systèmes industriels 57

5.3 Organisation des algorithmes génétiques 59

5.4 Fonctionnement des algorithmes génétiques 60

5.4.1 Génération de la population initiale 60

5.4.2 Évaluation des solutions potentielles 61

5.4.3 La sélection 61

5.4.4 Le croisement 65

5.4.5 La mutation 67

5.4.6 L'élitisme 68

5.5 Synthèse de réseaux d'échangeurs de chaleur par les algorithmes génétiques 68

5.5.1 Codage de la structure d'un REC 69

5.5.2 Espace de recherche des structures de REC 72

5.5.3 Application de l'opérateur de croisement sur les REC 73

5.5.4 Application de l'opérateur de mutation sur les REC 74

5.5.5 Codage de la distribution de charge d'un REC 75

5.5.6 Décodage du chromosome de distribution de charge 76

5.5.7 Choix de la taille de la population 77

5.5.8 Modèle mathématique de simulation de la récupération de chaleur 77

5.5.9 Méthode de la fonction pénalité 82

5.5.10 Processus de sélection pour la reproduction 88

5.6 Élaboration du programme d'optimisation 89

5.6.1 Recherche de la topologie du REC 90

5.6.2 Optimisation de la distribution des charges 91

5.6.3 Conditions d'arrêt de l'algorithme de recherche 92

5.6.4 Aperçu sur la méthode du 'Problem Table Algorithm' 92
5.7 Optimisation de la distribution de charges par la méthode du simplexe 97
5.7.1 Forme standard d'un programme linéaire 101
5.7.2 Le Tableau du simplexe 102
5.7.3 Solution de base faisable 103
5.7.4 Méthode des deux phases 104
5.8 Conclusion 107

CHAPITRE VI 108

6 EXEMPLES D' APPLICATION 108

6.1 Étude de cas 1 108
6.1.1 Optimisation par l'approche Génétique/Simplexe 109
6.1.2 Optimisation par l'approche Génétique/Génétique 116
6.2 Étude de cas 2 123
6.2.1 Résultats de test pour un niveau 124
6.2.2 Résultats de test pour trois niveaux 125
6.3 Étude de cas 3 128
6.4 Conclusion 130

CHAPITRE VII 131

7 CONCLUSION 131

7.1 Les performances de la méthode 131
7.2 Le réglage des paramètres de l'algorithme génétique 132
7.3 Insuffisances de la méthode 134
7.4 Propositions d'amélioration 134

RÉFÉRENCES 135

ANNEXES 141

LISTE DES TABLEAUX

Tableau 4.1 : Caractéristiques des courants 29
Tableau 4.2 : Valeurs expérimentales de ΔTmin 36
Tableau 4.3 : Caractéristiques des courants au-dessus du pincement 41
Tableau 4.4 : Matrice des *CP* au-dessus du pincement 42
Tableau 4.5 : Charges thermiques résiduelles au-dessus du pincement 43
Tableau 4.6 : Charges non récupérables au-dessus du pincement 44
Tableau 4.7 : Caractéristiques des courants au-dessous du pincement 45
Tableau 4.8 : Matrice des CP au-dessous du pincement 45
Tableau 4.9 : Charges thermiques résiduelles (système au-dessous du pincement) 46
Tableau 4.10 : Charges non récupérables au-dessous du pincement 46
Tableau 5.1 : Matrice d'incidence 70
Tableau 5.2 : Températures modifiées du PTA 93
Tableau 5.3 : Tableau du simplexe 102
Tableau 6.1 : Caractéristiques des courants : Étude de cas 1 109
Tableau 6.2 : Tableau final du simplexe (1e solution) 112
Tableau 6.3 : Tableau final du simplexe (2e solution alternative) 114
Tableau 6.4 : Tableau final du simplexe (3e solution alternative) 114
Tableau 6.5 : Solutions par l'approche G/S 115
Tableau 6.6 : Solutions par l'approche G/G 119
Tableau 6.7 : Caractéristiques des courants : Étude de cas 2 123
Tableau 6.8 : Solutions par l'approche G/G ($N_v = 1$) 124
Tableau 6.9 : Solutions par l'approche G/G ($N_v = 3$) 126
Tableau 6.10 : Caractéristiques des courants : Étude de cas 3 128
Tableau 6.11 : Solutions par l'approche G/G ($N_v = 2$) 129

LISTE DES FIGURES

Figure 3.1: Types d'échangeurs de chaleur en fonction de l'écoulement des fluides ... 16
Figure 3.2 : Mécanisme de transfert de chaleur ... 17
Figure 3.3 : Distribution de température dans un échangeur de chaleur ... 19
Figure 3.4 : Profils de température d'un échangeur à contre-courants ... 21
Figure 3.5 : Profil de température d'un échangeur à courants parallèles ... 22
Figure 3.6 : Conditions spéciales de fonctionnement d'un échangeur de chaleur ... 24
Figure 3.7 : Courbes d'échange de chaleur faisables et infaisables ... 25
Figure 4.1 : Diagramme de réseau d'échangeurs de chaleur ... 27
Figure 4.2 : Exemple de procédé industriel ... 29
Figure 4.3 : Construction de la courbe composite des courants chauds ... 31
Figure 4.4 : Construction de la courbe composite des courants froids ... 32
Figure 4.5 : Diagramme des courbes composites ... 33
Figure 4.6 : Influence de la température minimale sur la récupération de chaleur ... 35
Figure 4.7 : Compromis entre l'investissement et la récupération de chaleur ... 36
Figure 4.8 : Décomposition au pincement ... 37
Figure 4.9 : Règles de conception de REC par la méthode du pincement ... 39
Figure 4.10 : Caractéristiques des courants au-dessus et au-dessous du pincement ... 40
Figure 4.11 : Appariement des courants au pincement (système au-dessus) ... 42
Figure 4.12 : Appariement final des courants au-dessus du pincement ... 43
Figure 4.13 : Ajout d'une utilité chaude au-dessus du pincement ... 44
Figure 4.14 : Appariement des courants au pincement (système au-dessous) ... 45
Figure 4.15 : Appariement final des courants au-dessous du pincement ... 46
Figure 4.16 : Ajout d'une utilité froide au-dessous du pincement ... 47
Figure 4.17 : Réseau initial pour la récupération de chaleur ... 48
Figure 4.18 : Division des courants ... 49
Figure 4.19 : Boucles de charge de thermique ... 52

Figure 4.20 : Déplacement d'une charge thermique autour de la boucle 53
Figure 4.21 : Chemin de charge thermique 54
Figure 4.22 : Restauration de la température minimale d'approche. 55
Figure 5.1 : Structure d'un chromosome..... 59
Figure 5.2 : Cible divisée en secteurs proportionnels au fitness 63
Figure 5.3 : Attribution des fitness de classement 64
Figure 5.4 : Cible divisée en secteurs proportionnels aux fitness de classement..... 65
Figure 5.5 : Principe du croisement de deux chromosomes 66
Figure 5.6 : Principe de la mutation d'un chromosome..... 67
Figure 5.7 : Concept de niveau dans la représentation d'un REC..... 69
Figure 5.8 : Structure d'un réseau d'échangeurs de chaleur 71
Figure 5.9 : Croisement de deux chromosomes de REC 73
Figure 5.10 : Croisement de deux structures de REC..... 73
Figure 5.11 : Mutation d'un chromosome de REC 74
Figure 5.12 : Mutation de la structure de REC 74
Figure 5.13 : Représentation d'un chromosome de distribution de charge 75
Figure 5.14 : Structure complète d'un REC..... 78
Figure 5.15 : Les paramètres du REC..... 79
Figure 5.16 : Charge maximale d'un échangeur de chaleur du REC 85
Figure 5.17 : Espaces des solutions faisables et infaisables 86
Figure 5.18 : Mécanisme de recherche de la topologie optimale du REC..... 90
Figure 5.19 : Mécanisme de recherche de la distribution de charge du REC 91
Figure 5.20 : Bilan d'énergie par intervalle de température 94
Figure 5.21 : Cascade du surplus d'énergie 95
Figure 5.22 : Ajout d'une source chaude au système..... 96
Figure 6.1 : Convergence de la solution avec élitisme..... 110
Figure 6.2 : Convergence de la solution sans élitisme..... 111
Figure 6.3 : Solution pour la meilleure structure de REC (étude de cas 1) 111
Figure 6.4 : Réseau d'échangeurs de chaleur (solution G/S)..... 116

Figure 6.5 : Convergence de la solution (approche G/G) 117
Figure 6.6 : Influence du coefficient de pénalisation sur la convergence..... 118
Figure 6.7 : Effets de la mutation sur le niveau de pénalisation 120
Figure 6.8 : Effets du croisement sur le niveau de pénalisation..... 122
Figure 6.9 : Configuration du meilleur REC pour $N_v = 1$ (étude de cas 2)..... 125
Figure 6.10 : Configuration du meilleur REC pour $N_v = 3$ (étude de cas 2)..... 127
Figure 6.11 : Structure optimale du REC (étude de cas 3) 130

LISTE DES SIGLES ET ABRÉVIATIONS

Abréviations

AG : Algorithme Génétique

EMAT : Exchanger Minimum Approach Temperature

G/G : Optimisation structurale par les algorithmes génétiques/optimisation paramétrique par les algorithmes génétiques

G/S : Optimisation structurale par les algorithmes génétiques/optimisation paramétrique par le simplexe

GCC : Grand Composite Curve (Grande courbe composite)

LMTD : Log-Mean Temperature Difference (Moyenne logarithmique de la différence de température)

MILP : Mixed Integer Linear Programming (Modèle mixte de programme linéaire et des variables entières)

MINLP: Mixed Integer Non Linear Programming (Modèle mixte de programme non linéaire et des variables entières)

PTA : Problem Table Algorithm (Algorithme sous forme de tableau)

REC : Réseau d'échangeurs de chaleur

Sigles

τ : Taux de sélection d'un chromosome

λ : Longueur du chromosome

α : Nombre aléatoire généré par l'algorithme

δ : Précision des solutions

Φ : Fonction de pénalisation d'un échangeur de chaleur

ΔH : Variation d'enthalpie

ΔT : Différence de température entre un courant chaud et un courant froid

ΔT_k^i : Différence de température entre les courants chaud et froid à l'extrémité chaude de l'échangeur de chaleur k

ΔT_k^o : Différence de température entre les courants chaud et froid à l'extrémité froide de l'échangeur de chaleur k

ΔT_m : Moyenne logarithmique de la différence de température entre un courant chaud et un courant froid dans un échangeur de chaleur

ΔT_{min} : Température minimale d'approche

Δq : Quantité de charge thermique

A : Surface d'échange d'un échangeur de chaleur

A_{min} : Surface minimale d'un échangeur de chaleur

b : Valeur d'une position dans le codage d'un chromosome (binaire, entier, etc.)

C : Désignation d'un courant froid

Ck : Coefficient de pénalisation

cp : Chaleur massique

CP : Capacité calorifique

CP_{in} : Capacité calorifique de courant arrivant au pincement

CP_{out} : Capacité calorifique du courant quittant le pincement

dA : Variation élémentaire de la surface d'échange

dl : Variation élémentaire de la longueur

dq : Variation élémentaire de la charge thermique

dT : Variation de la température d'un courant

$dT_{h,k}$: Variation de la température du courant chaud à travers l'échangeur de chaleur k

$dT_{c,k}$: Variation de la température du courant froid à travers l'échangeur de chaleur k

f : Fonction objectif (ou fitness)

F : Fonction objectif composite combinant la fonction objectif f et la pénalité P

f^* : Fonction auxiliaire dans la méthode des deux phases

h : Enthalpie

H : Désignation d'un courant chaud

H/C : Appariement entre un courant chaud H et un courant froid C

hf : Coefficient de transfert de film du courant

I : Nombre total d'intervalles d'enthalpie

Kc : Position de croisement dans le chromosome

Km : Position de mutation dans le chromosome

Lp : Nombre de boucles

m : Débit massique d'un courant chaud

moy : Moyenne des fitness

n : Nombre de variables comprenant les puissances des échangeurs de chaleur et des utilités

N : Nombre total des courants et des utilités

n_b : Longueur d'un gène dans le codage binaire

N_c : Nombre de courants froids

N_{EC} : Nombre maximal d'échangeurs de chaleur admissible dans le réseau

N_{ECmin} : Nombre d'échangeurs de chaleur sur le courant de charge minimale auquel est connecté l'échangeur de chaleur k considéré

N_h : Nombre de courants chauds

$N_{inéq}$: Nombre total des contraintes inégalité

N_{REC} : Nombre de réseaux d'échangeurs de chaleur

N_{st} : Nombre total des courants chauds et froids

N_U : Nombre de types d'utilités

N_v : Nombre de niveaux dans le codage de la structure d'un réseau d'échangeurs de

$N_{x,j}$: Nombre d'échangeurs de chaleurs connectés à un courant j

P : Terme de pénalisation du réseau d'échangeurs de chaleur

Pc : Probabilité de croisement

Pm : Probabilité de mutation

pop : Taille de la population

Q ou q : Charge thermique d'un courant

QC_{min} : Puissance minimale requise des utilités froides

Q_{extra} : Quantité de chaleur transférée à travers le pincement, ajoutée ou soustraite des systèmes au-dessus et au-dessous du pincement

QH_{min} : Puissance minimale requise des utilités chaudes

Q_{REC} : Puissance récupérée par les échangeurs de chaleur

QU : Puissance d'une utilité

r : Coefficient de réparation de la borne supérieure de la variable X

S : Nombre de sous-systèmes

SOM : Somme des fitness

T : Température du courant

T_{mod} : Température modifiée

Th_k^i : Température du courant chaud entrant dans l'échangeur de chaleur k

Th_k^o : Température du courant chaud sortant de l'échangeur de chaleur k

Tc_k^i : Température du courant froid entrant dans l'échangeur de chaleur k

Tc_k^o : Température du courant froid sortant de l'échangeur de chaleur k

T^T : Température finale du courant

T^S : Température initiale du courant

U : coefficient global de transfert de chaleur

U_{min} : Nombre minimum d'unités

X : Puissance d'un échangeur ou d'une utilité

X' : Conversion dans la base 10 du codage binaire d'une variable X

X_{inf} : Borne inférieure de la variable X

$(X_k)_j$: Puissance de l'échangeur de chaleur k connecté au courant j

X_{sup} : Borne supérieure de la variable X

y : Variable auxiliaire dans la méthode des deux phases.

- **Indices**

c : Courant froid

h : Courant chaud

- **Exposants**

i : Côté chaud de l'échangeur de chaleur

o : Côté froid de l'échangeur de chaleur

S : Caractéristique initiale du courant

T : Caractéristique finale du courant

LISTE DES ANNEXES

ANNEXE 1 : Code de l'algorithme d'optimisation (approche G/G) 144
ANNEXE 2 : Code de l'algorithme d'optimisation (approche G/S)..... 180

CHAPITRE I

1 INTRODUCTION

1.1 Problématique de la réduction de la consommation d'énergie

Depuis quelques années, le secteur industriel connaît de fortes mutations structurelles et technologiques dont les effets sur la consommation d'énergie sont tangibles. Pour être plus compétitive sur le marché, il devient impérieux pour l'industrie de maîtriser ses consommations d'énergie. Les coûts de l'énergie dans l'industrie pèsent pour une part non négligeable (1 à 30%) des coûts de fabrication (Bernard, 1995) [1].

Une grande partie des coûts de l'énergie est consacrée à l'approvisionnement en combustibles (gaz, pétrole, etc.). Plus la demande en énergie est élevée plus les entreprises ont tendance à accroître leurs approvisionnement en énergie. Cependant, certains procédés industriels mettant en œuvre des fluides qui sont chauffés ou refroidis (industries chimique, métallurgique, de raffinerie, etc.) offrent des opportunités d'économiser l'énergie en procédant à une récupération de la chaleur. De bons choix technologiques et l'instauration de procédés efficaces peuvent permettre aux entreprises d'optimiser leur consommation énergétique, de réduire leurs coûts de production, et de limiter les émissions des gaz à effet de serre.

1.2 La récupération de chaleur

Une des méthodes employées dans le secteur industriel pour réduire les consommations d'énergie est la récupération de chaleur. La mise en place d'un réseau d'échangeurs de chaleur est d'autant plus complexe que le nombre de courants mis en œuvre dans les

procédés de transformation est souvent énorme. Synthétiser un réseau d'échangeurs de chaleur (REC) optimal (en termes de coût d'investissement et d'opération) a depuis longtemps été la préoccupation des chercheurs. La découverte du concept de ***pincement*** par Linnhoff et Hindmarsh (1983) [2], a été d'un apport important et fondamental dans la compréhension des problèmes liés à la récupération de chaleur. Ce concept de ***température de pincement*** a donné naissance à une méthodologie complète nommée la ***technologie du pincement***. À travers l'analyse du pincement, il est possible de connaître, avant même d'entreprendre la conception du réseau, la quantité de chaleur qui peut être récupérée, le nombre minimum d'unités (échangeurs de chaleur, refroidisseurs, réchauffeurs) dont on a besoin et la surface totale minimale nécessaire des échangeurs de chaleur. Il est aussi possible, en connaissant la quantité de chaleur récupérée, le nombre d'unités requis et la surface totale nécessaire des échangeurs de chaleur, d'estimer le ***coût total annuel*** de l'opération de récupération de chaleur.

Avec le concept de pincement, le design des REC est devenu systématique et méthodique. Il offre une stratégie et des règles pour la conception d'un REC optimal, en vue de la récupération de chaleur. Des méthodes mathématiques (la programmation mathématique) intégrant certains concepts thermodynamiques comme le pincement ont également permis de réaliser la synthèse de REC optimisés. Mais la complexité du problème d'optimisation des REC impliquant simultanément des variables discrètes et continues n'est pas toujours facile à résoudre par les procédures mathématiques.

De nos jours, le développement de l'informatique offre des outils puissants et souples pour réaliser la synthèse des REC. La méthode des algorithmes génétiques (AG) introduite par Holland (1975) [3], est une méthode stochastique d'optimisation des systèmes. Elle simule l'évolution d'une population d'individus à travers plusieurs générations à laquelle on applique différents opérateurs (croisement, mutation) et que l'on soumet à une sélection, à chaque génération. Les meilleurs individus étant sélectionnés à chaque génération, la population ne peut qu'évoluer vers un état optimal. C'est

une procédure itérative qui requiert l'outil informatique pour la résolution des problèmes d'optimisation. Dans cet ouvrage nous présentons une approche de synthèse de REC basée sur la méthode des algorithmes génétiques.

1.3 Contribution du présent ouvrage:

La synthèse d'un REC est un problème combinatoire complexe mettant en œuvre des variables discrètes de décision (appariement des courants) et des variables continues (charges thermiques des échangeurs) qu'il faut optimiser.

L'objectif du présent ouvrage est de proposer une démarche fondée sur les algorithmes génétiques, pour synthétiser des réseaux d'échangeurs de chaleur. Un modèle mathématique est de ce fait développé, qui est compatible pour l'application des AG. Nos travaux sont inspirés de l'ouvrage de Lewin et al., (1998a) [4] qui propose une méthode de synthèse de REC sans division de courants en utilisant les algorithmes génétiques. Dans ses travaux, Lewin propose une représentation compacte de la structure d'un REC mais de dimension constante, qui sert à l'application des AG pour l'optimisation structurale (topologie) et qui peut aussi être mise sous une forme convenable pour l'optimisation paramétrique. Le modèle de simulation qu'il a développé à partir de cette représentation pour l'optimisation paramétrique est un programme linéaire. Un programme linéaire comporte une fonction objectif qui traduit les objectifs de l'optimisation (minimisation ou maximisation d'une fonction) et des contraintes. Le problème est résolu à deux niveaux : Au premier niveau, la détermination de la topologie du REC est assurée par les algorithmes génétiques. Ensuite cette topologie est envoyée au second niveau pour l'optimisation paramétrique à l'aide des procédures mathématiques, notamment la méthode du simplexe.

Le modèle de programmation linéaire ne peut être traité directement par les algorithmes génétiques. Dans notre approche, nous reprenons la démarche développée par Lewin

pour la représentation de la structure de REC, mais nous proposons un modèle pour l'optimisation paramétrique qui est adapté aux algorithmes génétiques. La topologie du réseau ainsi que les charges des échangeurs de chaleur peuvent donc être entièrement optimisées par les algorithmes génétiques . Le modèle développé transforme le problème avec contraintes en un problème sans contraintes, par l'incorporation dans la fonction objectif d'une pénalité qui dégrade la qualité de la fonction objectif pour toute violation des contraintes. Cette approche permet donc de traiter les problèmes par les algorithmes génétiques, qu'ils soient sous forme de programme linéaire ou non linéaire. Nous proposons ensuite la codification d'un algorithme d'optimisation en FORTRAN 90 pour synthétiser un REC qui optimise la récupération de chaleur, en utilisant le modèle que nous avons établi.

Il faut toutefois noter que notre algorithme ne traite que des problèmes de synthèse de REC sans division de courants.

1.4 Organisation du mémoire

Le premier chapitre est consacré à l'introduction du sujet qui présente les objectifs de cet ouvrage. Au Chapitre 2 nous présentons l'historique des travaux de recherche qui ont été menés dans le domaine de la synthèse des REC. Le Chapitre 3 est consacré à l'échangeur de chaleur, l'élément central dans tout système de récupération de chaleur. Nous faisons un bref rappel du principe de fonctionnement, de la notion de EMAT (Exchanger Minimal Approach Temperature) qui caractérise la plus petite différence de température observable dans le diagramme Température/enthalpie (T/ΔH) d'un échangeur de chaleur. C'est une caractéristique importante de l'échangeur de chaleur car sa valeur détermine l'économie réalisée dans la construction de l'échangeur et par suite, l'économie générale du système de récupération de chaleur. La méthode manuelle de synthèse d'un réseau d'échangeur de chaleur par la décomposition au pincement est expliquée au Chapitre 4. Cette procédure manuelle fait usage de graphiques, notamment

les ***courbes composites***, qui sont fondamentales dans l'établissement des besoins énergétiques du système et des opportunités de récupération de chaleur. Au Chapitre 5 nous proposons l'approche des algorithmes génétiques pour la synthèse de REC optimal pour la récupération de chaleur. Dans ce chapitre nous décrivons le principe de fonctionnement d'un AG et les opérateurs indispensables (mutation, croisement et sélection) pour son fonctionnement. La représentation de la structure de REC proposée par Lewin est utilisée, et nous avons présenté brièvement la méthode du simplexe pour la résolution d'un programme linéaire. Le Chapitre 6 présente les résultats des exemples d'application que nous avons utilisés pour montrer la validité de notre approche. Ces résultats sont comparés à ceux rencontrés dans la littérature. Nous terminons par la conclusion au Chapitre 7 où nous relevons les possibilités offertes par les AG mais aussi les insuffisances de la méthode.

La codification du programme en FORTRAN 90 est présentée à l'Annexe 1, pour l'approche de l'optimisation paramétrique par les algorithmes génétiques, et à l'Annexe 2, pour l'approche d'optimisation paramétrique par la méthode du simplexe.

CHAPITRE II

2 HISTORIQUE

L'histoire de l'évolution de la recherche dans le domaine de la synthèse des réseaux d'échangeurs de chaleur a été largement rapportée par Gundersen et Naess. (1988) [5] dans une revue faisant la compilation des différentes approches qui ont été initiées.
Il ressort de cette compilation que les différents travaux de synthèse des réseaux d'échangeurs ont principalement évolués suivant trois approches qui sont l'utilisation des concepts thermodynamiques, des méthodes mathématiques et l'utilisation des systèmes de l'intelligence artificielle (algorithmes évolutionnaires).

2.1 Les méthodes thermodynamiques

Vers la fin des années 1970, la découverte du ***pincement*** qui constitue une sorte d'entrave à la minimisation de la consommation d'énergie a incité d'avantages les universitaires et l'industrie à doubler d'effort dans le développement et l'application de méthodes systématiques dans le design des systèmes.

Umeda et al., (1978) [6] ont montré que le point de contact des courbes composites tracées dans un diagramme de disponibilité d'énergie représente en quelque sorte un goulot d'étranglement. qui empêche les réseaux d'échangeurs de chaleur de récupérer d'avantage de la chaleur, et par conséquent, d'économiser l'énergie. Ils ont nommé ce point le pincement. Mais la compréhension fondamentale du pincement et les effets de la décomposition qu'il implique a été donnée par Linnhoff qui a démontré plus tard qu'il existe un lien entre d'une part, la consommation excessive d'énergie et d'autre part le

transfert de chaleur à travers le pincement, le chauffage au-dessous du pincement et le refroidissement au-dessus du pincement. L'élimination du flux de chaleur à travers le pincement permet de concevoir un réseau d'échangeurs de chaleur assurant la minimisation de la consommation d'énergie. Pour cela, trois règles doivent être respectées :

- Ne pas transférer la chaleur du procédé à travers le pincement
- Ne pas utiliser de refroidisseur au-dessus du pincement
- Ne pas utiliser de réchauffeur au-dessous du pincement.

Ces principes conduisent directement à une stratégie pour la synthèse de réseaux d'échangeurs de chaleur. Cette stratégie a été nommé "***Pinch Design Method***" par Linnhoff et Hindmarsh (1983) [2] et comporte quatre étapes :

- Diviser le problème en deux parties au niveau du pincement.
- Concevoir séparément deux réseaux d'échangeurs de chaleur, l'un au-dessus du pincement, et l'autre au-dessous du pincement.
- Fusionner ensuite les deux structures de REC.
- Optimiser la structure résultant de la fusion des deux REC.

En effet, la structure résultant de la fusion contient généralement plus d'échangeurs de chaleur que nécessaires. Cette structure est alors optimisée dans le but de réduire le nombre d'unités mis en œuvre dans le réseau.

Un autre concept important, la ***technologie du pincement***, a été introduit par Linnhoff et Vredeveld, (1984) [7]. C'est un ensemble de méthodes basées sur la première et la deuxième loi de la thermodynamique, garantissant la minimisation de la consommation d'énergie lors du design des réseaux d'échangeurs de chaleur. L'idée principale est d'établir avant la phase du design, les valeurs cibles (***targets***) qu'on peut atteindre. Il s'agit notamment :

- Du besoin minimum d'énergie (unités de chauffage et de refroidissement externes)
- Du nombre minimum d'unités utilisé dans la structure du REC (échangeurs de chaleur, refroidisseurs et réchauffeurs)
- De la surface minimale totale requise des échangeurs de chaleur.

Il est aussi possible de combiner ces trois cibles de performance en une estimation du coût total annuel.

2.1.1 Détermination de la consommation minimale d'énergie

Un concept important utilisé dans la prédétermination des cibles de performance est le concept des courbes composites. Les courbes composites constituent un outil graphique très important dans la synthèse des réseaux d'échangeurs. Elles offrent un meilleur moyen d'illustrer le concept du pincement. Le concept des courbes composites a été utilisé pour la première fois par Huang et Elshout en 1976 [8]. Dans un diagramme température/enthalpie, les courbes composites sont établies en sommant les changements d'enthalpie des courants chauds et froids du procédé. Le diagramme des courbes composites donne des informations sur la quantité de chaleur qui peut être récupérée (Q_{REC}), du besoin minimal en chauffage (QHmin), du besoin minimal en refroidissement (QCmin) et la localisation du point de pincement pour une température d'approche (ΔTmin) donnée. Une stratégie générale pour la modification des procédés peut aussi être établie à partir des courbes composites. Cette stratégie a été nommée principe ***du Plus/Moins*** (Linnhoff et Vredeveld, 1984) [7]. Elle consiste à accroître ("**Plus**") la chaleur disponible au-dessus du pincement et/ou la demande de chaleur au-dessous du pincement ou à réduire ("**Moins**") la demande de chaleur au-dessus du pincement et/ou la chaleur disponible au-dessous du pincement. Les modifications qu'on pourrait opérer sont par exemple le changement de pression des colonnes de distillation et des évaporateurs, le changement du débit de certains courants, le changement si possible des températures des courants.

Le concept de "***Heat Cascade Diagram***" a été introduit par Linnhoff 1979 [9]. C'est une procédure numérique qui remplace le diagramme des courbes composites. Le concept a par la suite donné naissance à un autre diagramme nommé "***Grand Composite Curve***" (GCC) (Linnhoff et al., 1982) [10]. Ce diagramme a un grand nombre d'applications industrielles. Il peut :

- Identifier les types d'utilités (en termes de puissance et de niveau d'énergie) à utiliser pour couvrir les besoins externes en chauffage et en refroidissement du procédé.
- Identifier les potentialités de production de vapeur au-dessous du pincement, si le pincement est à une température suffisamment haute.
- Identifier l'opportunité d'utilisation de pompe à chaleur dans le procédé afin de réduire les consommations en utilité chaude et froide.
- Identifier s'il y a une opportunité d'intégration d'équipements spéciaux, tels les colonnes de distillations ou des évaporateurs.

2.1.2 Détermination du nombre minimum d'unités

La règle des (N-1) proposée par Hohmann, (1971) [11], a été largement utilisée pour déterminer le nombre minimum d'unités U_{min} à installer dans un réseau d'échangeurs de chaleur pour récupérer la quantité maximale de chaleur :

$$U_{min} = N - 1 \qquad [2.\ 1]$$

où N est le nombre total des courants chauds et froids y compris les utilités.

En utilisant le théorème d'Euler sur la théorie des graphes, Boland et Linnhoff, (1979) [12] ont reformulé cette règle suivant l'expression :

$$U_{min} = N - S + Lp \qquad [2.\ 2]$$

Ceci a pour but de tenir compte du nombre de sous-systèmes (S) qui peut exister dans le réseau ainsi que du nombre de boucles de chaleur (Lp). On retrouve l'expression de

Hohmann en supposant qu'il n'y a pas de boucle de charge thermique (les boucles peuvent être supprimées, $Lp = 0$) et qu'il n'y a pas de sous-systèmes (équilibre thermique du système des courants chauds et froids) ($S = 1$).

En vue d'obtenir une récupération maximale de la chaleur disponible, la décomposition au point de pincement doit être respectée. Deux réseaux d'échangeurs de chaleur doivent donc être conçus séparément, au-dessus et au-dessous du pincement. En appliquant l'équation [2.1] à chacun des deux systèmes, le minimum total résultant du nombre d'unités est donné par :

$$U_{\min} = (N-1)_{au-dessus} + (N-1)_{au-dessous} \qquad [2.3]$$

Des ajustements ont été apportés dans l'analyse du pincement par Ahmad et Smith, (1989) [13], pour tenir compte du fait que dans la pratique, peu d'échangeurs de chaleur utilisés dans l'industrie sont purement du type à contre-courants. Les ajustements concernent en même temps le nombre d'unités d'échangeurs de chaleur (qui est maintenant compté comme le nombre de calandres plutôt que le nombre d'échangeurs de chaleur) et la surface de transfert de chaleur. Ces extensions s'appliquent seulement aux échangeurs à calandre et tube pour lesquels, des facteurs de correction de la surface de transfert de chaleur sont utilisés. Ces facteurs dépendent des capacités calorifiques et des températures des courants.

2.1.3 Détermination de la surface minimale de transfert de chaleur

L'estimation du besoin de la surface totale de transfert de chaleur du REC avant la phase de design nécessite beaucoup de temps et comporte beaucoup d'incertitudes, notamment sur les coefficients de transfert de chaleur; des approximations sont donc faites dans le calcul de la surface minimale totale. En discrétisant les courbes composites, et en supposant un transfert vertical de la chaleur dans chaque intervalle d'enthalpie, Nishida

et al., (1981) [14], ont présenté l'équation suivante pour déterminer la surface minimale d'échange :

$$A_{\min} = \frac{1}{U}\sum_{i}\frac{\Delta H_i}{\Delta T_{mi}} \qquad [2.4]$$

Avec :

ΔH_i : Changement d'enthalpie dans l'intervalle i

ΔT_{mi} : Moyenne logarithmique de la différence de température dans l'intervalle i

U : Coefficient global de transfert de chaleur, considéré constant à travers le procédé et l'échangeur de chaleur utilisé est considéré de type à contre courant.

Townsend et Linnhoff, (1984) [15] ont introduit une autre approche pour le calcul de la surface minimale de transfert de chaleur en prenant en compte les coefficients de transfert de chaleur individuels des courants. Ainsi l'équation 2.5 apporte une contribution importante au calcul de la surface minimale et de l'estimation du coût minimal.

$$A_{\min} = \sum_{i}^{I}\frac{1}{\Delta T_{m,i}}\left(\sum_{1}^{N_h}\frac{q_h}{hf_h}+\sum_{1}^{N_c}\frac{q_c}{hf_c}\right) \qquad [2.5]$$

Avec :

q_h : Charge thermique du courant chaud dans l'intervalle d'enthalpie i

q_c : Charge thermique du courant froid dans l'intervalle d'enthalpie i

hf_h : Coefficients de transfert de film du courant chaud

hf_c : Coefficients de transfert de film du courant froid

I : Nombre total d'intervalles d'enthalpie

N_h : Nombre total de courants chauds

N_c : Nombre total de courants froids

Par cette approche, la minimisation de la surface totale d'échange de chaleur requiert un plus grand nombre d'intervalles d'enthalpie, ce qui nécessite un nombre important

d'échangeurs de chaleur; elle a reçu le nom de "***Spaghetti Design***" par Linnhoff et ses collaborateurs. S'il existe de grandes différences entre les coefficients de transfert de chaleur, les courants ayant un coefficient de transfert bas seront appariés (grande différence de températures entre courants) au dépend des appariements entre courants ayant de valeurs élevées du coefficient de transfert de chaleur (faible différence de températures entre les courants). Comme conséquence, il peut y avoir des transferts de chaleur non
verticaux entre les courants (***criss-cross***) (Tjoe et Linnhoff, 1986) [16].

2.2 Les méthodes mathématiques

La programmation mathématique, dans laquelle les différentes sous tâches du problème de synthèse sont automatisées, a été fréquemment utilisée. Elle formule le problème d'optimisation des REC sous forme de modèle mixte de programme non-linéaire et de variables entières (Mixed Integer Non-Linear Programming, MINLP), qui comporte des variables continues et des variables entières de décision. Les problèmes d'optimisation non linéaires requièrent d'habitude une bonne solution initiale et souvent la convergence à l'optimum global n'est pas assurée. Une difficulté de la programmation mathématique vient du fait que les formulations mathématiques dans les algorithmes sont souvent basées sur un grand nombre de petits intervalles (soit des intervalles de température, soit des intervalles d'enthalpie ou les deux). Ainsi, même avec des problèmes de taille modérée, le nombre d'intervalles peut être suffisamment grand, rendant le problème d'optimisation complexe.

Les concepts de "***spaghetti design***" (Townsend et Linnhoff, 1984) [15] et de "***stage***" (Yee et Grossmann, 1990) [17] ont été utilisés pour réduire la taille des problèmes. Cependant, le "spaghetti design" requiert un grand nombre d'échangeurs de chaleur pour permettre de réaliser un transfert vertical de la chaleur dans chaque intervalle d'enthalpie. Il existe un autre problème relatif aux incertitudes dues à la non-convexité du problème. Une recherche de l'optimum global a été proposée par Ciric et Floudas, (1991) [18]. Il

consiste à décomposer le problème d'optimisation non-convexe en des sous-problèmes convexes qui représentent les limites inférieures et supérieures dont la solution peut déterminer la configuration du réseau avec un investissement minimal.

Zhu et al., (1995) [19] ont couplé la méthode de "***blocks***" pour synthétiser des réseaux d'échangeurs de chaleur utilisant des courants ayant des coefficients de transfert de chaleur différents. Rapidement, des appariements potentiels sont identifiés en utilisant ces techniques et des structures initiales de REC sont générées. Ces structures initiales peuvent être par la suite optimisées en utilisant la programmation mathématique non linéaire.

2.3 Les méthodes basées sur l'intelligence artificielle

Pendant que la programmation mathématique appartient à ce qu'on appelle les méthodes déterministes d'optimisation, il y a eu aussi des applications de méthodes stochastiques ou non-deterministes pour déterminer l'optimum global des réseaux d'échangeurs de chaleur. Une de ces méthodes stochastiques, les ***algorithmes génétiques***, a été utilisée. C'est un algorithme d'optimisation qui échantillonne les zones de l'espace des solutions ayant une forte probabilité de mener vers la meilleure solution. Les algorithmes génétiques sont modelés sur le principe de l'évolution naturelle (Goldberg, 1989) [20]. Ils sont dits ***génétiques*** parce que la manipulation des solutions ressemble au mécanisme de la sélection naturelle : Dans l'évolution, le problème auquel fait face chaque espèce est celui de chercher à s'adapter à l'environnement compliqué et changeant. Les leçons apprises sont ainsi capturées dans les chromosomes de sa population.

Tandis que les méthodes classiques d'optimisation comptent sur la recherche de gradient local, les algorithmes génétiques manipulent une population de solutions potentielles. Quelques études ont été reportées, montrant l'application des algorithmes génétiques à la synthèse des réseaux d'échangeurs de chaleur (Androulakis et Venkatasubramanian

1991) [21]. Dans leur approche, les algorithmes génétiques sont utilisés pour l'optimisation des structures pendant qu'une autre méthode assure l'optimisation des charges des échangeurs de chaleur du réseau. Stair et Fraga (1995) [22] ont utilisé les algorithmes génétiques pour effectuer l'optimisation des charges des échangeurs de chaleur dans un réseau en supposant que les structures sont déterminées par d'autres procédures. Lewin et al. (1998a) [4] ont proposé une méthode de synthèse de réseaux d'échangeurs de chaleur pour la récupération d'énergie. Cette méthode ne prend pas en compte les problèmes avec division de courants et le problème est formulé comme un modèle mixte de programme linéaire et de variables entières (MILP). Cette approche a été étendue par Lewin (1998b) [23], pour traiter des problèmes formulés sous la forme d'un modèle MINLP (non linéarité due à l'introduction de l'estimation du coût optimal dans la synthèse du réseau ou la division des courants).

Les recherches dans le domaine de la récupération de chaleur se poursuivent, et le développement de l'outil informatique offre aujourd'hui les possibilités de manipuler des modèles de simulations complexes et difficiles à résoudre par les méthodes conventionnelles d'optimisation.

CHAPITRE III

3 LES ECHANGEURS DE CHALEUR

3.1 Introduction

Les échangeurs de chaleur sont utilisés dans une large variété d'applications comme les centrales électriques, les industries de procédés chimiques, le raffinage, le conditionnement d'air, la réfrigération, la récupération de chaleur, etc. Ils sont caractérisés essentiellement par le type d'application, le processus de transfert de chaleur, la géométrie de construction, le mécanisme de transfert de chaleur et le mode d'écoulement des fluides (Sadik et Hongtan, 2002) [24]. L'une des applications des échangeurs de chaleur qui retient notre attention est la récupération de chaleur. Nous nous intéresserons dans ce chapitre particulièrement de deux caractéristiques à savoir, la géométrie de construction et le mode d'écoulement des fluides.

3.2 Géométrie de construction des échangeurs de chaleur :

Les types de construction des échangeurs de chaleur les plus répandues sont:

- Tubulaires : Ils sont constitués par un faisceau de tubes métalliques reliés à leurs extrémités à des chambres de collection communiquant avec l'entrée ou la sortie d'un des fluides, alors que l'autre circule autour des tubes, généralement munis de chicanes servant à augmenter la turbulence.

- À plaques : Des plaques métalliques à profil ondulé sont empilées et séparées les unes des autres par des joints étanches, de telle sorte qu'un espace sur deux reçoit un des fluides, tandis que l'espace intermédiaire reçoit l'autre fluide.

3.3 *Les principaux modes d'écoulement*

- Échangeur à courants parallèles (co-courant) : Dans ce type d'échangeur, les deux courants circulent dans la même direction le long de l'échangeur (Figure.3.1.a).
- Échangeur à contre-courants : C'est un échangeur de chaleur dans lequel les deux courants circulent le long de l'échangeur dans des directions opposées (Figure.3.1.b).
- Échangeur à courants croisés : Dans un échangeur à courants croisés, un des courants s'écoule dans une direction perpendiculaire à l'autre (Figure.3.1.c).

Il existe d'autres types d'échangeurs de chaleur, notamment les échangeurs de chaleur multi-passes qui sont fréquemment utilisés dans l'industrie et sont constitués de tubes et calandres. Les formes spécifiques diffèrent selon le nombre de passes de calandre et tubes, et la forme la plus simple qui comprend une seule passe de tube et calandre est montrée à la Figure 3.1.d.

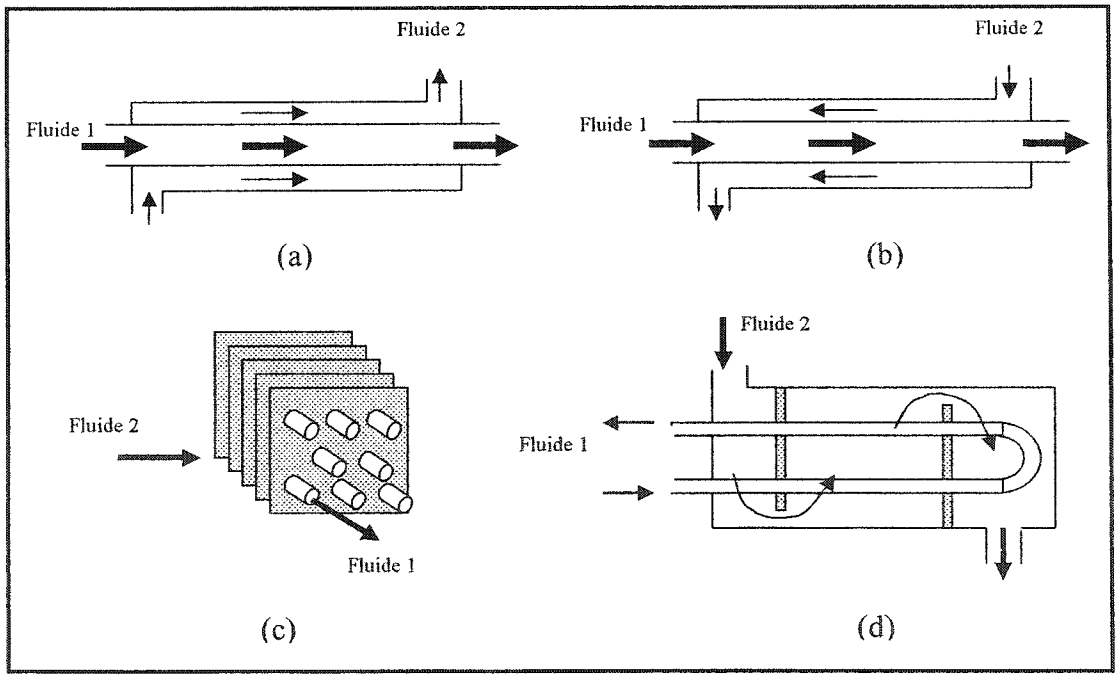


Figure 3.1: Types d'échangeurs de chaleur en fonction de l'écoulement des fluides

La principale différence entre les configurations d'échangeurs (figures 3.1) réside dans la distribution de température le long de l'échangeur de chaleur et la quantité de chaleur

échangée pour une température déterminée et pour une surface d'échange spécifiée. Un échangeur de chaleur à contre-courants requiert une surface d'échange minimale, un échangeur de chaleur à courants parallèles requiert une grande surface et un échangeur de chaleur à courants croisés requiert une surface intermédiaire entre les deux (Sadik et Hongtan, 2002) [24].

3.4 Équations fondamentales

Considérons un système ouvert, en régime permanent de la Figure 3.2.

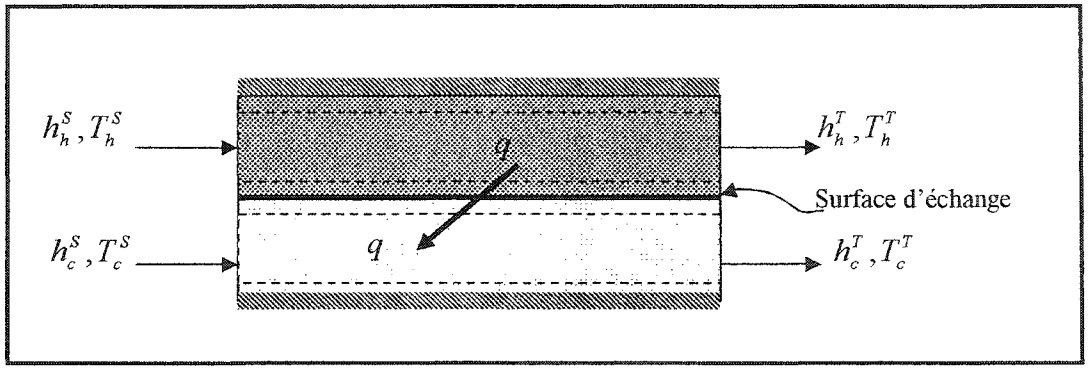


Figure 3.2 : Mécanisme de transfert de chaleur

En négligeant les énergies cinétique et potentielle ainsi que l'énergie échangée avec l'environnement, l'échange d'énergie entre les courants chaud et froid peut s'écrire, en vertu de la première loi de la thermodynamique:

$$q = m_h\left(h_h^S - h_h^T\right) \qquad [3.1]$$

On peut écrire également :

$$q = m_c\left(h_c^T - h_c^S\right) \qquad [3.2]$$

où h est l'enthalpie du fluide, m est le débit massique du fluide. Les indices h et c désignent respectivement le courant chaud et froid tandis que les exposants S et T désignent respectivement l'état initial et final du courant.

Si dans le processus d'échange de chaleur les fluides ne subissent pas de changement de phase, et en considérant que les chaleurs spécifiques sont constantes, alors les équations [3.1] et [3.2] peuvent se mettre sous la forme :

$$q = m_h cp_h \left(T_h^S - T_h^T\right) \qquad [3.3]$$

$$q = m_c cp_c \left(T_c^T - T_c^S\right) \qquad [3.4]$$

où les températures apparaissant dans les équations sont les valeurs moyennes des températures à la position indiquée. m_h et m_c désignent respectivement les débits massiques des fluides chaud et froid. Il est à noter que les équations [3.1], [3.2], [3.3] et [3.4] ne dépendent pas du type d'échangeur ou du type d'écoulement.

La différence de température entre un courant chaud et un courant froid dans un échangeur de chaleur ($\Delta T = T_h - T_c$), varie avec la position (Figure 3.3.b). Ainsi, dans l'analyse du transfert de chaleur, il convient d'établir une valeur moyenne appropriée de la différence de température entre les courants chaud et froid de sorte que l'énergie totale échangée entre les deux courants puisse être déterminée par l'équation suivante :

$$q = UA\Delta T_m \qquad [3.5]$$

où A est la surface totale d'échange de chaleur, U est le coefficient global de transfert de chaleur basé sur la surface A, ΔT_m est une moyenne appropriée de la différence de température et dépend de $T_h^S, T_h^T, T_c^S, T_c^T$.

En appliquant le bilan d'énergie à chaque élément différentiel de section dA et de longueur dl (Figure 3.3.a) nous avons :

$$dq = -m_h cp_h dT_h = -CP_h dT_h \qquad [3.\ 6.a]$$

et $dq = m_c cp_c dT_c = CP_c dT_c$ [3.6.b]

où $CP_h = m_h cp_h$ et $CP_c = mcp_c$ sont respectivement les capacités calorifiques des courants chaud et froid.

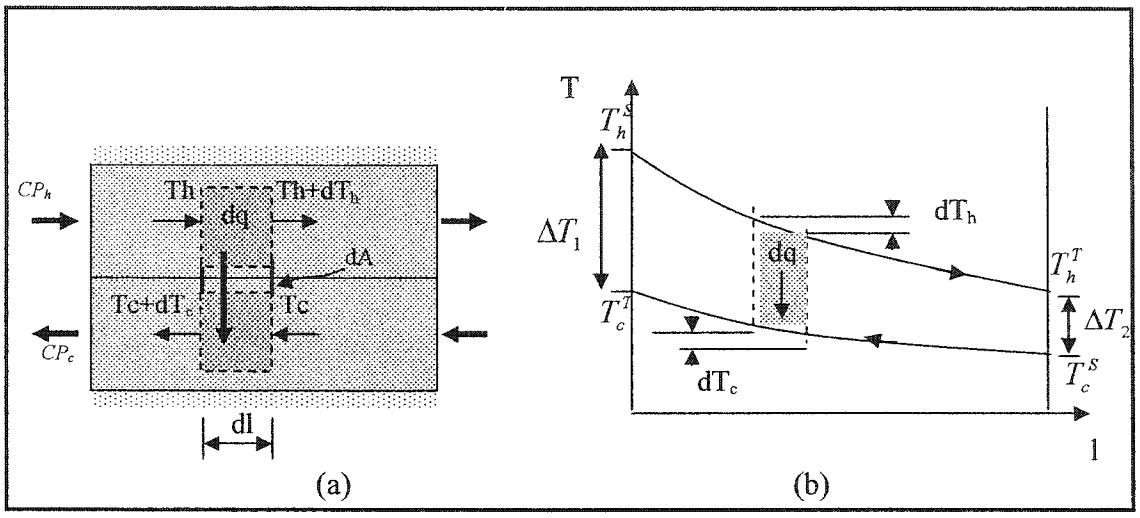


Figure 3.3 : Distribution de température dans un échangeur de chaleur

Le transfert de chaleur à travers la surface dA peut être aussi exprimée par :

$$dq = U(T_h - T_c)dA \qquad [3.\ 7]$$

À partir des équations [3.6.a] et [3.6.b] nous obtenons pour un échangeur de chaleur à contre-courant (où $dq = -CP_h dT_h = -CP_c dT_c$) l'expression suivante de la variation de la différence de température :

$$d(T_h - T_c) = dT_h - dT_c = dq\left(\frac{1}{CP_c} - \frac{1}{CP_h}\right) \qquad [3.\ 8]$$

En substituant la valeur de dq de l'équation [3.7] dans l'équation [3.8] nous obtenons :

$$\frac{d(T_h - T_c)}{(T_h - T_c)} = U\left(\frac{1}{CP_c} - \frac{1}{CP_h}\right)dA \qquad [3.9]$$

En considérant U , CP_c et CP_h constants, et en intégrant cette équation sur toute la longueur de l'échangeur de chaleur nous avons :

$$\ln\left(\frac{T_h^T - T_c^S}{T_h^S - T_c^T}\right) = UA\left(\frac{1}{CP_c} - \frac{1}{CP_h}\right) \qquad [3.10.a]$$

ou encore

$$T_h^T - T_c^S = \left(T_h^S - T_c^T\right)\exp\left[UA\left(\frac{1}{CP_c} - \frac{1}{CP_h}\right)\right] \qquad [3.10.b]$$

On peut également démontrer que pour un échangeur à courants parallèle l'équation [3.10.b] devient :

$$T_h^T - T_c^T = \left(T_h^S - T_c^S\right)\exp\left[-UA\left(\frac{1}{CP_c} + \frac{1}{CP_h}\right)\right] \qquad [3.11]$$

CP_c et CP_h peuvent être obtenus à partir des équations [3.3] et [3.4]. En substituant CP_c et CP_h dans l'équation [3.10.a] et en solutionnant pour q nous obtenons :

$$q = UA\frac{\left(T_h^S - T_c^T\right) - \left(T_h^T - T_c^S\right)}{\ln\left(\frac{T_h^S - T_c^T}{T_h^T - T_c^S}\right)} \qquad [3.12.a]$$

$$q = UA\frac{\Delta T_1 - \Delta T_2}{\ln\left(\Delta T_1 / \Delta T_2\right)} = UA\Delta T_m \qquad [3.12.b]$$

où $\Delta T_m = \dfrac{\Delta T_1 - \Delta T_2}{\ln\left(\Delta T_1 / \Delta T_2\right)}$ [3.13]

ΔT_m est donc une moyenne logarithmique de la différence de température entre le courant chaud et le courant froid (noté "Log-Mean Temperature Difference" ou LMTD).

La plus grande différence de température d'une extrémité de l'échangeur de chaleur est désignée par ΔT_1 et la plus petite différence de l'autre côté est désignée par ΔT_2.

3.5 *Profils de température selon le mode d'écoulement*

Dans un échangeur de chaleur à contre courant, la différence de température $(T_h - T_c)$ croît dans le sens de l'écoulement du courant chaud si $CP_c < CP_h$ (Figure 3.4.a). Si la longueur de l'échangeur de chaleur est infinie (ou $A = \infty$), la température de sortie du courant froid devient égale à la température d'entrée du courant chaud : La surface d'échange (A) de l'échangeur de chaleur dépendra donc de la valeur minimale que peut prendre ΔT_1 ($\Delta T_1 = T_h^S - T_c^T$). Si $CP_c > CP_h$, alors la différence de température $(T_h - T_c)$ décroît dans le sens de l'écoulement du courant chaud (Figure 3.4.b). Si la longueur de l'échangeur de chaleur est infinie (ou $A = \infty$), la température de sortie du courant chaud devient égale à la température d'entrée du courant froid : La surface d'échange (A) de l'échangeur de chaleur dépendra de la valeur minimale que peut prendre ΔT_2 ($\Delta T_2 = T_h^T - T_c^S$).

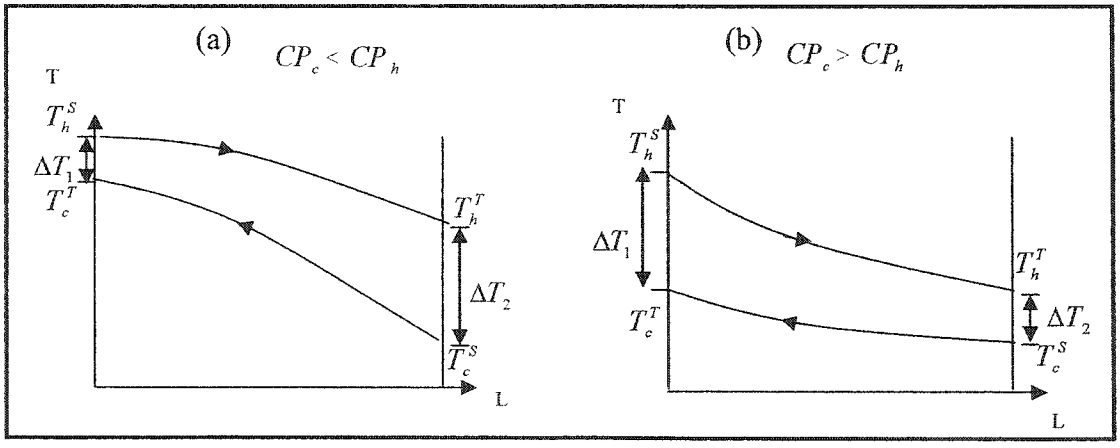


Figure 3.4 : Profils de température d'un échangeur à contre-courants

Un échangeur de chaleur à contre-courant permet donc un bon échange de chaleur aussi bien du côté le plus chaud que du côté le plus froid des courants. Pour cette raison la variation de la différence de température $\Delta T = T_h - T_c$ entre les deux courants est approximativement constante le long de l'échangeur de chaleur.

À l'entrée d'un échangeur de chaleur à courants parallèles, les deux courants ont une différence de température ΔT_1 élevée. À la sortie, la différence de température est petite à cause de la chaleur échangée le long de l'échangeur (Figure 3.5). Il est important de noter que pour un tel échangeur, la température de sortie du courant froid ne peut en aucun cas excéder celle du courant chaud.

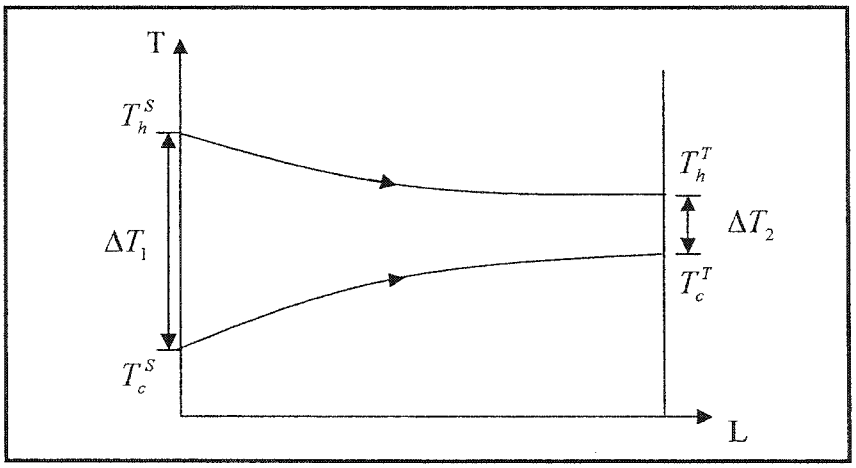


Figure 3.5 : Profil de température d'un échangeur à courants parallèles

Il est à noter également que pour les mêmes températures d'entrée et de sortie, le LMTD d'un échangeur à contre-courant est plus grand que celui d'un échangeur à courants parallèles. Ainsi de l'équation [3.12.b], nous déduisons que, pour le même coefficient global d'échange (U), la surface d'échange (A) requise pour transférer une certaine quantité de chaleur (q) sera plus petite pour un échangeur à contre-courant qu'un échangeur à courants parallèles.

Le LMTD développé précédemment n'est pas applicable directement pour l'analyse de transfert de chaleur dans les échangeurs à courants croisés ou à multi-passes (Tube et calandre). Il doit être utilisé en le multipliant par un coefficient de correction F (Incropera et DeWitt, 2001) [25]. Dans ce cas, ΔT_m est calculé en considérant un échangeur à contre-courant.

$$q = UAF\Delta T_m \qquad [3.\ 14]$$

F est un coefficient adimensionnel qui dépend du rapport (R) des capacités calorifiques, de l'effectivité de température (P), et de la configuration de l'échangeur de chaleur.

$$R = \frac{m_c cp_c}{m_h cp_h} \qquad [3.\ 15]$$

$$P = \frac{T_c^T - T_c^S}{T_h^s - T_c^s} \qquad [3.\ 16]$$

3.6 Conditions spéciales de fonctionnement des échangeurs de chaleur

La Figure 3.6.a présente la distribution de température dans un échangeur de chaleur où la capacité calorifique du courant chaud (CP_h) est largement supérieure à celle du courant froid (CP_c). La température du courant chaud reste pratiquement constante à travers l'échangeur de chaleur pendant que la température du courant froid croît. Cette situation correspond au cas où le courant chaud est de la vapeur en condensation ($CP_h \to \infty$). À la Figure 3.6.b, nous avons le cas d'une évaporation du courant froid qui s'effectue à une température presque constante ($CP_c \to \infty$) alors que la température du courant chaud décroît. La Figure 3.6.c présente un échangeur à contre-courant pour lequel les capacités calorifiques des courants froid et chaud sont égales ($CP_c = CP_h$). La différence de température ΔT reste constante le long de l'échangeur et $\Delta T_1 = \Delta T_2 = \Delta T_m$.

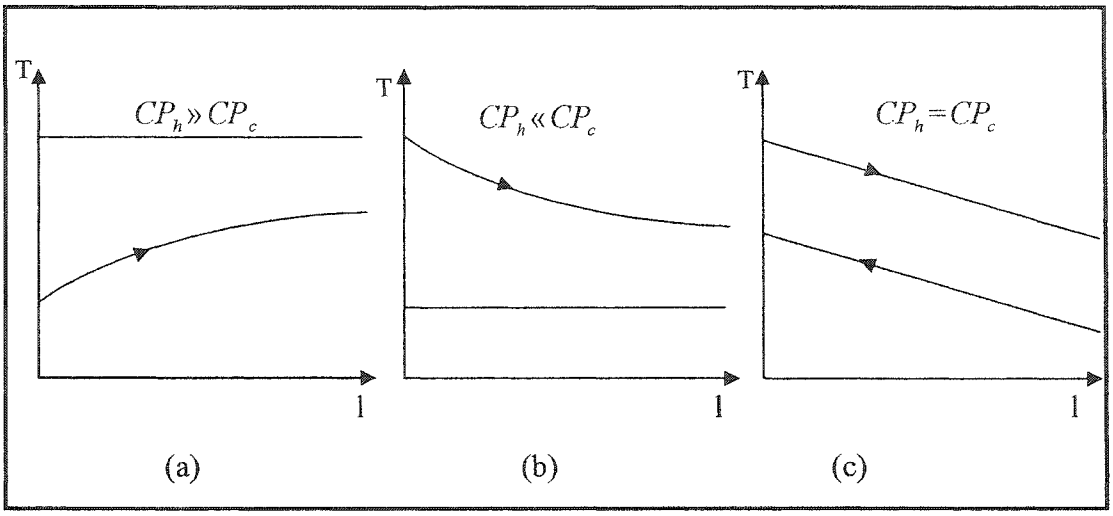


Figure 3.6 : Conditions spéciales de fonctionnement d'un échangeur de chaleur

3.7 *Concept de température minimale d'approche d'un échangeur de chaleur*

De façon pratique, pour qu'il y ait transfert de chaleur dans un échangeur de chaleur, la température du courant froid doit être inférieure à celle du courant chaud en tout point de l'échangeur par une quantité raisonnable. La plus petite différence de température acceptable est appelée ***température minimale d'approche de l'échangeur*** (Exchanger Minimal Approach Temperature, **EMAT**). Nous présentons à la Figure 3.7.a des courbes d'échange de chaleur faisables. La courbe de température du courant chaud est toujours au-dessus de celle du courant froid. Par contre, les courbes présentées à la Figure 3.7.b sont infaisables à cause du croisement des températures des courants chauds et froids. Le choix de EMAT a des implications aussi bien dans le coût de construction des échangeurs de chaleur que le coût d'approvisionnement en énergie. Nous pouvons le voir à travers le calcul de la surface de transfert de chaleur. La surface d'échange de chaleur peut être déduite de l'équation [3.12.b] :

$$A = \frac{Q}{U.\Delta T_m} \qquad [3.17]$$

où $\Delta T_m = \dfrac{\Delta T_1 - \Delta T_2}{\ln(\Delta T_1 / \Delta T_2)}$ (voir l'équation 3.13).

Pour une quantité de chaleur Q à transférer, si l'on choisit EMAT petite, la surface d'échange (A) requise sera grande. Par contre en fixant EMAT assez grande, la puissance échangée diminue, tandis que le besoin de puissance à partir d'une source extérieure augmente.

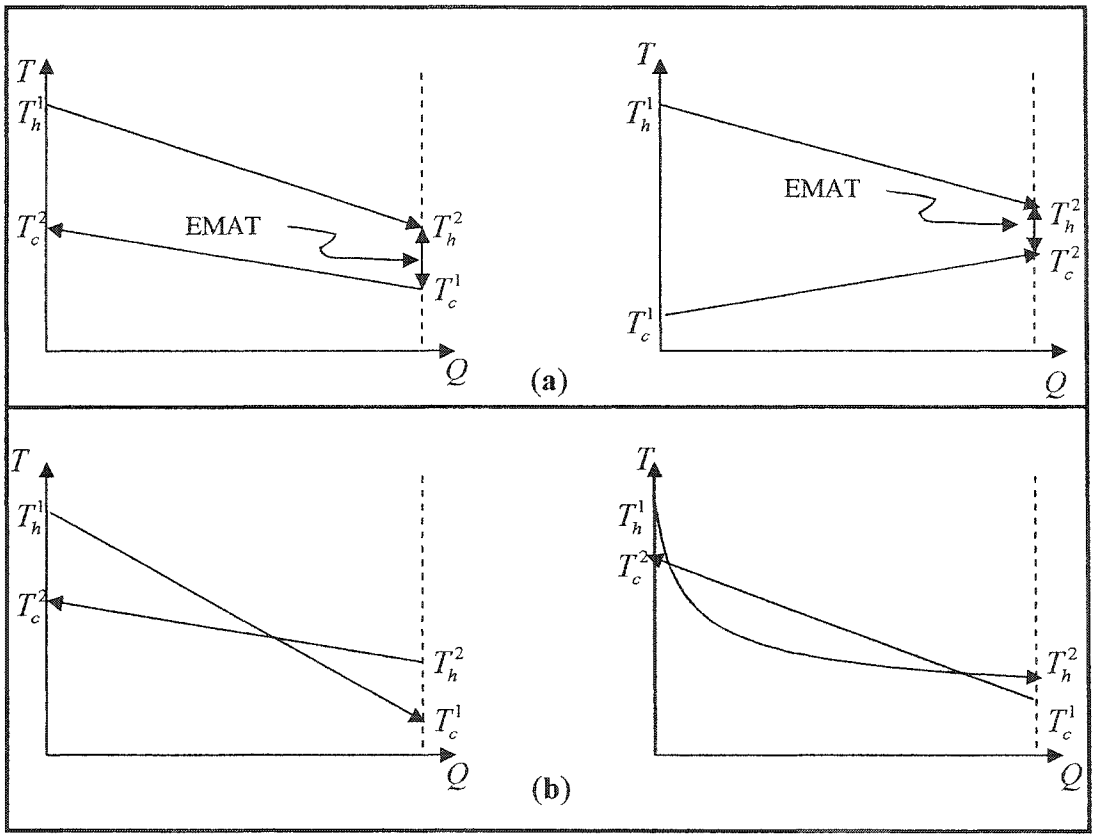


Figure 3.7 : Courbes d'échange de chaleur faisables et infaisables

Dans la conception des réseaux d'échangeurs de chaleur, le concept de la température minimale d'approche est utilisé pour désigner la plus petite différence de température entre les courbes composites des courants chauds et froids. Sa valeur fixe la quantité de chaleur récupérable et par conséquent, les économies d'énergie réalisables.

CHAPITRE IV

4 SYNTHÈSE D'UN RÉSEAU D'ÉCHANGEURS DE CHALEUR PAR LA MÉTHODE DU PINCEMENT

4.1 Introduction

Le développement de la méthode du pincement par Linnhoff et Hindmarsh, (1983) [2] a rendu le design des réseaux d'échangeurs de chaleur systématique et méthodique. Cette méthode est capable de prévoir les performances du réseau avant même sa conception. Ainsi l'on peut à travers l'analyse du pincement déterminer la puissance maximale à récupérer (ou l'énergie minimale requise), le nombre d'unités nécessaire pour la récupération optimale de chaleur, la surface minimale des échangeurs de chaleur ou le coût total annuel minimal du processus de récupération de chaleur. L'analyse du pincement pour le design d'un réseau d'échangeurs de chaleur se fait en général à travers les quatre étapes suivantes :

- La collecte des données de l'installation.
- La détermination de la puissance maximale récupérable (ou le besoin minimum en source extérieure d'énergie).
- La conception du REC en suivant les règles de la méthode du pincement.
- L'optimisation finale du réseau d'échangeurs de chaleur.

La représentation de la structure d'un réseau d'échangeurs de chaleur que nous allons utilisée a été proposée par Flowers et Linnhoff, (1980) [26], (voir la Figure 4.1). Les courants chauds sont regroupés ensemble et placés à la partie supérieure du diagramme, tandis que les courants froids sont placés à la partie inférieure. Le sens des flèches indique toujours le sens des écoulements des courants à partir de la température initiale

vers la température finale désirée. La mise en place d'un échangeur de chaleur entre un courant chaud et un courant froid est représenté par une ligne verticale joignant deux cercles sur les deux courants qui doivent échanger de la chaleur. Les réchauffeurs sont représentés sur le diagramme par des cercles placés sur les lignes des courants froids et les refroidisseurs par des cercles placés sur les lignes des courants chauds.

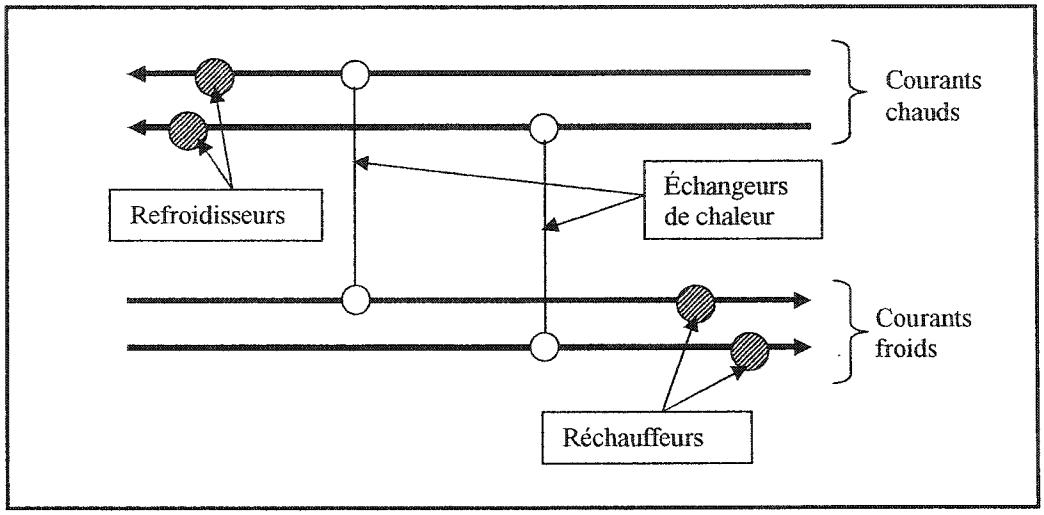


Figure 4.1 : Diagramme de réseau d'échangeurs de chaleur

4.2 La collecte des données de l'installation :

Avant toute conception d'un réseau d'échangeurs de chaleur, il importe de définir clairement le problème. La conception a pour but essentiellement de déterminer une topologie optimale du réseau d'échangeurs de chaleur qui permettra de satisfaire aux besoins des courants. La partie centrale de la collecte des données est l'identification des besoins en chauffage et en refroidissement.

Dans cet ouvrage, les définitions suivantes sont utilisées :

- Courants chauds : Ce sont les courants qui doivent être refroidis.
- Courants froids : Ce sont les courants qui doivent être réchauffés.
- Utilités : Elles sont utilisées pour réchauffer (vapeur, eau chaude, gaz d'échappement,

etc.) ou refroidir (eau de refroidissement, air, réfrigérant, etc.) les courants du procédé, quand l'échange de chaleur entre courants du procédé n'est pas pratique ou économique.

Les caractéristiques nécessaires des courants qui vont servir à la conception du réseau d'échangeurs de chaleur sont :

- La température initiale (T^S) : C'est la température à laquelle est disponible le courant chaud ou froid.
- La température finale (T^T) : C'est la température à laquelle doit être amené le courant chaud ou froid.
- La capacité calorifique (CP): C'est le produit du débit massique (m) et la chaleur spécifique (cp). Les chaleurs spécifiques sont supposées constantes par rapport aux variations de température dans la gamme d'opération. Ainsi, la variation d'enthalpie (ΔH) associée avec un fluide passant à travers l'échangeur de chaleur est exprimée par :

$$\Delta H = CP\left(T^T - T^S\right) \qquad [4.1]$$

Une valeur négative de ΔH indique un excédent de chaleur à extraire et une valeur positive indique un déficit en chaleur qu'il faut combler. Dans cet ouvrage, nous utiliserons toujours la valeur positive pour indiquer la quantité de chaleur fournie ou demandée, associée à chaque courant. Pour présenter la méthode du pincement, nous nous servirons de l'exemple du procédé industriel (Figure 4.2) utilisé par Smith, (1995) [27]. L'installation comporte deux courants chauds et deux courants froids. Le premier courant froid que nous appelons Alimentation 1 a une température initiale de 20°C et doit alimenter le réacteur 1 sous une température de 180°C. Le second courant froid (alimentation 2) dont la température est 140°C alimente le réacteur 2 sous une température 230°C. Le produit 1 obtenu à la sortie du réacteur 1, a une température de 250°C et doit être refroidi à 40°C. À la sortie du réacteur 2, le produit 2 a une température de 200°C et doit être refroidi à 80°C.

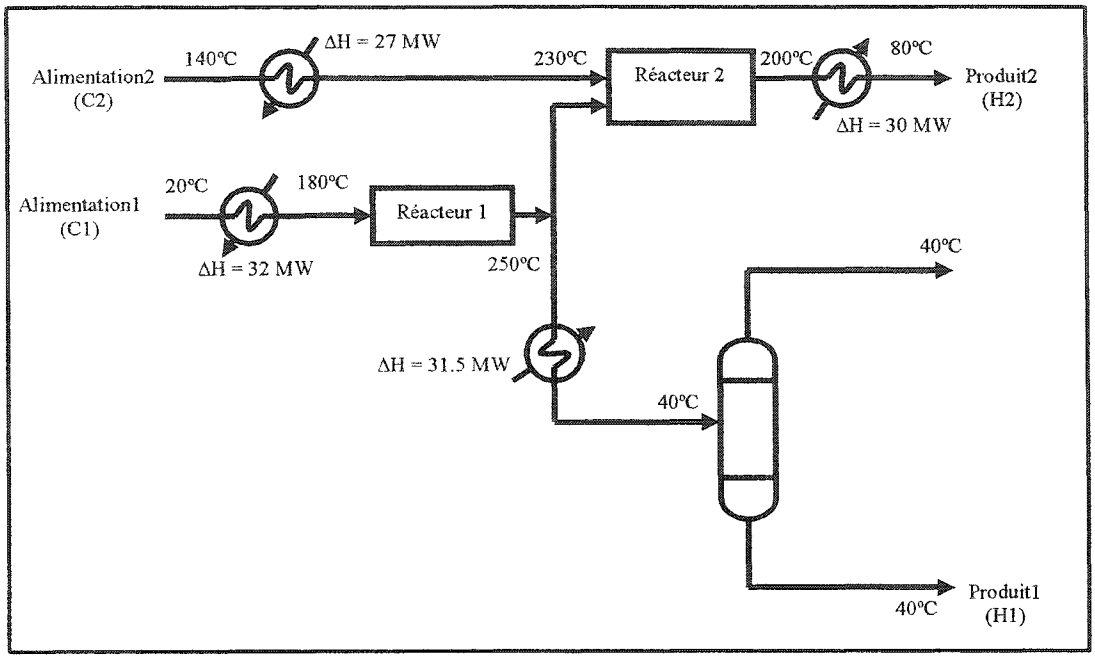


Figure 4.2 : Exemple de procédé industriel

L'analyse de l'installation fournit le bilan suivant :

Chaleur totale disponible = 31.5 + 30 = 61.5 MW

Chaleur total requise = 32 + 27 = 59 MW

Chaleur nette disponible = 2.5 MW

Les caractéristiques de cette installation sont reportées dans le Tableau 4.1.

Tableau 4.1 : Caractéristiques des courants

Courants	Température initiale T^S (°C)	Température finale T^T (°C)	Quantité de chaleur (MW)	Capacité calorifique CP (MW/°C)
H1	250	40	31.5	0.15
H2	200	80	30	0.25
C3	20	180	32	0.2
C4	140	230	27	0.3

4.3 Détermination de la puissance maximale récupérable

Nous avons dans cette installation, un excédent net de chaleur, ce qui montre que l'on peut satisfaire aux besoins de l'installation en procédant à une récupération de chaleur. Le problème qui se pose est de savoir quelle quantité de chaleur peut être réellement récupérée et quelle doit être la structure adéquate du réseau d'échangeurs de chaleur pour récupérer cette quantité de chaleur. Le tracé du diagramme des courbes composites que nous verrons par la suite montre qu'en général, toute la chaleur disponible ne peut être récupérée. Seule une partie de cette chaleur peut être transférée dont la quantité dépend de deux facteurs, à savoir les caractéristiques du problème (températures et capacités calorifiques) et la température minimale d'approche ΔT_{min}. Le premier facteur dépend du procédé tandis que le deuxième est fixé par le concepteur.

4.3.1 Choix de la température minimale d'approche

L'équipement de transfert de chaleur doit toujours satisfaire la seconde loi de la thermodynamique qui interdit tout croisement de température entre le courant chaud et le courant froid. Il doit toujours exister entre les deux courants, une différence de température (***driving force***), indispensable pour permettre un transfert de chaleur. La plus petite différence de température admissible sera notée ΔT_{min}. En pratique, la valeur de cette température découle d'une optimisation que nous n'abordons pas dans ce document. Nous supposons ici que cette valeur est connue et est fixée à ΔT_{min} =10°C.

4.3.2 Construction du diagramme des courbes composites

En général, tout courant avec une capacité calorifique constante est représenté dans un diagramme température/enthalpie par un segment linéaire allant de la température initiale du courant à la température finale. Quand il y a plusieurs courants chauds et froids la construction des courbes composites se fait par addition des changements d'enthalpie des courants dans les intervalles de températures respectifs. Nous présentons à la Figure 4.3, la procédure de construction de la courbe composite des courants chauds.

Le courant H1 apporte de la chaleur entre 250°C et 40°C, et a une capacité calorifique CP = 0.15 MW/°C. Le courant H2 apporte de la chaleur entre 200°C et 80°C avec une capacité calorifique CP = 0.25 MW/°C. Trois intervalles de température peuvent donc être identifiés à partir de la Figure 4.3 :

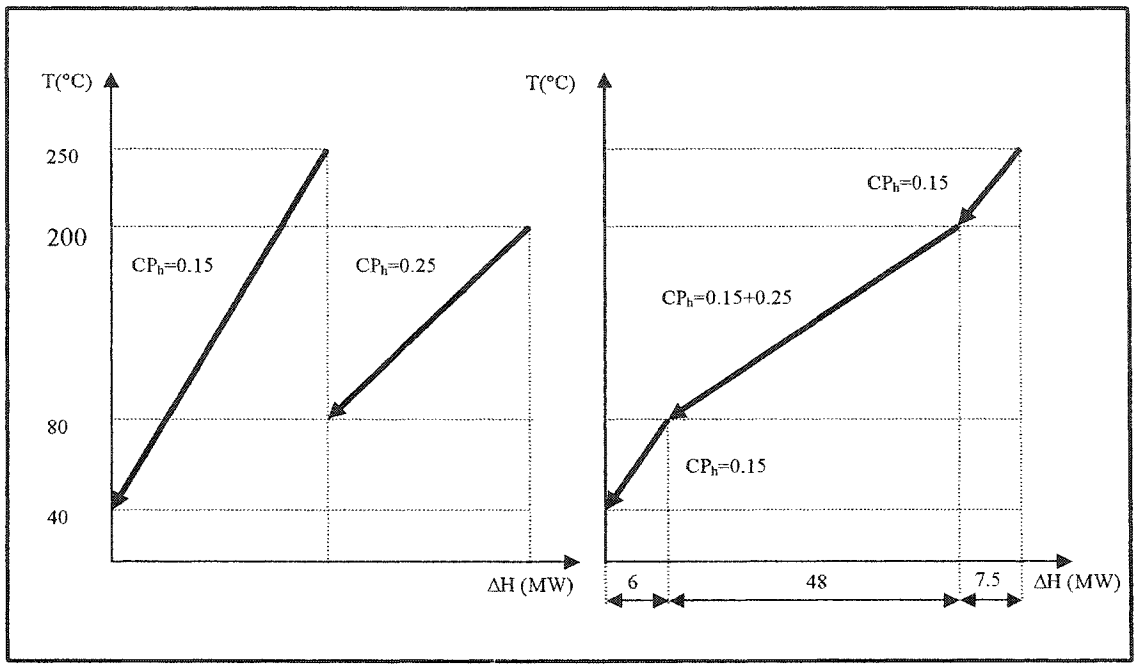


Figure 4.3 : Construction de la courbe composite des courants chauds

1- [250°C, 200°C] : seul le courant chaud H1 fournit de la chaleur :

$\Delta H = 0.15(250\text{-}200) = 7.5$ MW

2- [200°C, 80°C] : les courants chaud H1 et chaud H2 fournissent de la chaleur :

$\Delta H = (0.15+0.25)\ (200\text{-}80) = 48$ MW.

3- [80°C à 40 °C] : seul le courant chaud H1 fournit de la chaleur :

$\Delta H = 0.15(80\text{-}40) = 6$ MW

Nous avons sur la courbe composite des courants chauds trois intervalles de variations d'enthalpie, de valeurs 7.5; 48; et 6 MW dont la somme est 61.5 MW. La courbe

composite des courants chauds représente donc la somme de toutes les chaleurs disponibles dans le procédé.

La construction de la courbe composite des courants froids suit la même procédure que celle des courants chauds (Figure 4.4).

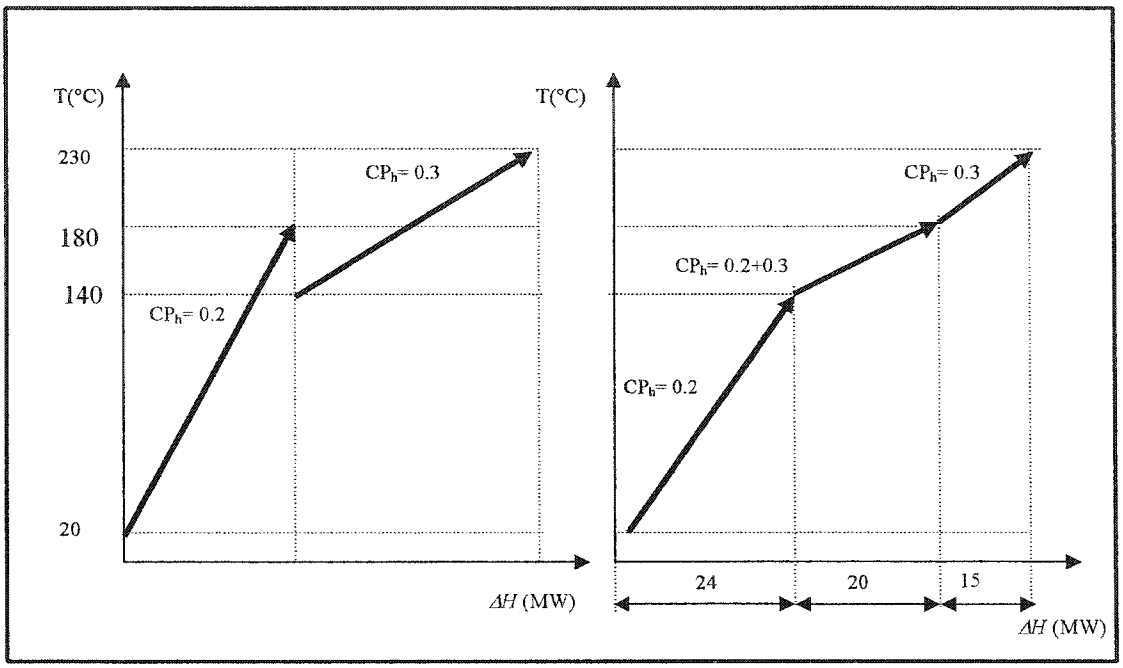


Figure 4.4 : Construction de la courbe composite des courants froids

Sur la Figure 4.4 nous avons trois intervalles de variations d'enthalpie de valeurs 15; 20; et 24 MW dont la somme est 59 MW. La courbe composite des courants froids représente la somme de toutes les chaleurs requises dans le procédé.

En regroupant les deux courbes composites dans le même diagramme nous obtenons le diagramme des courbes composites (Figure 4.5). La courbe à la partie supérieure est la courbe composite des courants chauds et celle qui est en dessous représente la courbe composite des courants froids. La position relative de ces deux courbes peut être variée

en les déplaçant horizontalement l'une par rapport à l'autre sans changer les caractéristiques de température ou d'enthalpie. Le déplacement dans le sens vertical n'est pas possible puisque cela modifierait les températures des courants.

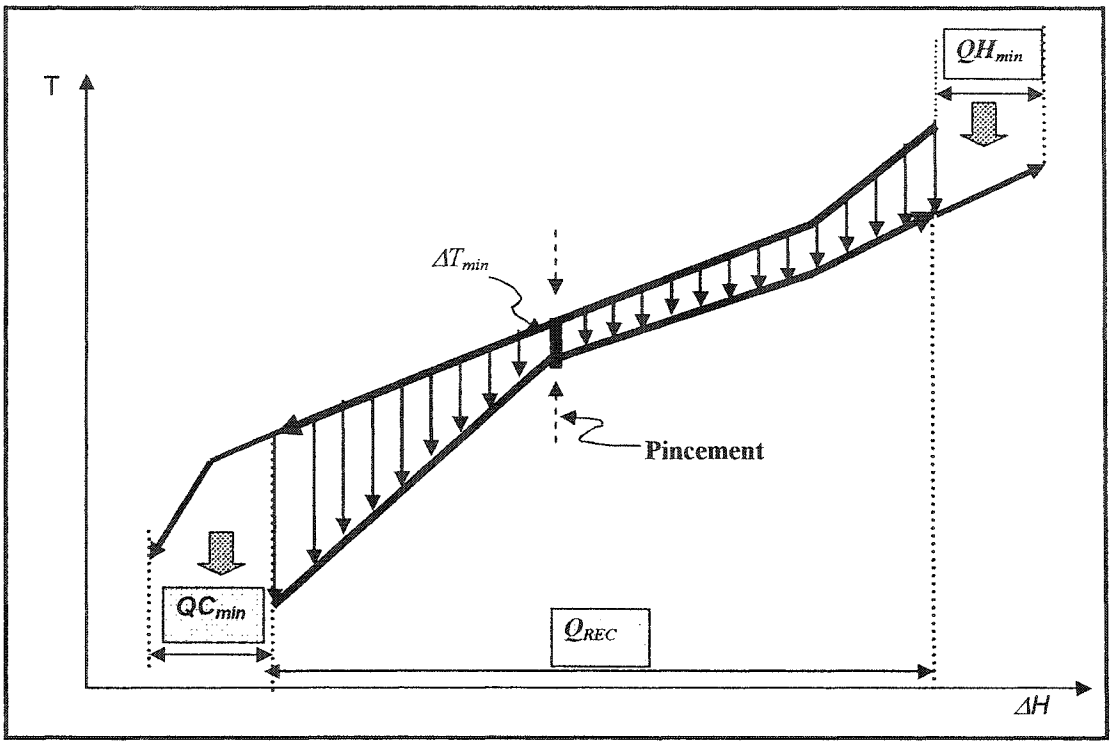


Figure 4.5 : Diagramme des courbes composites

En superposant les deux courbes de sorte à maintenir entre elles une différence de température minimale ΔT_{min} = 10°C, nous observons trois zones distinctes (Figures 4.5):

- **Zone centrale** (Q_{REC}): Dans toute cette zone, les deux courbes se chevauchent. La longueur de chevauchement détermine la quantité de chaleur maximale (Q_{REC}) qui peut être échangée entre les courants chauds et froids du procédé à travers des échangeurs de chaleur . Ainsi à partir de la figure 4.4 on note que la chaleur récupérée Q_{REC} = 51.5 MW.

- **Zone de dépassement à gauche** (QC_{min}): Dans cette zone, la courbe composite des courants chauds ne chevauche aucune courbe composite des courants froids. Sa longueur détermine la quantité de chaleur rejetée dans l'atmosphère. Cette partie ne peut donc être refroidie à l'aide d'un échangeur de chaleur. Elle nécessite une utilité froide de puissance QC_{min} pour refroidir les courants chauds : La valeur de cette puissance vaut : QC_{min} = 10 MW (Figure 4.5).
- **Zone de dépassement à droite** (QH_{min}): Dans cette zone se trouve un dépassement de la courbe composite des courants froids. Cette partie ne peut être réchauffée par l'utilisation d'un échangeur de chaleur. Sa longueur est une mesure directe du déficit en calories et requiert une utilité chaude de puissance QH_{min} pour réchauffer les courants froids. La puissance requise vaut : QH_{min} = 7.5 MW (Figure 4.5).

La position relative des deux courbes composites définit en général dans le diagramme un lieu où la différence de température entre les deux courbes est la plus petite. Ce point est appelé ***pincement*** (Linnhoff et al., 1982) [10], et sa position dépend de ΔT_{min}. Dans notre exemple, la température de pincement des courants chauds est 150°C et celle des courants froids est 140°C.

4.3.3 Influence de la température minimale sur la récupération de chaleur

La puissance maximale récupérée (Q_{REC}), le besoin minimum en chauffage (QH_{min}) et le besoin minimal en refroidissement (QC_{min}) dépendent de la valeur de la température minimale d'approche ΔT_{min} :

- Si l'on augmente la température minimale d'approche (de ΔT_{min1} à ΔT_{min2}) en déplaçant horizontalement les courbes composites l'une par rapport à l'autre (voir la Figure 4.6), le chevauchement entre les courbes composites des courants chauds et froids se rétrécit. Ceci entraîne une augmentation des besoins en utilités chaudes et froides (de QH_{min1} à QH_{min2} et de QC_{min1} à QC_{min2}), mais permet cependant de réduire la surface totale de transfert de chaleur requise.

- Si l'on diminue la température minimale d'approche, les besoins en utilités chaudes et froides seront réduits mais une plus grande surface de transfert de chaleur sera requise.

Par exemple : Pour ΔT_{min} = 10°C , nous avons Q_{REC} = 51.5 MW, QH_{min} = 7.5 MW, et QC_{min} = 10MW

Pour ΔT_{min} = 20°C , nous avons Q_{REC} = 47.5 MW, QH_{min} = 11.5 MW et QC_{min} = 14MW

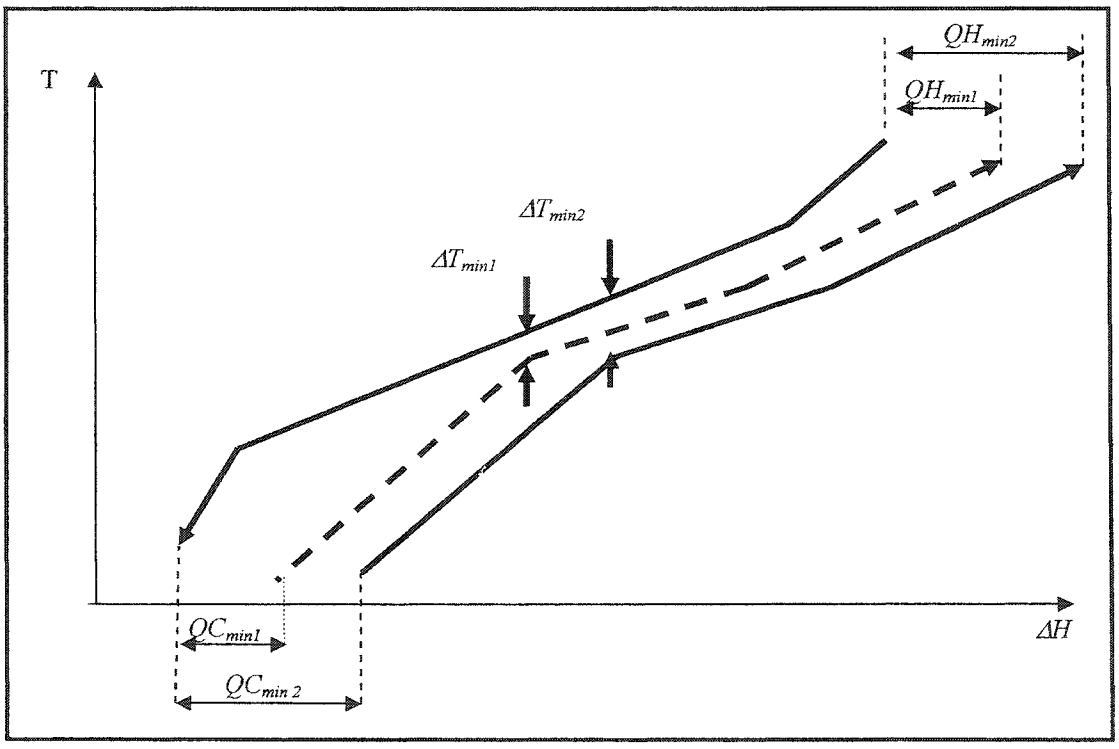


Figure 4.6 : Influence de la température minimale sur la récupération de chaleur

La Figure 4.7 illustre des variations typiques des coûts d'un système de récupération de chaleur en fonction de la température minimale d'approche (Smith, 1995) [27]. Si nous choisissons une température minimale d'approche ΔT_{min} = 0, cela nécessiterait théoriquement une surface de transfert de chaleur infinie (coût d'investissement infini) mais les coûts de l'énergie seraient réduits. Par contre, le choix de ΔT_{min} assez large conduit à

une réduction des coûts d'investissements mais les dépenses pour l'énergie sont élevées. Il est donc évident qu'il existe une température minimale d'approche optimale qui permet de minimiser le coût total des opérations de l'installation.

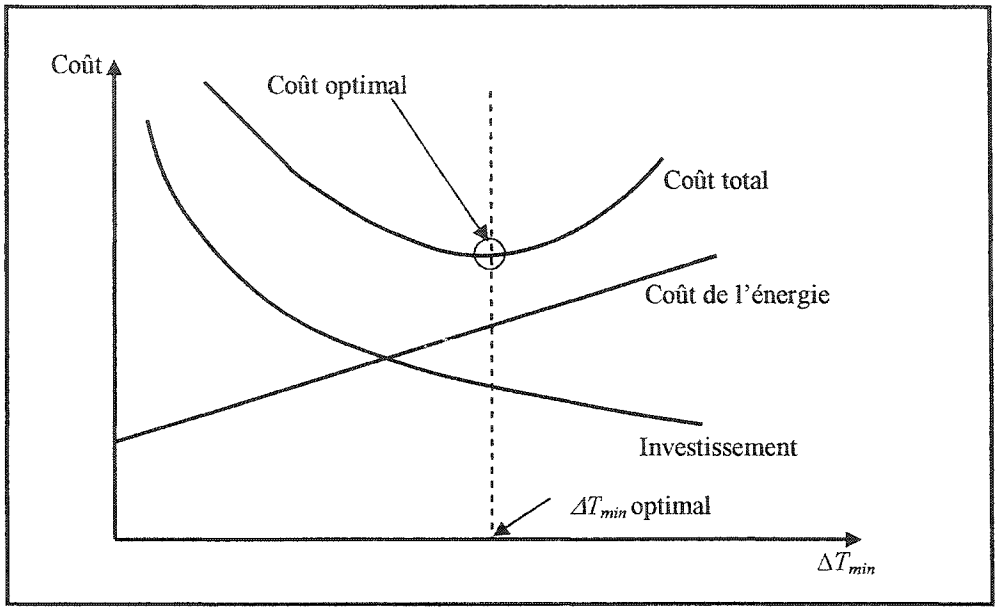


Figure 4.7 : Compromis entre l'investissement et la récupération de chaleur

Nous présentons dans le Tableau 4.2 quelques valeurs typiques de ΔT_{min} pour des échangeurs de types calandre et tube (site Internet de *The Chemical Engineers' Resource*) [28].

Tableau 4.2 : Valeurs expérimentales de $\Delta Tmin$

Secteur industriel	Valeurs de ΔT_{min}
Raffinerie	20 – 40 °C
Pétrochimie	10 – 20 °C
Chimie	10 – 20 °C
Procédés à basse température	3 – 5 °C

4.4 Procédure de conception d'un REC

Après avoir déterminé la quantité de chaleur disponible pour la récupération, la méthode du pincement offre des stratégies pour concevoir un recevoir d'échangeur de chaleur qui récupère cette quantité disponible, en utilisant le moins d'unités possibles.

4.4.1 Décomposition du système au point de pincement

Le ***point de pincement*** est le lieu dans le diagramme température/enthalpie où les deux courbes composites (chaudes et froides) ont la plus petite différence de température. Il divise les courbes composites en deux régions, (Voir la Figure 4.8):

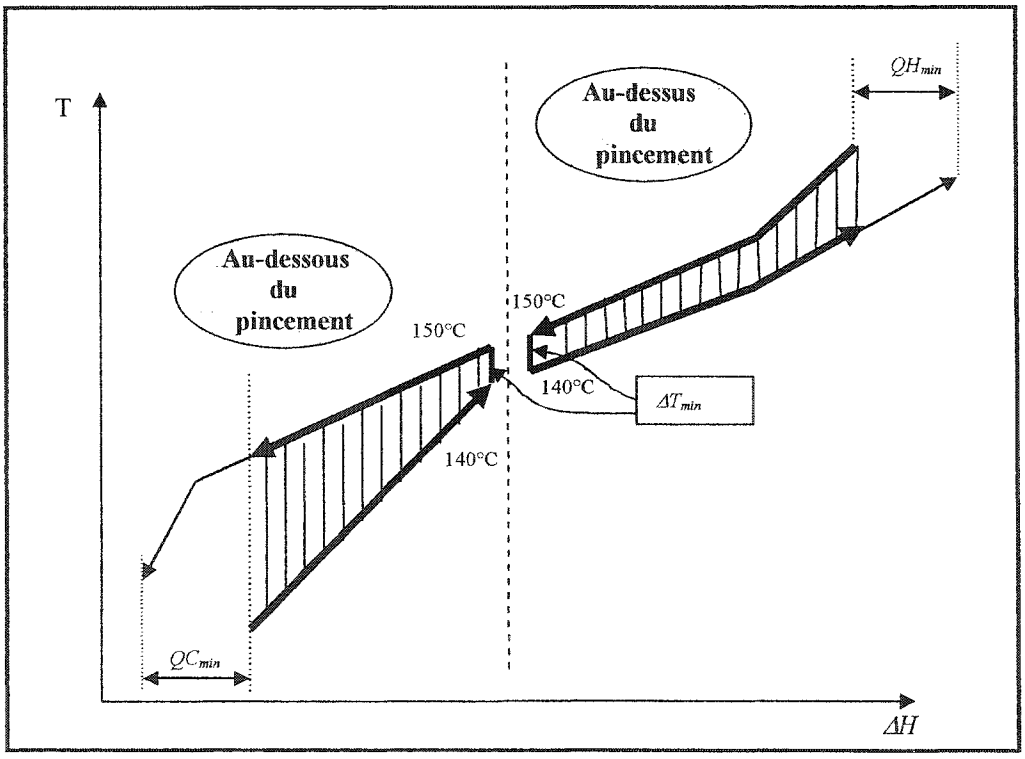


Figure 4.8 : Décomposition au pincement

- La région ***au-dessus du pincement*** : Il y a un échange de chaleur entre les courants chauds et froids dans la partie où les courbes composites se chevauchent. La partie

excédentaire de la courbe composite des courants froids ne peut recevoir de la chaleur à travers un échangeur de chaleur et doit être compensée par une source extérieure de chauffage de puissance QH_{min}.

- La région ***au-dessous du pincement*** : Il y a un échange de chaleur entre les courants chauds et froids dans la partie où les courbes composites se chevauchent. La partie excédentaire de la courbe composite des courants chauds ne peut être refroidie à travers un échangeur de chaleur et doit être compensée par une source extérieure de refroidissement de puissance QC_{min}.

Pour satisfaire les besoins en énergie établis par les courbes composites, trois règles fondamentales doivent être respectées (Linnhoff et Hindmarsh, 1983) [2] :

a) Ne jamais transférer de la chaleur à travers le pincement (Voir la Figure 4.9.a): Toute quantité de chaleur (Q_{extra}) transférée à partir du système au-dessus du pincement vers le système au-dessous du pincement entraîne systématiquement une double pénalité : Une quantité de chaleur Q_{extra} doit être fournie au système au-dessus du pincement. La même quantité de chaleur doit être soustraite au système au-dessous du pincement.

b) Ne jamais utiliser un refroidisseur au-dessus du pincement (voir la Figure 4.9.b): La quantité de chaleur (Q_{extra}) soustraite au-dessus du pincement doit être compensée par un réchauffeur supplémentaire de même puissance afin de rétablir l'équilibre du système.

c) Ne jamais utiliser un réchauffeur au-dessous du pincement (Voir la Figure 4.9.c): La quantité de chaleur (Q_{extra}) fournie au-dessous du pincement doit être soustraite par un refroidisseur supplémentaire de même puissance afin de rétablir l'équilibre du système.

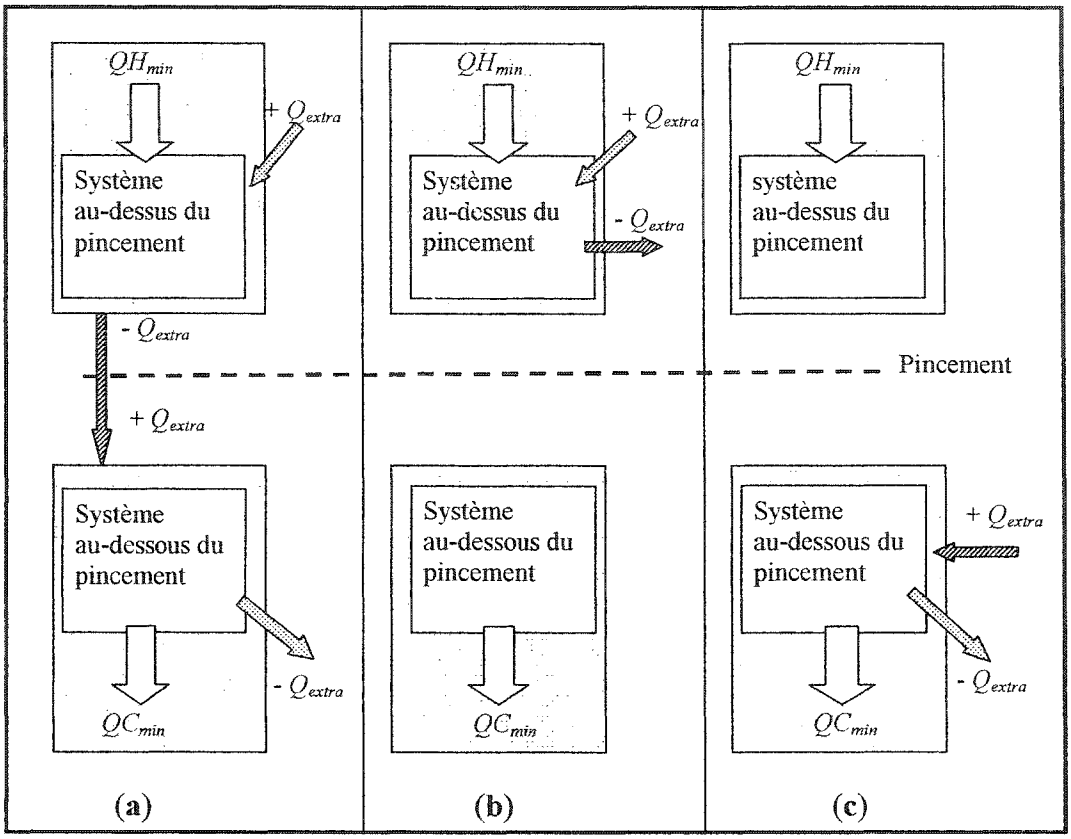


Figure 4.9 : Règles de conception de REC par la méthode du pincement

Le moyen le plus sûr de respecter les règles de conception par la méthode du pincement est de construire deux réseaux d'échangeurs de chaleur distincts en respectant la décomposition au pincement : Le réseau au-dessus du pincement comprend les courants ou parties de courants se trouvant au-dessus du pincement et le réseau au-dessous du pincement comprend les courants ou parties de courants se trouvant au-dessous du pincement (Voir la Figure 4.10).

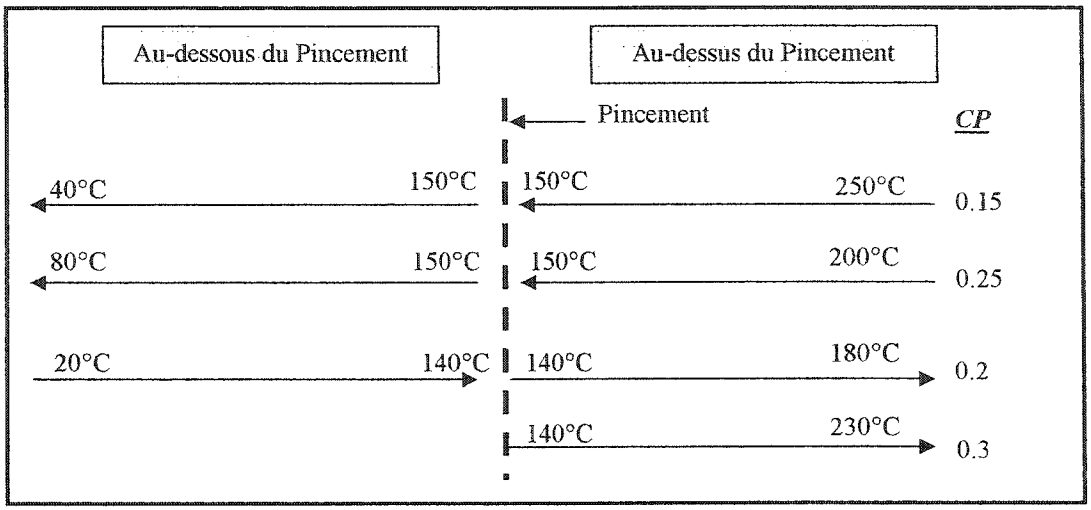


Figure 4.10 : Caractéristiques des courants au-dessus et au-dessous du pincement

4.4.2 Conception séparée d'un REC de chaque côté du pincement

Si nous observons les profils des courbes composites (Voir la Figure 4.8), nous remarquons dans chaque partie de la décomposition que :

- Au-dessus du pincement, la courbe composite des courants froids est sortante (par rapport au point de pincement), tandis que celle des courants chauds est entrante. Pour obtenir un profil de température qui diverge lorsque l'écoulement sort du pincement, la sélection de deux courants à apparier doit donc respecter les conditions suivantes :

  $$CP_c \geq CP_h \qquad [4.2]$$

- Au-dessous du pincement, la courbe composite des courants froids est entrante, pendant que celle des courants chauds est sortante. Pour obtenir un profil de température qui diverge lorsque l'écoulement sort du pincement, les appariements devront respectés les conditions suivantes :

  $$CP_h \geq CP_c \qquad [4.3]$$

où CP_h et CP_c sont respectivement la capacité calorifique des courants chauds et froids.

Les deux règles ci-dessus énoncées sont traduite par la règle suivante appelée la ***règle des CP*** : L'appariement de deux courants au pincement doit être tel que la capacité calorifique du courant sortant du pincement soit supérieure ou égale à la capacité calorifique du courant entrant au pincement (site Internet de "*The Environmental Technology Network*") [29].

$$CP_{out} \geq CP_{in} \qquad [4.4]$$

a) **Construction du réseau au-dessus du pincement**

Après la décomposition au pincement, on retrouve au-dessus du pincement deux courants chauds et deux courants froids dont les caractéristiques sont reportées dans le Tableau 4.3.

Tableau 4.3 : Caractéristiques des courants au-dessus du pincement

Au-dessus du pincement				
Courants présents	T^S (°C)	T^T (°C)	CP (MW/°C)	ΔH (MW)
H1	250	150	0,15	15
H2	200	150	0,25	12.5
C1	140	180	0,2	8
C2	140	230	0,3	27

Pour faciliter l'identification des options d'appariement des courants, nous comparons les *CP* des différents courants en utilisant un tableau appelée matrice des *CP*. Dans la matrice des *CP*, les appariements possibles sont marqués par une croix (x) et les appariements infaisables par un tiret (-). Nous présentons au Tableau 4.4 une matrice des *CP* du système au-dessus du pincement. Cette matrice indique les courants entrants et sortants en présence ainsi que leurs capacités calorifiques.

Tableau 4.4 : Matrice des *CP* au-dessus du pincement

			Courants sortants	
			C1	C2
			0.2	0.3
Courants entrants	H1	0.15	x	x
	H2	0.25	-	x

Après avoir identifié les courants à apparier, il faut déterminer la charge thermique associée à cet appariement. Ceci est généralement défini par des considérations économiques liées à l'investissement à réaliser. Or le coût d'investissement est lié au nombre d'échangeurs de chaleur utilisés. Ainsi pour minimiser le nombre d'échangeurs de chaleur, il convient de maximiser la chaleur échangée dans un appariement. Le choix d'un appariement doit être tel que l'appariement obéisse à la règle des *CP*, et que le besoin thermique du ***courant entrant*** (de préférence) puisse être entièrement satisfait (plutôt que le courant sortant, ce dernier pouvant toujours être satisfait par une utilité).

Ainsi, pour le problème défini au-dessus du pincement, les appariements nécessaires à faire au pincement sont H1/C1 et H2/C2 (Voir la Figure 4.11).

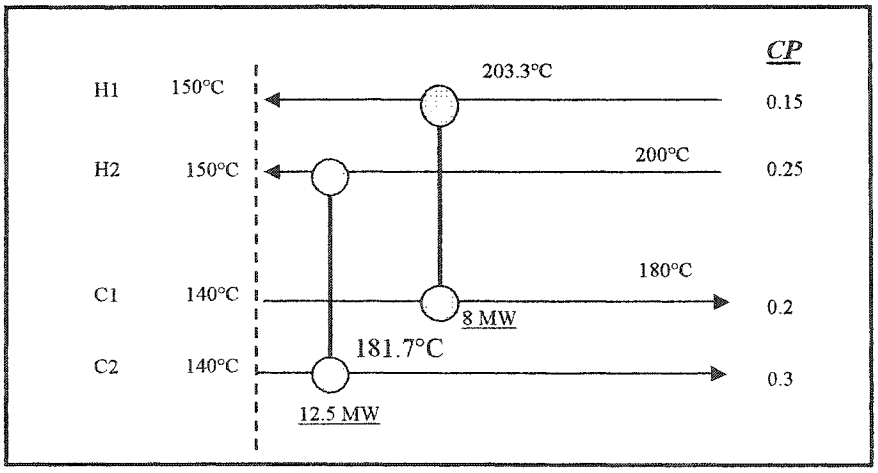


Figure 4.11 : Appariement des courants au pincement (système au-dessus)

Après avoir réalisé les appariements H1/C1 et H2/C2, la chaleur récupérable au-dessus du pincement n'est pas encore complétée et les besoins thermiques restants à satisfaire sont présentés dans le Tableau 4.5.

Tableau 4.5 : Charges thermiques résiduelles au-dessus du pincement

Au-dessus du pincement				
Courants présents	T^S (°C)	T^T (°C)	CP (MW/°C)	ΔH (MW)
H1	250	203.3	0.15	7
C2	181.7	230	0.3	14.5

L'appariement restant à faire donc est H1/C2 (Voir la Figure 4.12). Puisque les appariements à effectuer au pincement ont été déjà faits, l'appariement H1/C2 peut se faire sans tenir compte de la règle des CP.

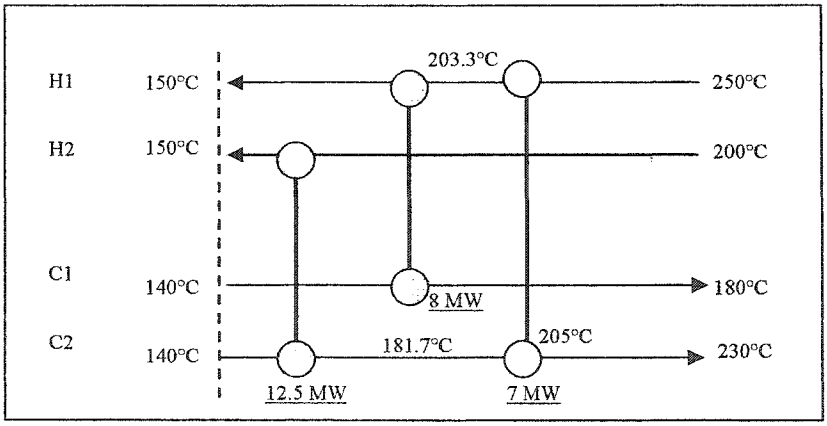


Figure 4.12 : Appariement final des courants au-dessus du pincement

Après l'appariement de H1 et C2, les besoins thermiques ne sont pas encore totalement satisfaits. Les besoins résiduels à satisfaire sont indiqués dans le Tableau 4.6.

Tableau 4.6 : Charges non récupérables au-dessus du pincement

Au-dessus du pincement				
Courants présents	T^S (°C)	T^T (°C)	CP (MW/°C)	ΔH (MW)
C2	205	230	0.3	7.5

Il reste juste à satisfaire les besoins résiduels du courant froid C2 par une utilité chaude de puissance QH_{min} = 7.5 MW (Voir la Figure 4.13).

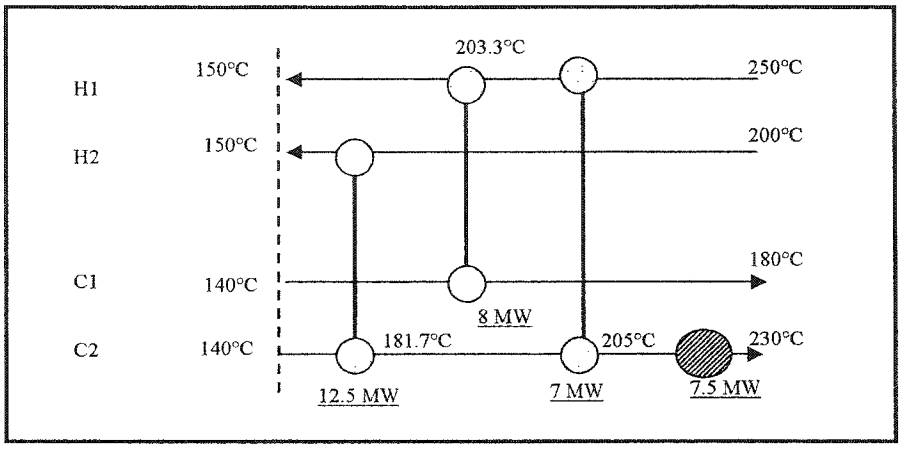


Figure 4.13 : Ajout d'une utilité chaude au-dessus du pincement

La construction du REC au-dessus du pincement est ainsi terminée (Figure 4.13). La puissance de l'utilité chaude obtenue est bien celle qui a été définie par les courbes composites (Paragraphe 4.3.2).

L'étape suivante reprend les mêmes procédures pour la construction du réseau au-dessous du pincement en respectant les mêmes règles.

b) Construction du réseau au-dessous du pincement

Au-dessous du pincement se trouvent deux courants chauds (H1 et H2) et un courant froid (C1) dont les caractéristiques sont présentées dans le Tableau 4.7

Tableau 4.7 : Caractéristiques des courants au-dessous du pincement

Au-dessous du pincement				
Courants présents	T^S (°C)	T^T (°C)	CP (MW/°C)	ΔH (MW)
H1	150	40	0.15	16.5
H2	150	80	0.25	17.5
C1	20	140	0.2	24

La matrice des *CP* présentée dans le Tableau 4.8 montre que le courant froid C1 peut être apparié à l'un ou à l'autre des deux courants chauds (H1 ou H2).

Tableau 4.8 : Matrice des CP au-dessous du pincement

			Courants sortants	
			H1	H2
			0.2	0.3
Courant entrant	C1	0.15	x	x

Nous avons deux options possibles d'appariement au pincement : H1/C1 ou H2/C1. Pour maximiser la puissance échangée dans l'appariement des courants, il est commode de choisir l'appariement H2/C1 (Voir la Figure 4.14). Cet appariement satisfait entièrement aux besoins thermique du courant chaud H2, mais le courant froid C1 n'a pas atteint la température finale désirée.

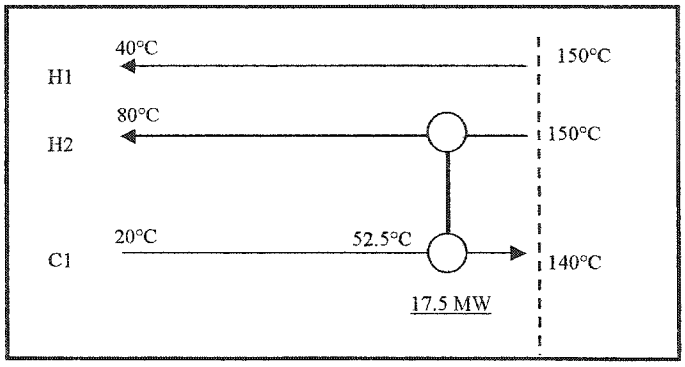


Figure 4.14 : Appariement des courants au pincement (système au-dessous)

Les besoins thermiques restants à satisfaire sont présentés dans le Tableau 4.9

Tableau 4.9 : Charges thermiques résiduelles (système au-dessous du pincement)

Au-dessous du pincement				
Courants présents	T^S (°C)	T^T (°C)	CP (MW/°C)	ΔH (MW)
H1	150	40	0.15	16.5
C1	20	52.5	0.2	6.5

Dans le Tableau 4.9, nous avons deux courants en présence : H1 et C1 qui doivent être appariés (Voir la Figure 4.15).

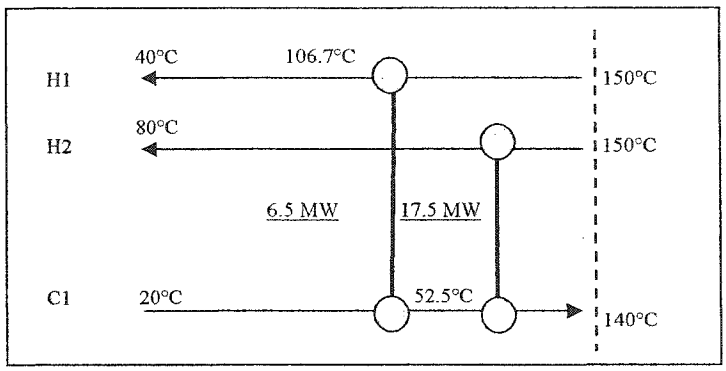


Figure 4.15 : Appariement final des courants au-dessous du pincement

Après l'appariement H1/C1, les besoins thermiques restants à satisfaire sont présentés dans le tableau 4.10.

Tableau 4.10 : Charges non récupérables au-dessous du pincement

Au-dessous du pincement				
Courants présents	T^S (°C)	T^T (°C)	CP (MW/°C)	ΔH (MW)
H1	106.7	40	0.15	10

L'appariement H1/C1 a donc complété la récupération de chaleur identifiée au-dessous

du pincement. Il ne reste qu'à satisfaire aux besoins résiduels du courant chaud H1 par une utilité froide de puissance QC_{min} = 10 MW(Voir la Figure 4.16)

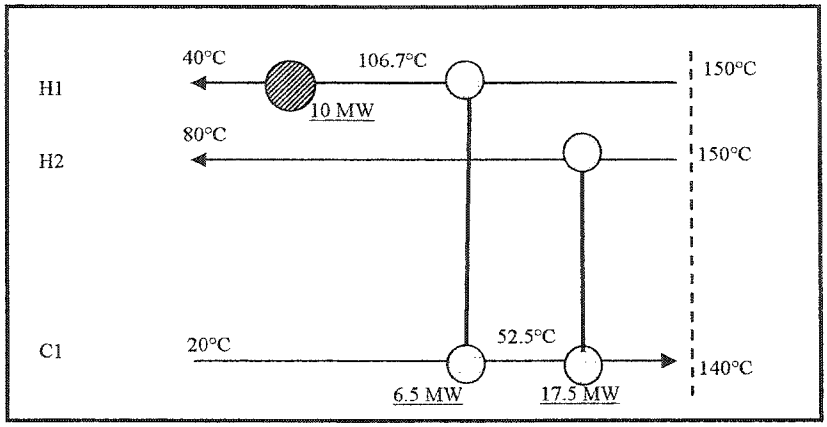


Figure 4.16 : Ajout d'une utilité froide au-dessous du pincement

La construction du réseau d'échangeurs de chaleur au-dessous du pincement et terminée. La puissance de l'utilité froide utilisée est celle précédemment définie par les courbes composites (Paragraphe 4.3.2).

La construction des deux réseaux étant complétée, il ne reste plus qu'à les fusionner pour former un réseau unique, correspondant au réseau initial pour la récupération de chaleur.

4.4.3 Fusion des deux réseaux séparés

Le conception du réseau est complétée en fusionnant les deux portions de réseaux conçus au-dessus et au-dessous du pincement (Voir la Figure 4.17). Le réseau ainsi conçu par la méthode du pincement comporte 2 utilités (une utilité froide et une utilité chaude) et 5 échangeurs de chaleur, soit au total sept unités, qui permettent de récupérer une puissance maximale Q_{REC} = 51.5 MW.

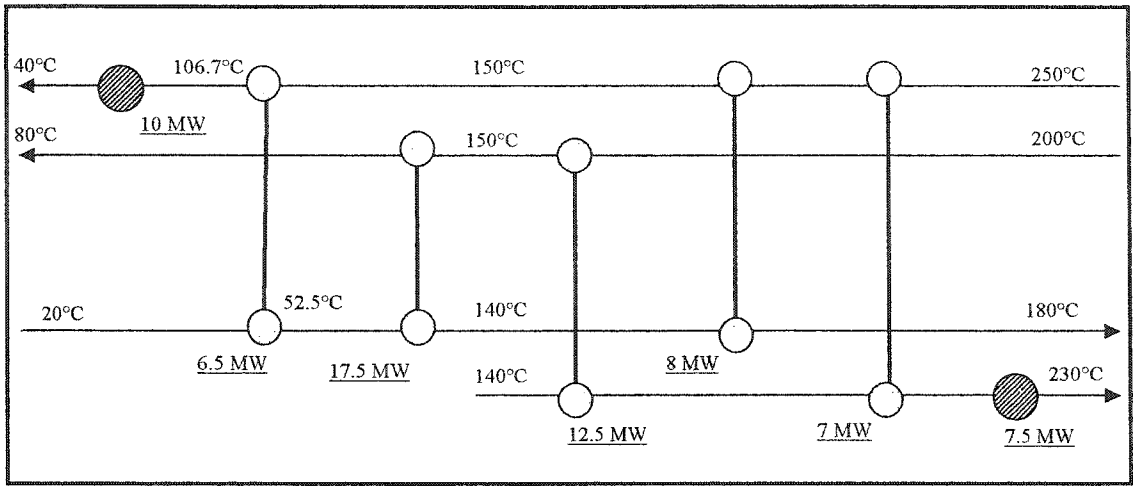


Figure 4.17 : Réseau initial pour la récupération de chaleur

4.4.4 Optimisation du réseau initial

Le réseau d'échangeurs de chaleur construit selon les règles du pincement, doit être vu comme un réseau initial qui peut être sujet à une éventuelle optimisation. L'optimisation concerne des changements d'ordre topologique et paramétrique du réseau initial, dans le but de minimiser le coût total. Les trois degrés de liberté disponibles pour l'optimisation du réseau sont : la ***division des courants***, les ***boucles de charge thermique*** et les ***chemins de charge thermique.***

a) La division des courants

Le design par la méthode du pincement suit certaines règles (la règle des *CP*) afin de permettre la construction de réseaux d'échangeurs de chaleur qui récupèrent le maximum de chaleur avec un nombre minimum d'unités. Lors de la conception du réseau, il peut arriver que certains appariements essentiels à faire au pincement ne peuvent l'être parce que la règle des *CP* n'est pas respectée. Cela a pour conséquence la construction d'un réseau initial qui ne récupère pas la chaleur maximale disponible.

Considérons par exemple le schéma de la Figure 4.18.a (Smith, 1995) [27], illustrant un système quelconque au-dessus du pincement.

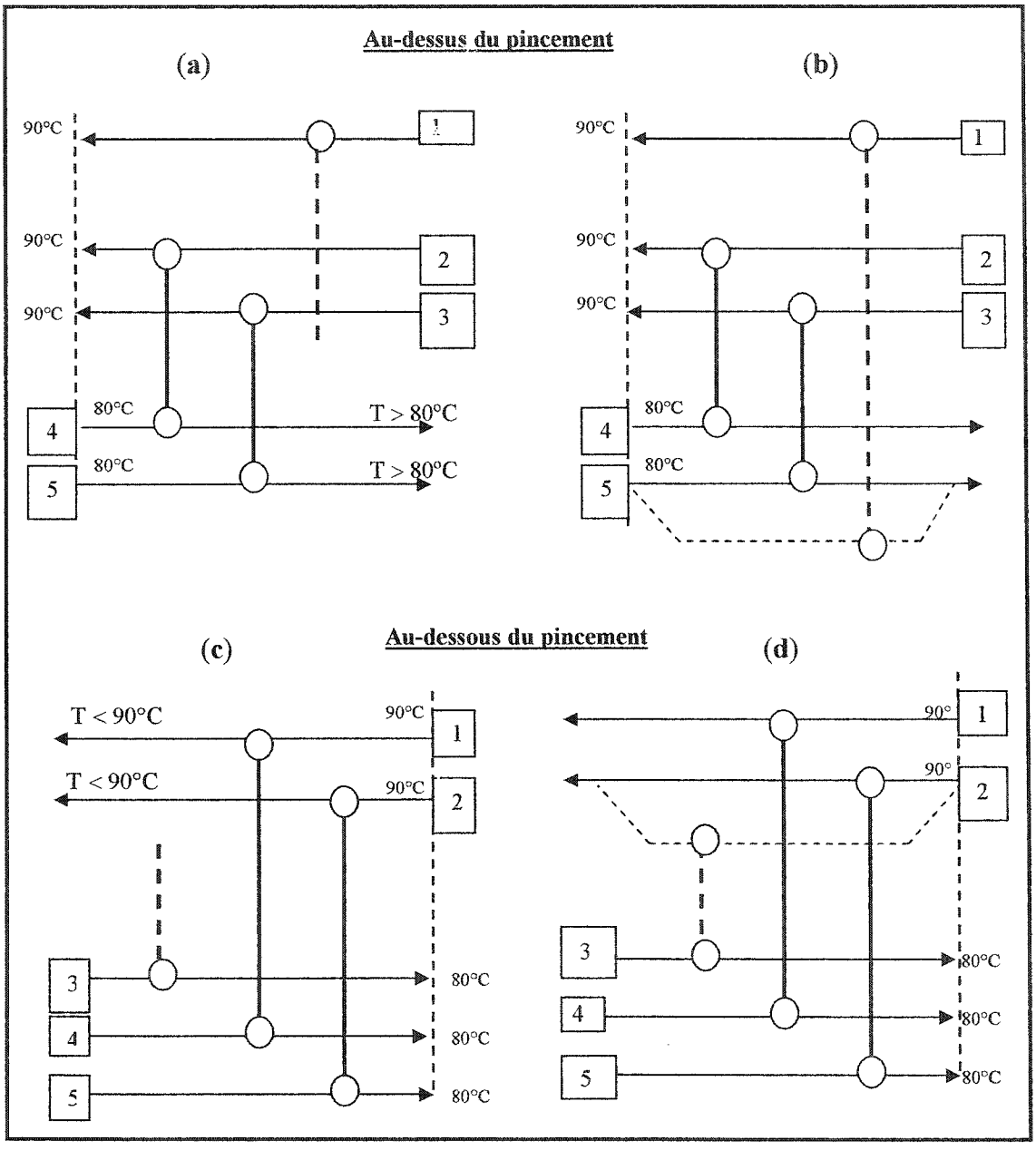


Figure 4.18 : Division des courants

La mise en place d'un échangeur de chaleur entre les courants chaud (2) et froid (4), ainsi qu'entre les courants chaud (3) et froid (5) permet d'amener les courants chauds (2) et (3) à la temperature de pincement. Il reste cependant le courant chaud (1) qui ne peut plus être apparié avec aucun des courants froids sans violer la contrainte de température minimale d'approche ΔT_{min}. Pour amener le courant (1) à la température de pincement, il faudra donc recourir à une utilité froide (installée au-dessus du pincement), ce qui est contraire aux règles de la méthode du pincement. Pour contourner cette incompatibilité, il faut corriger les courants au pincement en procédant à une ***division de courant.*** Cette procédure est présentée à la Figure 4.18.b. En divisant un courant froid, nous créons une nouvelle branche qui permet au courant chaud (1) d'avoir un partenaire d'appariement. Les Figures 4.18.c et 4.18.d montrent la même situation de division de courants au-dessous du pincement et la solution apportée. Pour assurer une récupération maximale de chaleur, le nombre de courants au pincement doit être tel que :

- Au-dessus du pincement : $N_h \leq N_c$ [4.5]

- Au-dessous du pincement : $N_h \geq N_c$ [4.6]

où N_h et N_c sont respectivement le nombre de courants chauds et froids au pincement.

b) Les boucles de charge thermique

Grimes (1980) [30] a montré que la minimisation du nombre d'unités et la maximisation de la puissance récupérée sont souvent incompatibles dans certains problèmes de réseaux d'échangeurs de chaleur. Cette incompatibilité est essentiellement due à la décomposition du problème au pincement. En appliquant la règle des (N – 1) au problème divisé au pincement, (Figure 4.10), nous avons : $U_{\min} = 3+4 = 7$

Par contre l'application de la même règle au problème en entier sans la division au pincement, donne le nombre minimum d'unités $U_{\min} = 6-1 = 5$

C'est-à-dire donc que pratiquement, pour réaliser un réseau avec 5 unités, il faut satisfaire aux conditions suivantes:

- Une quantité de chaleur (ΔQ) doit être transférée à travers le pincement, ce qui a pour conséquence d'augmenter les besoins en utilités chaudes et froides.
- La contrainte de la température minimale ΔT_{min} imposée doit être relaxée au niveau de certains échangeurs de chaleur.

Dans le premier cas, en choisissant de transférer de la chaleur à travers le pincement, donc d'augmenter les besoins en utilités, on peut réduire le nombre d'unités. Il y a donc un compromis à faire entre le nombre d'unités (coût d'investissement), et les besoins en utilités (coût de l'énergie). Dans le second cas il y a un compromis de coût à faire entre le nombre d'unités et la surface de transfert de chaleur.

Le réseau initial établi en respectant la décomposition au pincement va donc souvent comporter plus d'unités que le minimum requis. Ceci va créer dans le réseau des ***boucles de charge thermique***. Une boucle de charge thermique est un circuit fermé composé d'échangeurs de chaleur tel que la redistribution des charges entre ces derniers n'affecte pas la consommation totale d'énergie du réseau. Il y'a une boucle pour chaque unité excédentaire. Ce phénomène a été observé par Hohmann, (1971) [31] et confirmé plus tard par Linnhoff et al., (1979) [32].

Le réseau initial de la Figure 4.17 (établi en utilisant la décomposition au pincement) comporte deux unités excédentaires ; il y aura donc deux boucles de charge thermique dans le réseau (Voir les Figure 4.19.a et 4.19.b). De telles boucles peuvent être utilisées pour éliminer les petites unités du réseau ou simplement redistribuer la surface totale de transfert de chaleur dans le réseau afin d'assurer un coût total annuel minimum.

Une propriété importante des boucles est que les charges thermiques peuvent être déplacées autour de la boucle, d'une unité à une autre de sorte à minimiser les coûts d'investissements liés à la surface des échangeurs de chaleur. Cette procédure de déplacement de la charge thermique a été présentée par Linnhoff et Hindmarsh (1983)

[2], et se déroule de la façon suivante: Une charge thermique (ΔQ) est soustraite d'une unité, ajoutée à l'unité suivante dans la boucle, soustraite de l'unité qui suit et ainsi de suite tout autour de la boucle

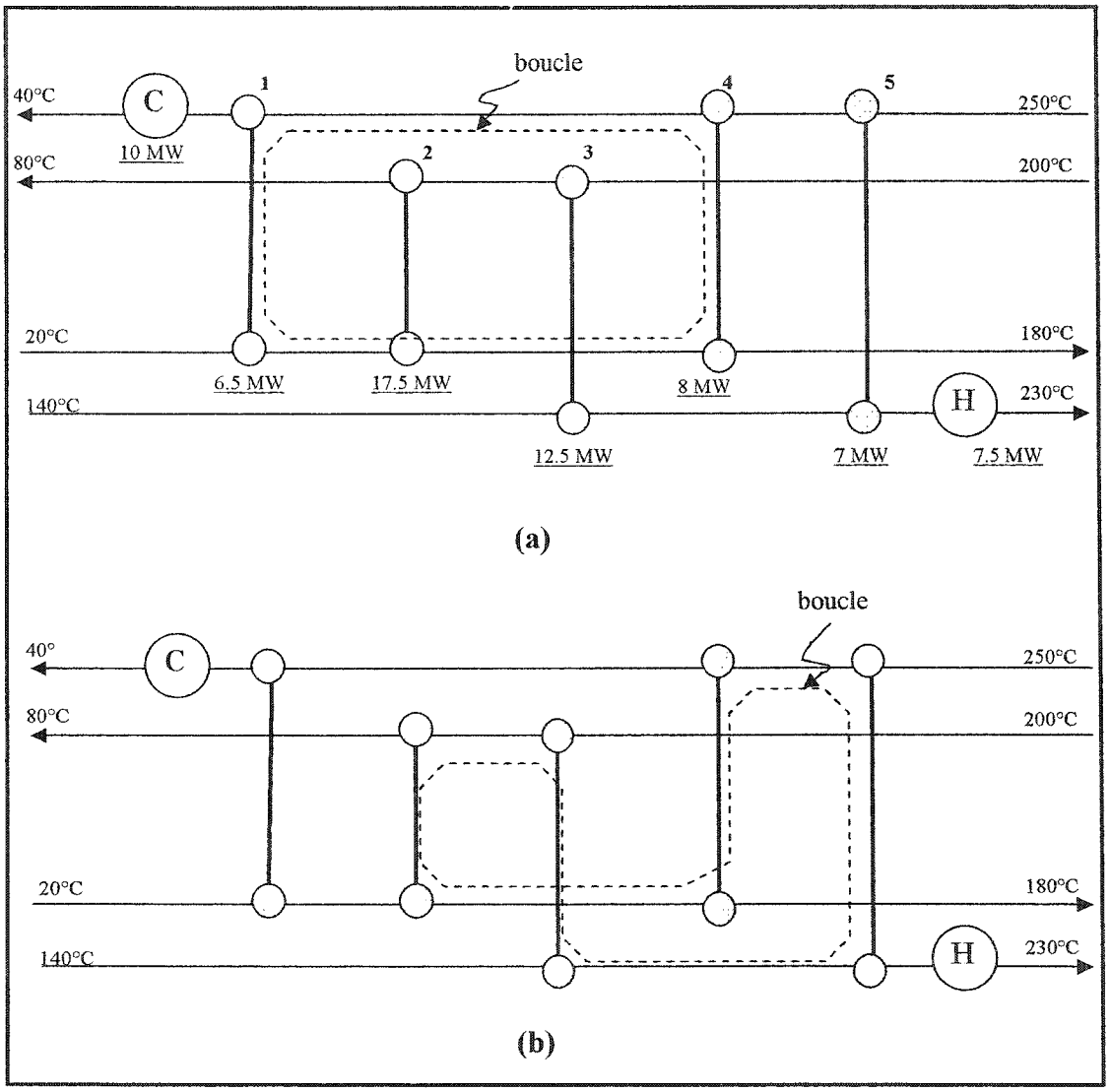


Figure 4.19 : Boucles de charge de thermique

En pratique, le choix de ΔQ résulte d'une optimisation en évaluant le coût du réseau pour différentes valeurs de ΔQ. Appliquons le principe des boucles de charge thermique à la boucle de la Figure 4.19.a. en choisissant ΔQ égale à la charge thermique de l'échangeur de chaleur (1). La valeur de ΔQ est soustraite de l'échangeur de chaleur (1) et ajoutée à l'échangeur de chaleur (4). La charge résultante sur l'échangeur (1) est zéro et il est supprimé du réseau; la boucle est ainsi supprimée. L'échangeur de chaleur (4) a maintenant une charge thermique de (6,5 + 8) MW (Voir la Figure 4.20). Le déplacement de la charge thermique autour de la boucle maintient la charge thermique des courants inchangée, mais les charges des échangeurs de chaleur sont modifiées, pouvant entraîner la violation de la contrainte de température minimale d'approche.

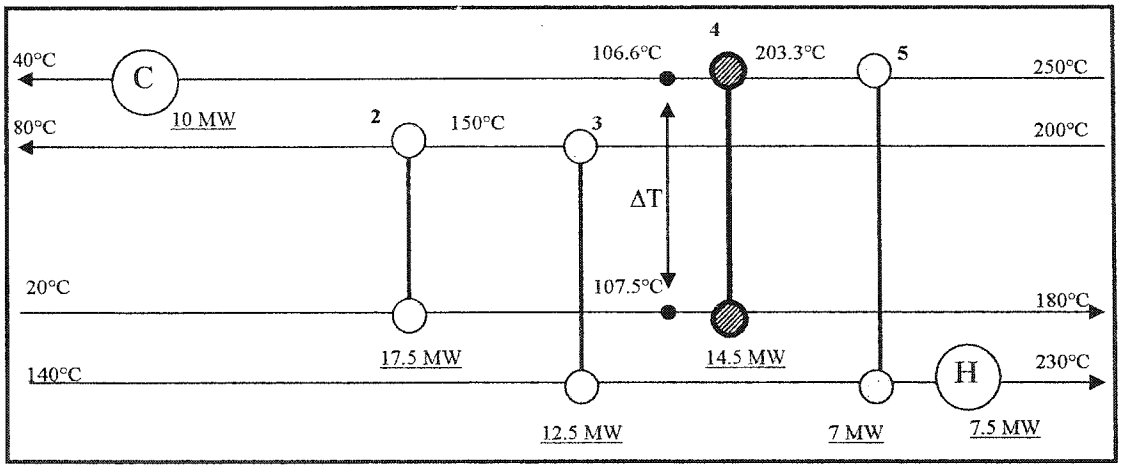


Figure 4.20 : Déplacement d'une charge thermique autour de la boucle

Aux bornes de l'échangeur (4), ΔT = 106.6 – 107.5 $\cong$ -1°C . Cette différence de température est infaisable. Elle peut cependant être restaurée à une valeur supérieure ou égale à la température minimale d'approche (ΔT_{min}) en utilisant le ***chemin de charge thermique***.

c) Les chemins de charge thermique

Un chemin de charge thermique est une séquence quelconque d'échangeurs de chaleur assurant la liaison entre une utilité chaude et une utilité froide. Le déplacement d'une charge thermique le long d'un chemin de charge affecte aussi bien les coûts d'investissements liés à la surface des échangeurs de chaleur que les coûts des utilités extérieures utilisées. La Figure 4.21 illustre un des chemins dans le réseau d'échangeurs de chaleur obtenu précédemment par déplacement de charge thermique. Le chemin (en trait pointillé) assure la liaison continue entre l'utilité chaude (H), les échangeurs de chaleurs (3), (2), (4) et l'utilité froide (C).

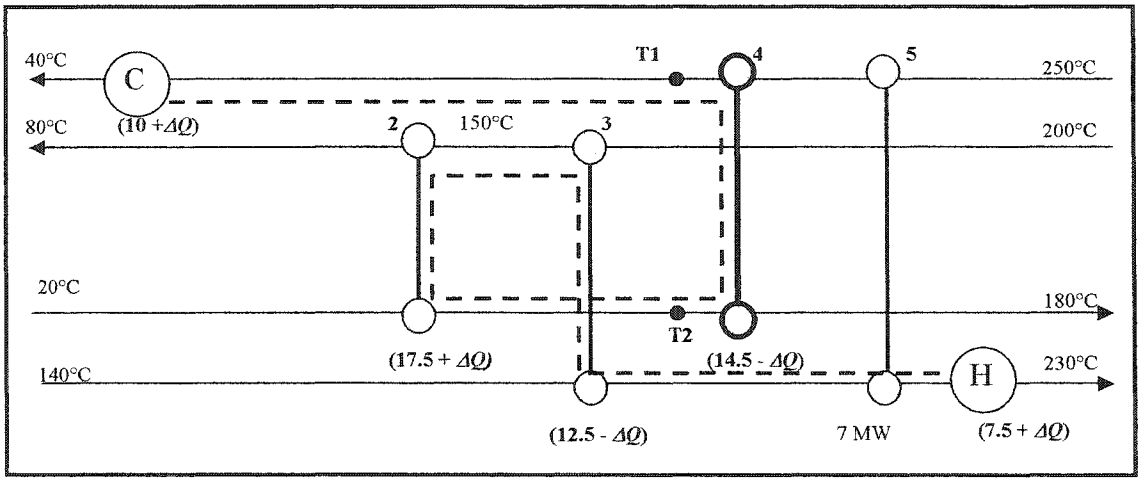


Figure 4.21 : Chemin de charge thermique

Le déplacement des charges thermiques le long des chemins de charges thermiques suit des règles identiques à celles du déplacement de charges autour des boucles. Une charge thermique (ΔQ) est ajoutée à une utilité chaude, soustraite d'un échangeur de chaleur, ajoutée à l'échangeur de chaleur suivant sur le chemin, soustraite de l'échangeur de chaleur suivant et ainsi de suite le long du chemin jusqu'à ce quelle soit finalement ajoutée à une utilité froide. Dans cette opération, les charges des échangeurs de chaleur subissent des modifications de même que les températures d'opération. Cette dernière

propriété est utilisée pour restaurer certaines différences de températures inacceptables qui surviennent dans le réseau après la manipulation des boucles de charges thermiques.

À la Figure 4.21, la différence de température entre le courant chaud sortant de l'échangeur (T1) et le courant froid entrant dans l'échangeur (T2) est ΔT = -1°C. Cette température peut être restaurée à la température minimale d'approche (ΔT_{min} = 10°C) à l'aide du chemin de charge thermique. La valeur de la charge thermique (ΔQ) à déplacer le long du chemin est ΔQ = 6.5 MW. Le réseau d'échangeurs de chaleur résultant est présenté à la Figure 4.22.

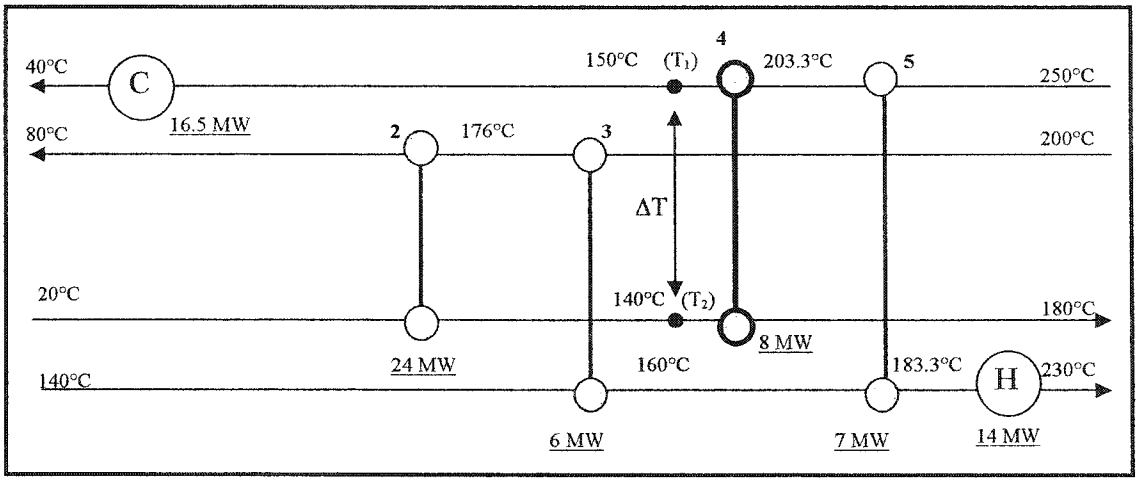


Figure 4.22 : Restauration de la température minimale d'approche.

La différence de température aux extrémités de l'échangeur de chaleur (4), infaisable après le déplacement de charge thermique dans la boucle, a été restaurée (ΔT= 10°C). Dans la configuration de la Figure 4.22, nous avons 6 unités dont 4 échangeurs de chaleur, 1 utilité chaude (H) et 1 utilité froide (C). Il subsiste encore dans la structure du réseau, une boucle de charge thermique qu'on peut éliminer en suivant les mêmes procédures que précédemment.

Le réseau obtenu après réduction du nombre d'unités n'est pas forcément le réseau le plus économique. Des études économiques doivent être faites par rapport au nombre d'unités, la surface totale d'échange et l'énergie utilisée. Plusieurs scénarios doivent donc être envisagés en appliquant les propriétés de boucles de charge thermique et de chemin de charge thermique en vue de déterminer la configuration la plus économique.

4.5 Conclusion

Nous avons présenté dans ce chapitre, la méthode du pincement qui est fondamentale pour la compréhension des problèmes liés à la conception des réseaux d'échangeurs de chaleur .Elle présente un certain nombre de concepts, de représentations et de diagrammes graphiques qui sont d'importants outils d'aide à l'apprentissage et à la compréhension de la méthode. Elle constitue aussi un puissant outil pour des applications industrielles. Un des inconvénients de la méthode est qu'elle s'exécute manuellement et la résolution des problèmes de grande taille devient complexe. De plus elle n'indique pas comment prendre en compte les contraintes lors de la conception.

CHAPITRE V

5 SYNTHÈSE DE RÉSEAUX D'ÉCHANGEURS DE CHALEUR PAR LES ALGORITHMES GÉNÉTIQUES

5.1 Introduction

La technologie du pincement a été utilisée avec beaucoup de succès dans les applications industrielles, prouvant que la méthode est applicable et fiable. Cependant, son application peut être limitée, lorsque les réseaux deviennent plus complexes, avec des contraintes souvent contradictoires.

Le développement de l'informatique offre aujourd'hui la possibilité de traiter des problèmes d'optimisation très complexes avec des temps de calcul très courts. Des techniques d'optimisation tirant avantages des possibilités qu'offrent les outils informatiques ont été développées. L'une de ces méthodes que nous utilisons dans ce projet est l'algorithme génétique qui présente une meilleure performance et une souplesse d'utilisation par rapport aux méthodes conventionnelles d'optimisation Nous présentons dans ce chapitre une procédure d'optimisation des systèmes de récupération de chaleur basée sur les algorithmes génétiques.

5.2 De la sélection naturelle à l'optimisation des systèmes industriels

La notion de ***sélection naturelle*** apparaît pour la première fois dans ***"The Origin of species"*** de Darwin (1859) (Site Internet de POLYTECH'LILLE) [33]. Il s'agissait pour lui de rendre compte par là du processus par lequel les espèces ont pu s'adapter. Darwin montre que l'apparition d'espèces distinctes se fait par le biais de la sélection naturelle. Cette sélection naturelle est l'idée que les modifications successives des générations sont

orientées par les pressions extérieures à l'espèce, comme la limitation des ressources, les modifications de l'environnement, les prédateurs et parasites, en fait tout ce qui influence la survie et la reproduction des individus (Site Internet de WIKIPÉDIA) [34]. Il en résulte que les individus les plus adaptés à l'environnement (***the fittest*** en anglais) tendent à survivre plus longtemps et à se reproduire plus aisément.

S'inspirant du mécanisme de la sélection naturelle développé par Darwin, Holland (1975) [3], a posé les bases d'une technique d'optimisation appelée "algorithmes génétiques". Un de ses étudiants, Goldberg,(1989) [20], s'est investi dans l'étude des AG et est à l'origine du développement de la méthode moderne actuelle des AG que nous connaissons. Dans l'approche des algorithmes génétiques, l'optimisation d'un système est simulée par l'évolution d'une population d'individus divers (généralement tirée aléatoirement au départ) à laquelle on applique différents opérateurs génétiques (recombinaisons, mutations, etc.) et que l'on soumet à une sélection à chaque génération. Le vocabulaire employé dans les AG est directement calqué sur celui de la théorie de l'évolution et de la génétique. Ainsi les termes comme individus, population, gènes, chromosomes, parents, descendants, reproduction, croisement et mutation sont utilisés.

Deux aspects importants caractérisent toute stratégie de recherche de solutions optimales: Il s'agit de l'exploitation des meilleures solutions et l'exploration de l'espace de recherche (Michalewicz ,1996) [35]. La méthode de ***l'ascension de la pente*** est une stratégie de recherche qui exploite les meilleures solutions susceptibles d'améliorer le résultat, tout en ignorant l'exploration de l'espace de recherche. La ***recherche aléatoire*** quant à elle, est une stratégie de recherche qui explore l'espace de recherche tout en ignorant d'exploiter les régions les plus prometteuses. Les ***algorithmes génétiques*** constituent une méthode de recherche robuste combinant les éléments d'une recherche dirigée et stochastique, pouvant faire un équilibre entre l'exploitation et l'exploration de l'espace de recherche. Ils diffèrent des autres méthodes conventionnelles d'optimisation et procédures de recherche en plusieurs aspects:

- Ils manipulent un codage de solutions et non les solutions elles-mêmes
- Ils cherchent à partir d'une population de solutions et non une seule solution
- Ils utilisent l'information sur le gain (fonction d'adaptation) et non les dérivés
- Ils utilisent des règles de transition probabilistes et non déterministes

5.3 *Organisation des algorithmes génétiques*

Les algorithmes génétiques manipulent en général un codage des paramètres (et non les paramètres eux-mêmes). La première étape dans la mise en œuvre d'un algorithme génétique consiste à définir et à coder convenablement le problème. À chaque paramètre (ou variable) X_k du problème à optimiser, correspond un ***gène***. Le gène est défini par une séquence de symboles (nombres binaires, nombres entiers, alphabet, etc.) de longueur n_b, l'équivalent de l'ADN génétique. Un ensemble de gènes (les variables du problème) forme un ***chromosome*** (Figure 5.1). Si le problème à résoudre comporte une seule variable, alors le chromosome comportera un seul gène.

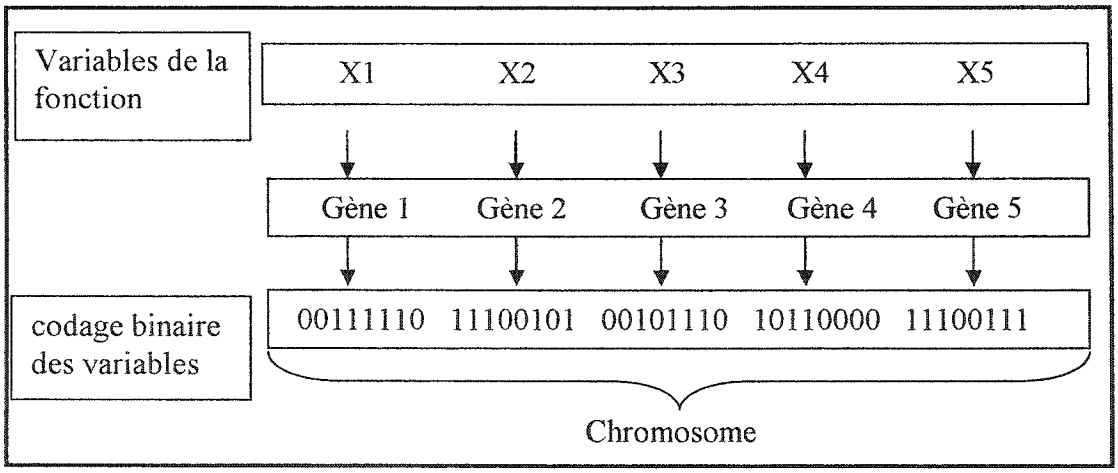


Figure 5.1 : Structure d'un chromosome

La ***population*** est un ensemble de plusieurs chromosomes simulant différentes solutions potentielles au problème d'optimisation.

5.4 Fonctionnement des algorithmes génétiques

La structure générale de fonctionnement d'un algorithme génétique décrite par Mélanie (1996) [36] est la suivante :

1 - Génération aléatoire d'une population de chromosomes.
2 - Évaluation de l'adaptation (fitness) de chaque chromosome.
3 - Création d'une nouvelle population en répétant les étapes suivantes jusqu'à ce que la nouvelle population soit complète :
    a) Sélection de deux chromosomes parents en fonction de leur fitness.
    b) Avec une probabilité de croisement, croiser le deux parents pour former deux nouveaux chromosomes (enfants). Si les deux parents ne sont pas croisés, les enfants sont une réplique exacte des parents.
    c) Avec une probabilité de mutation, muter les nouveaux enfants.
    d) Placer les nouveaux enfants dans la population.
4 - Utilisation de la nouvelle population pour la prochaine exécution de l'algorithme.
5 - Si la condition d'arrêt est satisfaite, alors arrêter la recherche et retenir la meilleure solution. Dans le cas contraire, continuer à l'étape 6.
6 - Retourner à l'étape 2.

Chaque itération de cette procédure est appelée ***génération***. L'ensemble de toutes les générations est appelé ***lancement***. À la fin d'un lancement il y a souvent dans la population finale un ou plusieurs chromosomes hautement adaptés à la fonction. La procédure décrite ci-dessus est la base de la plupart des applications des algorithmes génétiques.

5.4.1 Génération de la population initiale

Un des avantages des algorithmes génétiques sur les autres méthodes d'optimisation est qu'ils manipulent simultanément un ensemble de solutions et non une solution unique.

Ils commencent avec plusieurs solutions représentées par les chromosomes. Ces chromosomes dont le nombre est constant (fixé par l'opérateur) sont généralement générés de façon aléatoire. Chaque chromosome est considéré être une solution potentielle au problème posé. Cette population initiale servira au lancement de l'algorithme génétique et évoluera d'une génération à une autre grâce à des opérations simples qui ne font intervenir que des copies et des échanges de morceaux de chromosomes. Il s'agit de la reproduction qui fait appel aux opérateurs de croisement de mutation et de sélection.

5.4.2 Évaluation des solutions potentielles

Un algorithme d'optimisation nécessite généralement la définition d'une fonction rendant compte de la pertinence des solutions potentielles, à partir des grandeurs à optimiser. Cette fonction f est nommée ***fonction d'adaptation*** (ou ***fitness function*** ou simplement ***fitness*** en terminologie anglo-saxonne). L'algorithme convergera vers un optimum de cette fonction, quelle que soit sa définition. La fonction f doit donc exprimer le plus fidèlement possible le désir de l'utilisateur sous forme mathématique. C'est une fonction des variables X_k du problème à résoudre. Chaque chromosome de la population sera évalué par rapport à cette fonction pour donner une mesure exacte de son adaptation (ou son fitness), donc de sa capacité à être une solution au problème posé.

5.4.3 La sélection

La procédure de recherche par les algorithmes génétiques est une procédure itérative qui manipule une population initiale de solutions arbitraires pour l'amener vers un état où les individus de la population sont améliorés. L'évolution de la population vers l'optimum est due à des opérations de croisement, de mutation et de sélection , qui sont effectuées à chaque génération. Si la sélection s'opère à partir de la fonction d'adaptation, alors la population tend à s'améliorer (Bäck, 1996) [37].

Selon la théorie de Darwin sur l'évolution, ce sont les meilleurs individus (ceux qui sont adaptés à l'environnement) qui survivent et procréent. Les chromosomes qui sont sélectionnées pour former la nouvelle génération sont donc sélectionnés en fonction de leur adaptation (fitness). Un individu a d'autant plus de chance d'être sélectionné qu'il présente une performance élevée.

Il semble que deux éléments guident le processus d'évolution dans la recherche génétique (Michalewicz, 1996) [35]. Il s'agit de la pression de sélection et de la diversité de la population. La pression de sélection définit en quelque sorte un degré d'adaptation que doit avoir un chromosome pour être sélectionné comme une solution prometteuse. La diversité de la population se mesure par la similitude des gènes des chromosomes. Plus la population comporte des chromosomes qui se ressemblent, moins on explore l'espace de solutions. Ces deux facteurs sont intimement liés : Ainsi, une augmentation de la pression de sélection entraîne la diminution de la diversité génétique de la population et vice versa. En d'autres termes, une forte pression de sélection peut entraîner une convergence prématurée vers la solution. Une pression de sélection faible peut cependant entraîner une recherche inefficace. Il est donc important de trouver le juste équilibre entre ces deux facteurs.

Plusieurs techniques de sélection peuvent être utilisées pour la sélection des meilleurs chromosomes en vue de la reproduction. Citons entre autres la méthode de la 'roue biaisée', la méthode de 'Boltzmann', la méthode 'par tournoi', la méthode du 'classement', la méthode de 'mise à l'échelle', etc. Nous décrirons ici deux méthodes de sélection, notamment les méthodes de la roue biaisée et du classement.

a) La méthode de la roue biaisée

Les chromosomes qui participent à la reproduction sont sélectionnés en fonction de leur ***fitness***. Nous pouvons illustrer cette méthode par une cible divisée en plusieurs secteurs,

chacun ayant une largeur proportionnelle au fitness du chromosome qui lui est associé (Voir la Figure 5.2). Supposons que pour sélectionner un secteur, nous tirons une flèche sur la cible. Évidemment, les secteurs ayant une plus grande surface auront une chance plus élevée d'être sélectionnés. À la Figure 5.2, nous présentons une cible divisée en 4 secteurs de largeurs proportionnelles aux fitness (arbitraires), de 4 chromosomes donnés. Nous pouvons voir que le secteur 2 représente à elle seule près de 80% de la cible. Le chromosome qui lui est associé est un ***super individu*** qui aura plus de chance d'être sélectionné plusieurs fois pour participer à la reproduction. Il s'ensuit alors que les individus dans les générations futures seront essentiellement constitués de ses descendants. En quelques générations, le ***super individu*** peut provoquer l'élimination complète de ses congénères. On obtient alors une population homogène contenant un seul type d'individu, conduisant à une convergence prématurée vers un optimum, probablement local.

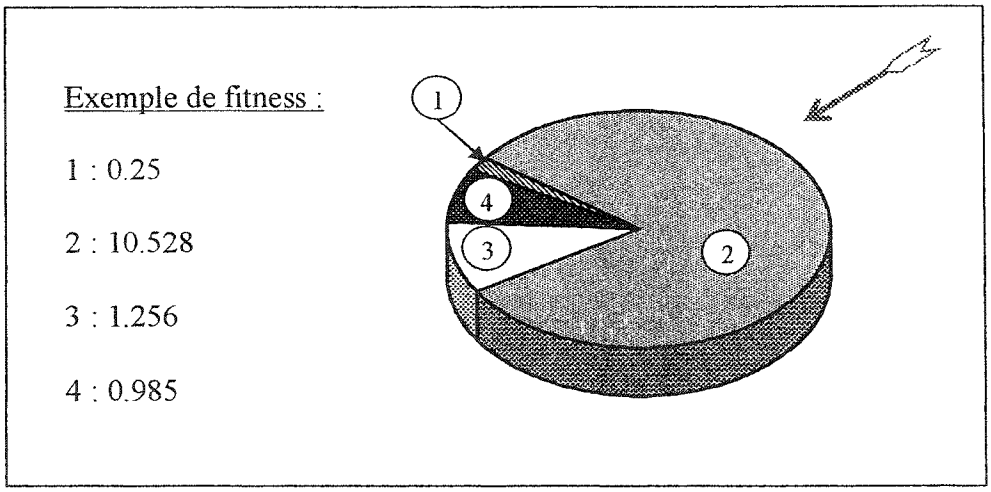


Figure 5.2 : Cible divisée en secteurs proportionnels au fitness

La méthode de sélection de la roue biaisée peut être simulée par l'algorithme suivant :

1- Calculer la somme (SOM) des fitness de tous les chromosomes

2- Générer un nombre aléatoire (α) dans l'intervalle (0, SOM)

3- Parcourir la population en faisant la sommation des fitness des chromosomes. Dès que la somme (SOM) devient supérieure à (α), arrêter et sélectionner le chromosome se trouvant à cet emplacement.

b) La méthode du 'classement'

C'est une méthode qui permet de corriger la méthode précédente, en empêchant les individus 'forts' d'éliminer les plus 'faibles'. Elle modifie les fitness originaux afin de réduire ou d'amplifier artificiellement les écarts entre individus. La méthode du classement procède d'abord à un classement des chromosomes en fonction de leurs fitness. Ensuite chaque chromosome reçoit un fitness de classement déterminé par son rang. Le moins bon reçoit la valeur 1, le suivant reçoit la valeur 2 et ainsi de suite jusqu'au meilleur qui reçoit la valeur de fitness de classement égale au nombre de chromosomes dans la population. En utilisant l'exemple des fitness de la méthode précédente (Figure 5.2), nous présentons la procédure d'attribution des fitness dans la méthode de classement (Figure 5.3).

Fitness originaux	Rang	Fitness attribués
Chromosome 1 : 0.25	1	Chromosome 1 : 1.0
Chromosome 2 : 10.528	4	Chromosome 2 : 4.0
Chromosome 3 : 1.256	3	Chromosome 3 : 3.0
Chromosome 4 : 0.985	2	Chromosome 4 : 2.0

Figure 5.3 : Attribution des fitness de classement

La sélection n'opère plus sur le fitness réel du chromosome, mais sur le fitness de classement. À la Figure 5.4, nous pouvons voir comment la situation change en procédant par la méthode du classement. Après le classement, un plus grand nombre de chromosomes a la chance d'être sélectionné. Cette méthode permet d'éviter les effets des super individus, mais en contrepartie, elle peut contribuer à ralentir la convergence, parce que le meilleur chromosome ne diffère pas beaucoup des autres.

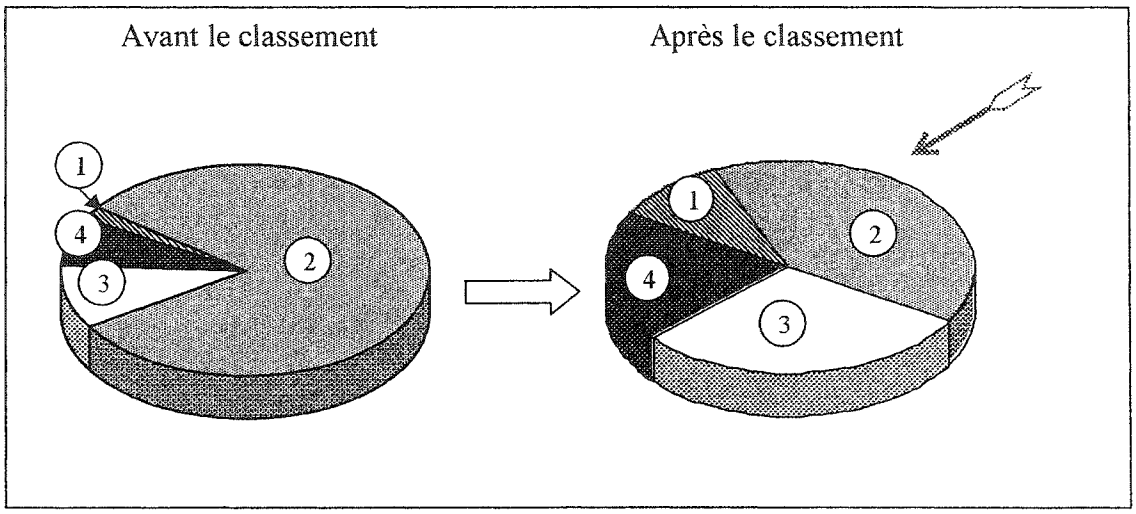


Figure 5.4 : Cible divisée en secteurs proportionnels aux fitness de classement

5.4.4 Le croisement

Le croisement est motivé par l'espoir que la nouvelle population sera meilleure que l'ancienne. C'est une opération qui permet d'exploiter les meilleurs chromosomes de l'espace de solutions. Le croisement est défini par la probabilité de croisement (Pc) dont la valeur est généralement fixée par l'opérateur. La procédure du croisement est effectuée de la façon suivante : Deux chromosomes parents sont sélectionnés de façon arbitraire et un mécanisme de tirage au sort est ensuite employé. Si un nombre aléatoire (entre 0 et 1) généré par l'algorithme est inférieur ou égal à Pc, les deux chromosomes parents doivent subir un croisement. Une position arbitraire kc est déterminée le long de

la chaîne de caractères (1, λ-1) de ces chromosomes et deux nouveaux chromosomes enfants sont alors créés en échangeant tous les caractères compris entre la position 1 et *kc* des chromosomes parents. Si le nombre aléatoire généré par l'algorithme est supérieur à *Pc*, alors aucun croisement n'est admis pour ce couple de chromosomes parents. Les nouveaux enfants reproduits sont une copie exacte des parents.

Si la probabilité de croisement est fixée à $Pc = 1$ (soit 100%), alors tous les enfants sont reproduits par croisement. Si elle est fixée à $Pc = 0$ (soit 0%), aucun chromosome enfant n'est issu d'un croisement de deux parents. La Figure 5.5 donne le résultat de croisement entre deux chromosomes parents (A et B) donnant naissance à deux enfants (A' et B'). Chaque nouveau chromosome (enfant) ainsi créé combine les caractéristiques de deux chromosomes parents.

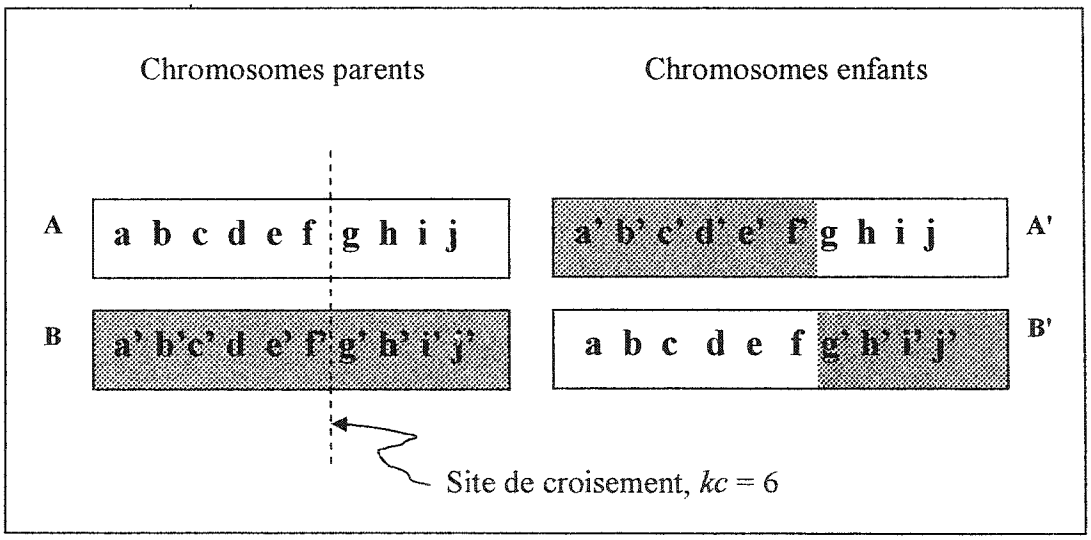


Figure 5.5 : Principe du croisement de deux chromosomes

Le croisement permet donc d'exploiter les ressources existantes tandis que la mutation permet de rechercher de nouvelles variétés génétiques.

5.4.5 La mutation

La mutation est une altération occasionnelle et aléatoire d'une position dans la chaîne de caractères des enfants issus de l'étape de croisement. Le paramètre principal de la mutation est la probabilité de mutation (*Pm*) qui est fixée par l'opérateur. Dans cette étape, un tirage au sort est effectué pour chaque chromosome enfant en générant un nombre aléatoire compris entre 1 et 0. Si le nombre aléatoire généré est inférieur à *Pm*, le chromosome subit une mutation. Dans un codage binaire, un ou plusieurs bits sont inversés; ainsi 0 devient 1 et 1 devient 0. Si le nombre aléatoire du tirage au sort est supérieur à *Pm*, le chromosome ne subit aucune modification.

Une probabilité de mutation $Pm = 1$ (soit 100%) entraîne une mutation de tous les chromosomes de la population.. Si elle vaut 0 (soit 0%), aucun changement n'intervient dans les chromosomes enfants issus de l'étape de croisement. La mutation empêche généralement l'algorithme génétique de tomber dans un optimum local. Cependant elle ne doit pas être utilisée fréquemment sinon la recherche devient complètement aléatoire. La Figure 5.6 illustre la mutation du chromosome enfant (**A'**) issu de l'étape de croisement précédent. La mutation du bit situé à la position $km = 8$ permet d'obtenir un nouveau chromosome (**A"**).

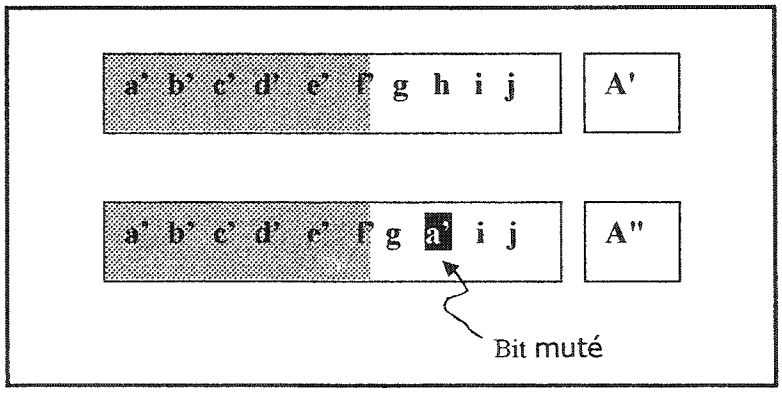


Figure 5.6 : Principe de la mutation d'un chromosome

On peut remarquer qu'aucun croisement entre **A** et **B** ne pouvait générer le chromosome **A''**. La mutation a donc permis de créer une nouvelle variété génétique. Elle apparaît comme une sorte d'exploration dans l'espace des solutions.

5.4.6 L'élitisme

L'élitisme est une procédure utilisée par les algorithmes génétiques pour préserver le meilleur chromosome dans chaque génération, contre une éventuelle altération de ses gènes, causée par le croisement ou la mutation. Il consiste donc à copier directement les gènes du meilleur chromosome de chaque génération, dans la génération suivante. Cette procédure peut accroître les performances de l'algorithme génétique puisqu'elle évite la perte du meilleur chromosome.

5.5 Synthèse de réseaux d'échangeurs de chaleur par les algorithmes génétiques

La synthèse d'un réseau d'échangeurs de chaleur optimal conduit à une recherche dans deux espaces différents : L'espace des structures (topologies) de REC et l'espace de la distribution des charges des échangeurs de chaleur. Un tel problème comporte des variables entières de décision (les unités d'échangeurs de chaleur) et des variables continues caractérisant les puissances échangées par les échangeurs de chaleur. La manipulation des structures de réseaux d'échangeurs de chaleur par les algorithmes génétiques requiert une représentation adaptée et compréhensible, favorisant l'application des opérateurs génétiques. La nature du problème exige donc deux codages différents : le codage de la structure du REC et le codage des charges des échangeurs de chaleur. La représentation des structures que nous allons utilisée est celle présentée dans les travaux de Lewin et al.(1998a) [4]. C'est une représentation compacte de la structure du REC avec une dimension constante, qui se prête aux manipulations par l'algorithme génétique. Elle peut également être mise sous une forme qui s'adapte à l'optimisation paramétrique.

5.5.1 Codage de la structure d'un REC

Pour le codage de la structure d'un REC, on utilise le concept de ***niveau*** (Voir la Figure 5.7). Chaque niveau consiste en un bloc d'échangeurs de chaleur agissant sur les courants. Au sein d'un niveau, chaque courant froid est autorisé à recevoir de la chaleur d'un seul courant chaud. Ainsi, pour une installation comportant 2 courants froids (C1 et C2) et 2 courants chauds (H1 et H2), le nombre maximal admissible d'échangeurs de chaleur dans le réseau sera égal à 6 si nous fixons le nombre de niveaux à 3. À la Figure 5.7, nous présentons la division en niveaux (3 niveaux) ainsi que les 6 positions d'appariements possibles des courants (représentées par des cercles). À chaque position d'appariement, l'on peut installer ou non un échangeur de chaleur.

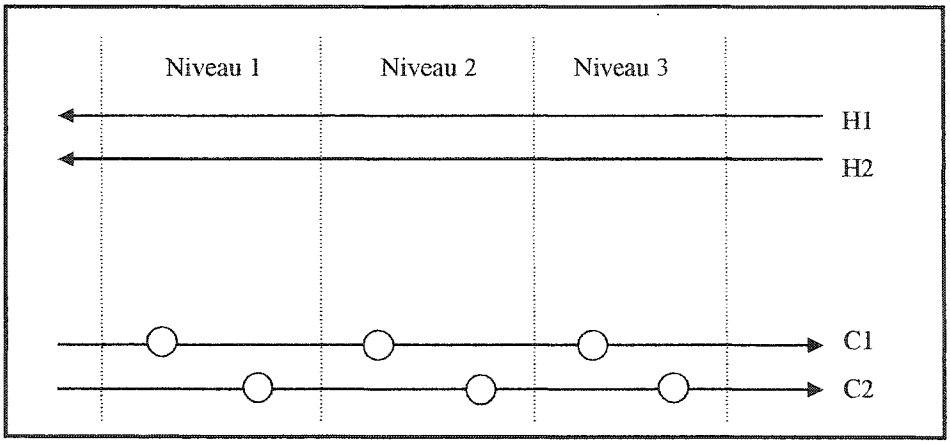


Figure 5.7 : Concept de niveau dans la représentation d'un REC

Le nombre maximal d'échangeurs de chaleur dans un REC peut être exprimé par la relation :

$$N_{EC} = N_v \times N_c \qquad [5.1]$$

où N_v est le nombre de niveaux, N_c est le nombre de courants froids et N_{EC} est le nombre maximal d'échangeurs de chaleur.

Pour la génération des REC par l'algorithme, nous utilisons un tableau appelé matrice d'incidence (Voir Tableau 5.1). Ce tableau comporte N_v lignes représentant le nombre de niveaux, et N_c colonnes représentant le nombre de courants froids. Chaque cellule de la matrice d'incidence, identifiée par le couple (N_v, N_c) contient un nombre entier ***b*** représentant le numéro d'un courant chaud.Une valeur de ***b*** non nulle dans la cellule (N_v, N_c) indique que le courant froid N_c reçoit de la chaleur du courant chaud ***b*** au niveau N_v. Si $b = 0$, alors le courant froid N_c ne reçoit pas de la chaleur au niveau N_c. Nous présentons au Tableau 5.1 une matrice d'incidence d'un REC généré par l'algorithme. Les valeurs ***b*** du tableau sont lues de gauche à droite et de haut en bas. La structure équivalente du REC est :

- Au niveau 1 : Le courant froid 1 (C1) reçoit de la chaleur du courant chaud 1 (H1). Le courant froid 2 (C2) ne reçoit de la chaleur d'aucun courant chaud.
- Au niveau 2 : Le courant froid 1 (C1) reçoit de la chaleur du courant chaud 2 (H2). Le courant froid 2 (C2) reçoit de la chaleur du courant chaud 2 (H2).
- Au niveau 3 : Le courant froid 1 (C1) reçoit de la chaleur du courant chaud 1 (H1). Le courant froid 2 (C2) reçoit de la chaleur du courant chaud 1 (H1).

Tableau 5.1 : Matrice d'incidence

		Courants froids	
		1	2
Niveau	1	1	0
	2	2	2
	3	1	1

Le codage de la matrice d'incidence est une représentation explicite et compréhensible du REC. Cependant les algorithmes génétiques requièrent d'habitude une représentation unidimensionnelle sous forme de chaîne de caractères représentant un chromosome. Ainsi, la matrice d'incidence bidimensionnelle (N_v, N_c) est transformée en une série de

N_v x N_c entiers représentant les gènes du chromosome. Cette forme est plus convenable pour la manipulation par les opérateurs de l'algorithme génétique. Le chromosome découlant de la matrice d'incidence serait ainsi représenté par la chaîne de nombres entiers **[1 0 2 2 1 1]**. Ces nombres sont lus à partir du Tableau 5.1, de la gauche vers la droite et de haut en bas. La structure du réseau d'échangeurs de chaleur équivalente est présentée à la Figure 5.8.

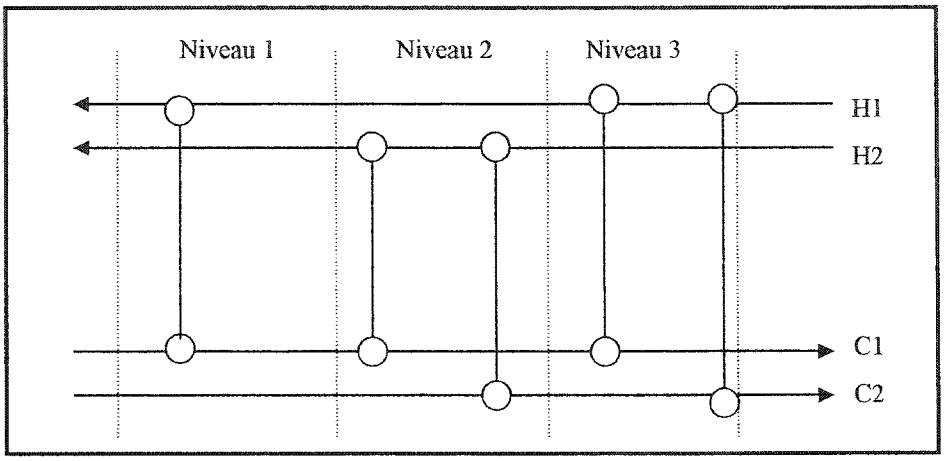


Figure 5.8 : Structure d'un réseau d'échangeurs de chaleur

La construction du REC à partir d'un chromosome est également possible si l'on connaît le nombre de niveaux N_v ainsi que le nombre de courant froid (N_c) du problème. Ceci permet d'établir la matrice d'incidence à partir de laquelle le réseau peut être construit.

Pour concevoir un REC par les algorithmes génétiques, il est nécessaire de spécifier la taille du chromosome lors du codage, à travers le choix du nombre de niveaux. Le choix de cette taille doit tenir compte de la règle des N-1 qui définit le nombre minimal d'unités permettant une récupération maximale de chaleur. Ainsi, le nombre de niveaux sera choisi tel que :

$$N_v \geq \frac{U_{\min} - N_U}{N_c} \qquad [5.2]$$

où N_v est un nombre entier, N_U est le nombre de types d'utilités, U_{min} est le nombre minimum d'unités (échangeurs de chaleur + utilités), N_c est le nombre de courants froids.

5.5.2 Espace de recherche des structures de REC

Optimiser la structure d'un REC pour la récupération de chaleur nous amène à rechercher une structure qui est meilleure parmi plusieurs autres structures alternatives. Les techniques utilisées pour le codage des REC définissent un nombre maximum de positions d'appariement qui est fonction du nombre de niveaux et de courants froids. Par exemple, pour un système comportant deux courants froids et deux courants chauds, le chromosome de la structure du REC comportera six positions d'appariement si nous fixons le nombre de niveaux N_v =3. S'il n'y a pas de restrictions particulières pour un appariement donné, nous sommes en situation d'équiprobabilité et à chaque position d'appariement, les conditions suivantes sont possibles : 0, 1, 2.

0 : Aucun appariement entre un courant froid et un courant chaud.
1 : Appariement entre un courant froid et le courant chaud 1.
2 : Appariement entre un courant froid et le courant chaud 2.

Chaque position d'appariement peut donc prendre 3 valeurs distinctes. Ceci implique pour tout le chromosome, 3^6 combinaisons possibles y compris bien sûr la combinaison [0 0 0 0 0 0]. Le nombre de structures distinctes possibles (N_{REC}) peut être établi par la relation suivante :

$$N_{REC} = (N_h + 1)^{N_{EC}} \qquad [5.3]$$

où N_h est le nombre de courants chauds, N_{REC} est le nombre de structures possibles, N_{EC} est la taille du chromosome (nombre maximal d'échangeurs dans un REC).

5.5.3 Application de l'opérateur de croisement sur les REC

L'exploitation de l'espace des solutions implique d'opérer des croisements des chromosomes de la populations de REC. Le principe du croisement de deux chromosomes de REC est démontré à la Figure 5.9. La position de croisement est fixée à $kc = 3$, puis les deux portions de structures à gauche du point de croisement sont échangées afin de créer deux nouveaux chromosomes.

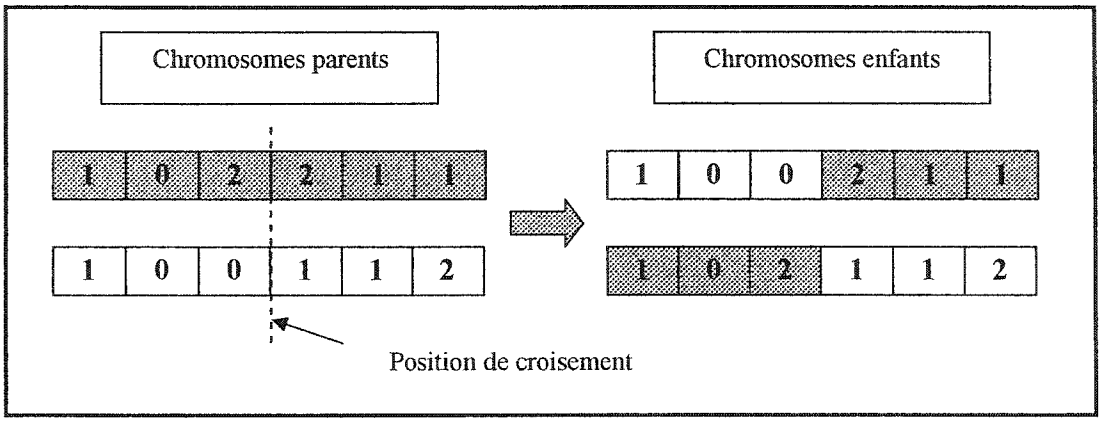


Figure 5.9 : Croisement de deux chromosomes de REC

Le croisement correspondant des structures des REC sont illustrées à la Figure 5.10.

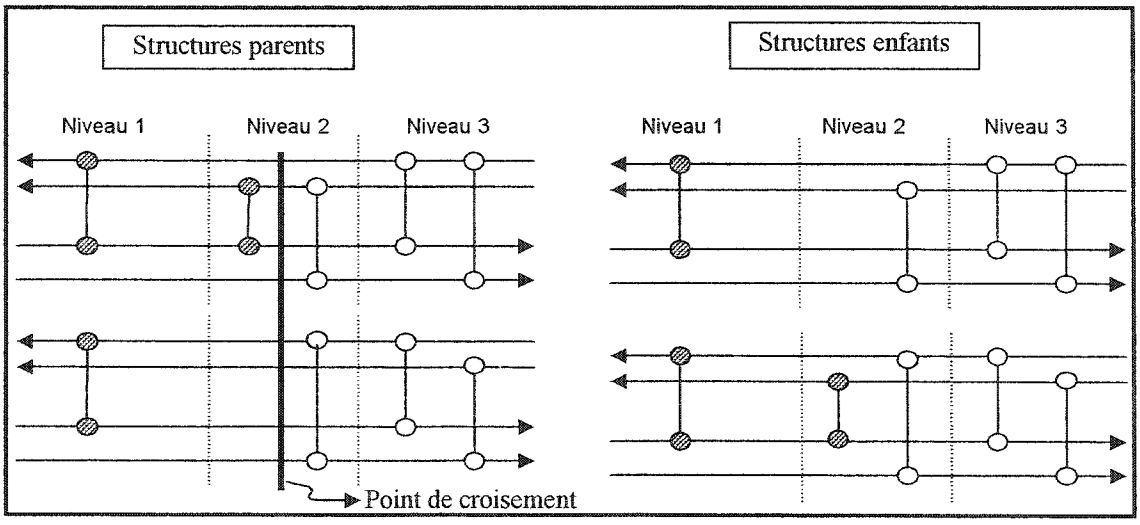


Figure 5.10 : Croisement de deux structures de REC

5.5.4 Application de l'opérateur de mutation sur les REC

La mutation dans notre cas est le changement aléatoire d'un appariement entre courant chaud et froid. Une position d'appariement *km* du chromosome à muter est sélectionnée de façon arbitraire (exemple : la position *km* = 5 à la Figure 5.11), puis la valeur b_i de cette position est mutée pour prendre une nouvelle valeur telle que :

$$0 \le b_i \le (N_h + 1) \qquad [5.4]$$

où b_i est le numéro du courant chaud à la position i, N_h est le nombre de courants chauds.

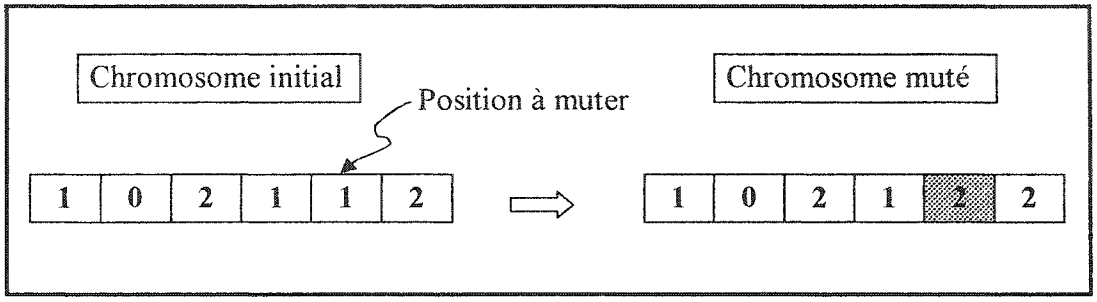


Figure 5.11 : Mutation d'un chromosome de REC

L'appariement initial du courant froid 1 et du courant chaud 1 (position 5) est muté donnant ainsi lieu à un appariement entre le courant froid 1 et le courant chaud 2 (figure 5.12).

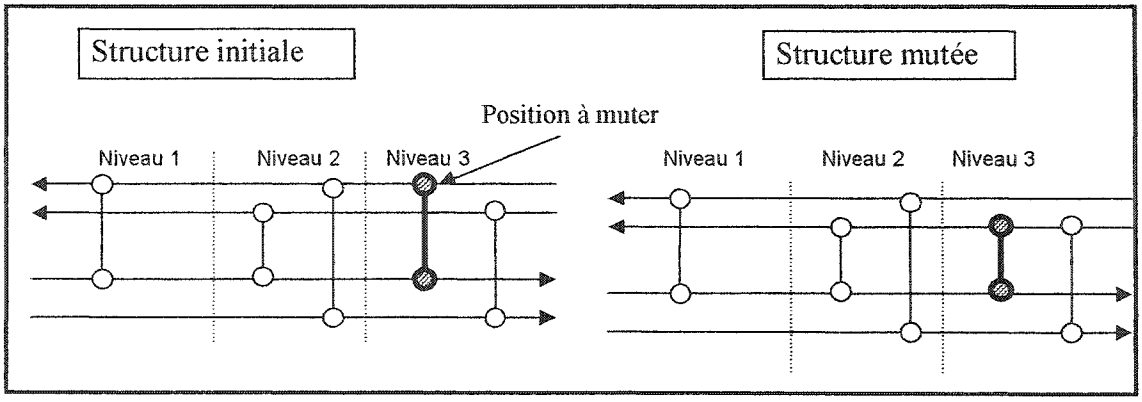


Figure 5.12 : Mutation de la structure de REC

5.5.5 Codage de la distribution de charge d'un REC

La charge thermique de chaque unité d'échangeur de chaleur dans le réseau sera codée par une séquence de nombres binaires, c'est-à-dire par des 0 et 1, d'une longueur finie n_b. Ainsi, pour le réseau comportant un nombre maximal de N_{EC} échangeurs de chaleur, le nombre total de bits nécessaire(λ) pour coder le chromosome sera :

$$\lambda = n_b \times N_{EC} \qquad [5.5]$$

Par exemple : Si nous choisissons de coder la charge thermique d'un échangeur de chaleur avec 9 bits, alors la longueur totale d'un chromosome de distribution de charge d'un REC comportant 6 échangeurs de chaleur sera égale à λ = 6x9 = 54 bits (figure 5.13).

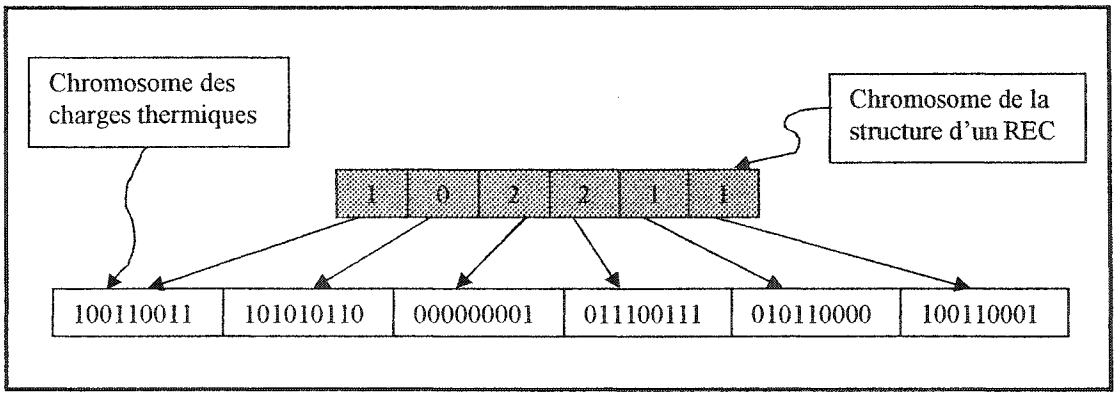


Figure 5.13 : Représentation d'un chromosome de distribution de charge

En fonction de la précision recherchée, chaque variable peut être codée avec un nombre différent de bits. La précision des résultats de l'algorithme de recherche dépend du nombre de bits utilisés pour coder les variables. La précision (δ) requise pour chaque variable X_k définie dans le domaine $[X_{inf}, X_{sup}]$ est :

$$\delta = \frac{X_{\sup} - X_{\inf}}{2^{n_b} - 1} \qquad [5.6]$$

où X_{inf} et X_{sup} désignent respectivement les bornes inférieure et supérieure de la variable X_k et n_b est le nombre de bits utilisés pour coder la variable.

Le codage binaire du chromosome définit un espace de recherche d'une taille qui est fonction de la précision δ recherchée pour les variables. Si nous admettons que toutes les variables requièrent la même précision, alors la longueur totale λ du chromosome de distribution de charge doit être:

$$\lambda \geq 1.44 \times \ln\left(1 + \frac{X_{\sup} - X_{\inf}}{\delta}\right) \times N_{EC} \qquad [5.7]$$

Par exemple, pour un chromosome de 6 gènes dont les variables correspondantes sont définies dans le domaine $[0, 32]$, chaque gène doit avoir au moins une longueur n_b = 8 bits si la précision requise est de 10^{-2}. La taille totale du chromosome sera ainsi égale à λ = (8x6) = 48 bits. Ceci, en retour, génère un espace de recherche d'à peu près $3x10^{14}$ solutions. Donc, le chromosome de distribution optimale des charges thermiques doit être cherché parmi une population de $3x10^{14}$ chromosomes.

5.5.6 Décodage du chromosome de distribution de charge

Pour évaluer la performance (ou le fitness) d'un chromosome, celui-ci doit d'abord être décodé, et ensuite envoyé au modèle de simulation. Le modèle retournera une valeur indiquant le fitness. Chaque variable étant définie par ses bornes inférieure et supérieure, la séquence de bits de longueur n_b peut être représentée par $\left(2^{n_b} - 1\right)$ intervalles. Une séquence de bits constituée uniquement de zéros (0) représente zéros intervalles; ainsi la valeur X_k du gène sera égale à la borne inférieure. Si la séquence de bits est constituée uniquement de 1, cela représente $\left(2^{n_b} - 1\right)$ intervalles et donc la valeur du gène sera égale à la borne supérieure.

La conversion de la chaîne de bits $[b_{n-1}, b_{n-2}, ..., b_0]$ en un nombre réel X_k dans le domaine $[X_{\inf}, X_{\sup}]$ se fait selon les deux étapes suivantes :

1 – Conversion du codage binaire en base 10 telle que :

$$X'_k = \left(\sum_{i=0}^{n_b-1} b_i \times 2^i \right) \qquad [5.8]$$

2 – Calcul de la valeur réelle de X_k correspondant en utilisant la relation suivante :

$$X_k = X_{\inf} + \frac{X_{\sup} - X_{\inf}}{2^n - 1} X'_k \qquad [5.9]$$

5.5.7 Choix de la taille de la population

En pratique, le nombre de chromosomes constituant la population dans une génération ne sera pas pris égal au nombre de structures distinctes possibles. Une population échantillon, de taille constante, sera prise à partir de l'espace de recherche et servira au lancement de l'algorithme. Le choix de la taille de la population est fonction de la complexité du problème. En général, s'il n'y a pas assez de chromosomes dans la population, il y aura peu de diversités génétiques et dans ce cas, seule une petite partie de l'espace de recherche est explorée. Par contre, si le nombre de chromosomes est élevé, une grande partie de l'espace de recherche est explorée mais au prix d'un temps de calcul plus long.

5.5.8 Modèle mathématique de simulation de la récupération de chaleur

Le problème commun à tout calcul d'optimisation est la définition d'une fonction objectif qui traduit de façon adéquate les objectifs que l'on veut atteindre. Dans le cas particulier de notre étude, l'objectif principal est la conception d'un réseau d'échangeurs de chaleur qui assure une récupération de la quantité maximale de chaleur. Dans le chapitre 4 (Paragraphe 4.3.4) nous avons vu que la maximisation de la puissance

récupérée à travers des échangeurs de chaleur conduit à une minimisation des utilités utilisées. Pour des questions de simplicité d'élaboration de l'algorithme d'optimisation, nous choisissons d'établir un modèle qui minimise les besoins en utilités chaudes et froides.

Pour établir le modèle de simulation, nous nous servirons de l'exemple de la structure générée par l'algorithme (voir la Figure 5.8). Cette structure doit être complétée en ajoutant systématiquement une utilité sur chaque courant. Nous désignons ces utilités par le terme QU_j, l'indice j est le numéro du courant sur lequel est installée l'utilité. Elles sont représentées par les cercles hachurés (Figure 5.14). Une utilité ou un échangeur de chaleur est éliminé du réseau final si le programme d'optimisation paramétrique lui affecte une charge thermique de valeur nulle.

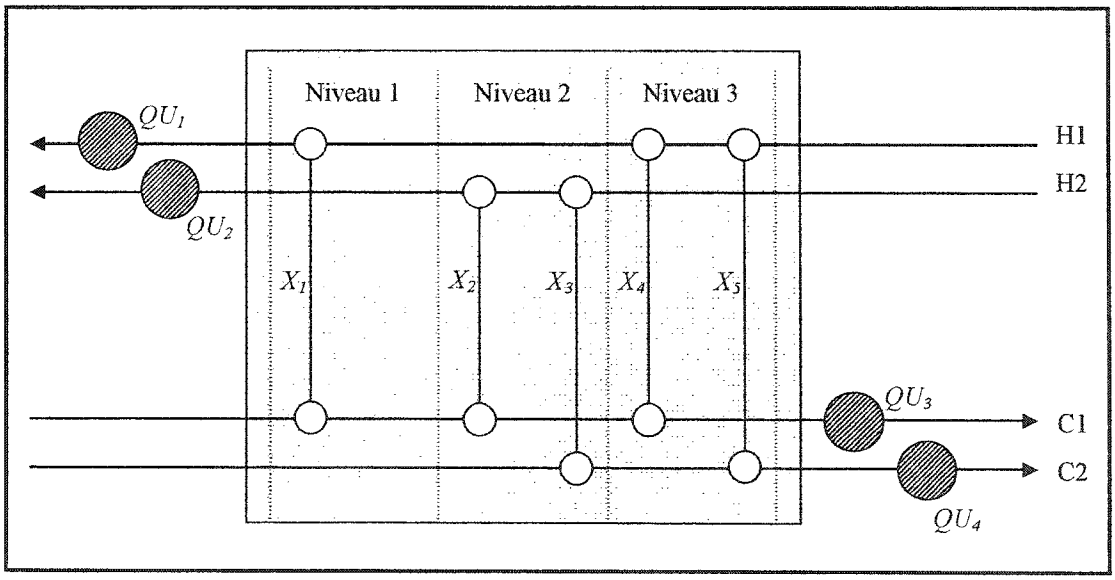


Figure 5.14 : Structure complète d'un REC

Les différentes notations qui serviront à l'établissement des équations et à la modélisation du réseau d'échangeurs de chaleur sont présentées à la Figure 5.15.

Pour toute structure de REC générée par l'algorithme, l'étape d'optimisation des charges consistera à minimiser la puissance des utilités, en maximisant la récupération de chaleur à travers les échangeurs de chaleur. La fonction objectif (ou fonction de fitness) est traduite par la relation :

$$f = \sum_{j=1}^{N_{st}} QU_j \qquad [5.10]$$

où N_{st} est le nombre total d'utilités, équivalent au nombre de courants du système (puisqu'on suppose à priori qu'il y a une utilité sur chaque courant).

La puissance minimale requise des utilités dans une installation est définie pour une température minimale d'approche spécifiée, en traçant le diagramme des courbes composites des courants (chapitre 4). Le réseau d'échangeurs de chaleur construit doit donc amener les courants à leurs températures finales désirables, tout en assurant que la différence de température dans un échangeur de chaleur ne soit pas inférieure à la température minimale d'approche fixée.

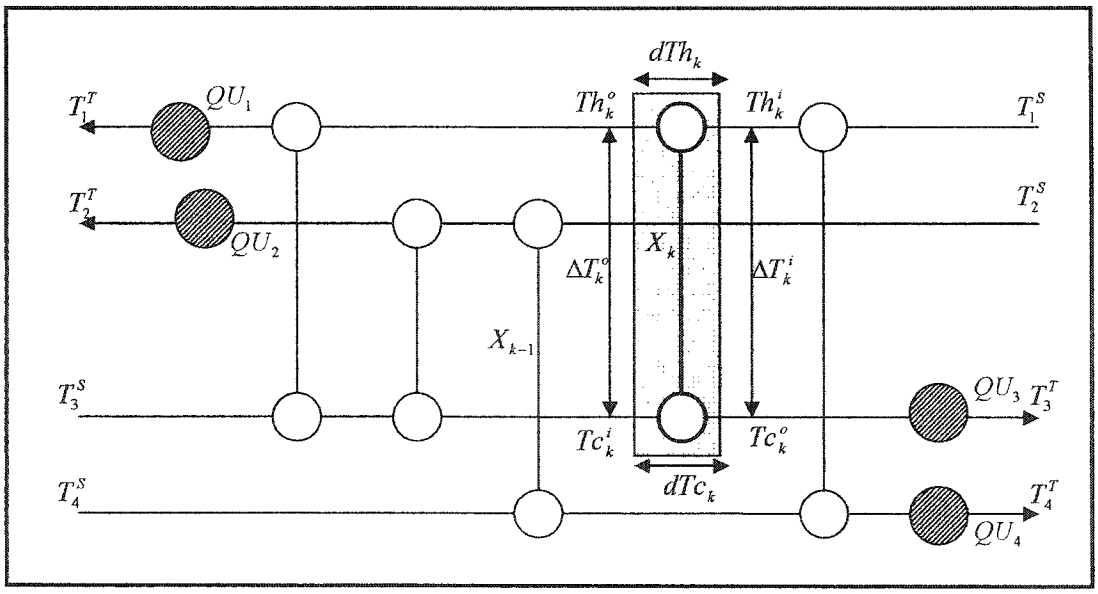


Figure 5.15 : Les paramètres du REC

Le problème à résoudre comporte donc deux types de contraintes : Il s'agit d'une part, des contraintes dues aux limitations sur la valeur possible que peut prendre la différence de température entre un courant chaud et un courant froid dans un échangeur de chaleur, et d'autre part, des contraintes liées à la satisfaction du bilan d'énergie de chaque courant.

Le bilan d'énergie pour chaque courant peut être exprimé par la relation suivante :

$$\sum_{k=1}^{N_{x,j}} (X_k)_j + QU_j = CP_j \times \left| T_j^S - T_j^T \right| \qquad [5.11]$$

où CP_j est la capacité calorifique du courant j , T_j^S et T_j^T désignent respectivement les températures initiales et finales du courant j, $N_{x,j}$ désigne le nombre d'échangeurs de chaleur connectés au courant j, $(X_k)_j$ désigne la puissance de l'échangeur de chaleur k connecté au courant j, QU_j désigne la puissance de l'utilité installée sur le courant j.

La différence de température entre les courants s'écoulant dans un échangeur de chaleur doit être en tout point, supérieure ou égale à la température minimale d'approche ΔT_{min}. Une paire de contraintes inégalité caractérisera donc chaque échangeur de chaleur à travers les relations suivantes :

$$\begin{cases} \Delta T_k^o = Th_k^o - Tc_k^i \geq \Delta T_{\min} \\ \Delta T_k^i = Th_k^i - Tc_k^o \geq \Delta T_{\min} \end{cases} \qquad [5.12]$$

où ΔT_k^i et ΔT_k^o désignent respectivement les différences de température entre les courants aux extrémités chaude et froide de l'échangeur de chaleur. Les termes Th_k^o et Th_k^i désignent respectivement la température du courant chaud sortant de l'échangeur de chaleur et la température du courant chaud entrant dans l'échangeur de chaleur. Tc_k^o et Tc_k^i désignent respectivement la température du courant froid sortant de l'échangeur de chaleur et la température du courant froid entrant dans l'échangeur de chaleur.

Les températures Th_k^i et Th_k^o désignent respectivement les températures d'un courant chaud à l'entrée et à la sortie de l'échangeur de chaleur k.

$$Th_k^o = T_j^T + \frac{QU_j}{CP_j} + \sum_{i=1}^{k-1} \varepsilon \, dTh_i \qquad [5.13]$$

$$Th_k^i = Th_k^o + dTh_k \qquad [5.14]$$

où dTh_k est La variation de la température d'un courant chaud à travers l'échangeur de chaleur k. elle est exprimée par la relation :

$$dTh_k = \frac{X_k}{CP_j} \qquad [5.15]$$

$\varepsilon = 1$ si l'échangeur k est connecté sur le courant j et $\varepsilon = 0$ dans le cas contraire.

Les températures Tc_k^i et Tc_k^o désignent respectivement les températures d'un courant froid à l'entrée et à la sortie de l'échangeur k.

$$Tc_k^i = T_j^S + \sum_{i=1}^{k-1} \varepsilon \, dTc_i \qquad [5.16]$$

$$Tc_k^o = Tc_k^i + dTc_k \qquad [5.17]$$

où dTc_k est la variation de la température d'un courant froid à travers l'échangeur de chaleur k. Elle est exprimée par la relation :

$$dTc_k = \frac{X_k}{CP_j} \qquad [5.18]$$

$\varepsilon = 1$ si l'échangeur k est connecté sur le courant j et $\varepsilon = 0$ dans le cas contraire.

Il faut mentionner que le terme extrémité chaude désigne le côté de l'échangeur de chaleur par lequel entre le courant chaud et l'extrémité froide désigne le côté par lequel sort le courant chaud.

Le modèle mathématique que l'on peut établir pour la minimisation de la puissance des utilités est un programme linéaire qui est formulé selon l'expression suivante:

Trouver les valeurs de X_1, X_2, X_3, X_4, X_5, QU_1, QU_2, QU_3, QU_4, *qui*

$$\textit{Minimisent } f = \sum_{j=1}^{N_{st}} QU_j \qquad [5.19]$$

liée aux contraintes

$$\sum_{i=1}^{N_{s,j}} (X_i)_j + QU_j = CP_j \times \left| T_j^S - T_j^T \right| \qquad [5.20]$$

$$\left.\begin{aligned} Th_k^i - Tc_k^o &\geq \Delta T_{\min} \\ Th_k^o - Tc_k^1 &\geq \Delta T_{\min} \end{aligned}\right\} \qquad [5.21]$$

où $j = 1,\ldots,N_{st}$, $k = 1,\ldots,N_{EC}$

$X_k \geq 0$, $QU_j \geq 0$

L'équation [5.19] est la fonction objectif du problème d'optimisation. L'équation [5.20] représente les contraintes égalité et exprime le bilan d'énergie pour chaque courant. Quatre contraintes égalité caractériseront donc le réseau présenté à la Figure 5.15. La paire de contraintes inégalité (équation [5.21]) représente les contraintes de température pour chaque échangeur de chaleur. Ainsi, dix contraintes inégalité caractériseront le réseau présenté à la Figure 5.15

5.5.9 Méthode de la fonction pénalité

Le modèle de programmation linéaire ainsi défini pour la simulation de la récupération de chaleur comporte une fonction objectif et un ensemble de contraintes (soit 14 contraintes dans notre cas). Sous cette forme , le modèle ne peut pas être supporté directement par les AG du fait des contraintes qui sont associées à la fonction objectif. Pour permettre aux AG de traiter ce problème comportant des contraintes, le modèle doit être mis sous une forme compatible avec un traitement par les algorithmes génétiques.

Différentes techniques ont été proposées pour permettre aux AG de traiter les problèmes d'optimisation avec contraintes. Une description de ces techniques est présentée dans l'ouvrage de Mitsuo et Cheng, (1997) [38]. Il s'agit notamment des méthodes du rejet, de réparation, de modification des opérateurs génétiques et de pénalité. Dans notre approche pour la synthèse des REC, nous allons utiliser la méthode de pénalité pour transformer le problème avec contraintes en un problème sans contraintes.

Nous avons établi dans l'équation 5.11, le bilan d'énergie pour chaque courant j par l'expression : $\sum_{k=1}^{N_{x,j}} (X_k)_j + QU_j = CP_j \times \left| T_j^S - T_j^T \right|$, où le second membre de cette équation traduisant la charge totale du courant est une constante. Il faut noter que les variables X_k (charges thermiques des échangeurs de chaleur) sont générées sous forme de séquence de nombres binaires et de façon arbitraire par l'algorithme. La valeur réelle de la variable X_k est déterminée après le décodage à partir de l'équation 5.9 ($X_k = X_{\inf} + \frac{X_{\sup} - X_{\inf}}{2^n - 1} X'_k$). Il s'ensuit que la minimisation du terme QU_j (équation 5.11) ne sera pas toujours facile, d'autant plus que la limite supérieure de la variable X_k, n'est pas clairement définie.

La détermination de la limite inférieure (X_{inf}) de la variable ne pose pas de problème puisque celle-ci peut être mise à zéro, signifiant qu'il n'y a pas d'échange de chaleur:

$$X_{\inf} = 0 \qquad [5.22]$$

Sur la figure 5.16, nous pouvons voir que l'échangeur de chaleur k partage avec deux autres échangeurs, la charge thermique du courant 1. En revanche, nous ne pouvons pas connaître dans quelles proportions la distribution de charge est faite entre ces trois échangeurs.

Considérons dans un premier temps que nous fixons la valeur maximale prise par la variable X_k telle que :

$$X_{\sup} = \min(Q_1, Q_3), \qquad [5.23]$$

c'est-à-dire la charge minimale parmi celles des deux courants sur lesquels est connecté l'échangeur de chaleur considéré. $Q_1 = CP_1\left(T_1^S - T_1^T\right)$ est la charge du courant 1 (courant chaud) et $Q_3 = CP_3\left(T_3^T - T_3^S\right)$ est la charge du courant 3 (courant froid). En appliquant le même principe aux autres échangeurs de chaleur, le terme $\sum_{k=1}^{N_{s,j}}\left(X_k\right)_j$ de l'équation 5.11 peut excéder le second membre de l'équation, rendant le terme QU_j négatif, ce qui est infaisable. L'algorithme passera alors le temps à évaluer des chromosomes infaisables, ralentissant par conséquent la convergence vers la solution optimale.

Si nous adoptons pour une distribution équitable de la charge ($\min[Q_1, Q_3]$) entre tous les échangeurs de chaleur installés sur le même courant chaud, la limite supérieure de X_k sera alors :

$$X_{\sup} = \frac{1}{N_{EC\min}}\left[\min(Q_1, Q_3)\right] \qquad [5.24]$$

où N_{ECmin} est le nombre d'échangeurs de chaleur sur le courant de charge minimale auquel est connecté l'échangeur de chaleur (k) considéré.

Dans ce cas, le terme , $\sum_{k=1}^{N_{s,j}}\left(X_k\right)_j$, de l'équation 5.11 risque de ne pas être maximisé pour chaque courant, et il faudra alors utiliser des utilités de puissances élevées pour amener les courants à leurs températures finales, ce qui est contraire à l'objectif d'optimisation que l'on s'est fixé. L'algorithme passera la majorité du temps à manipuler des chromosomes qui ne maximisent pas la puissance récupérée.

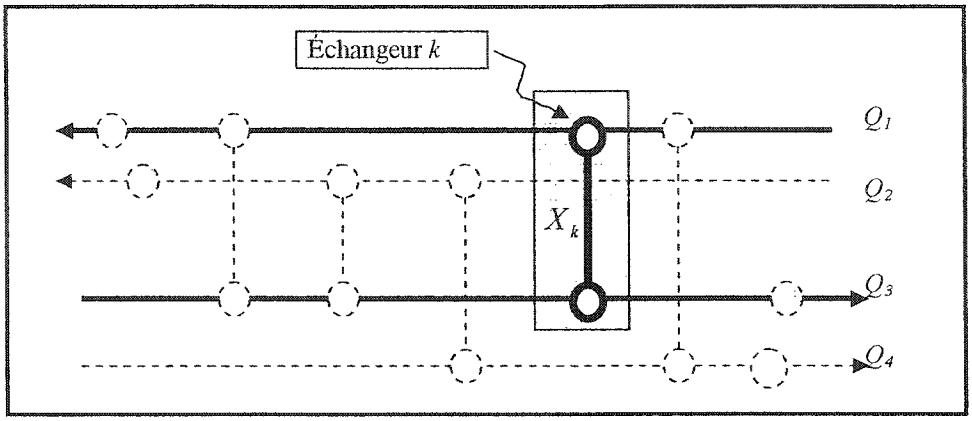


Figure 5.16 : Charge maximale d'un échangeur de chaleur du REC

Pour trouver un compromis entre les deux situations précédemment évoquées, nous allons introduire un coefficient ***r***, que nous appelons *coefficient de réparation*, tel que, pour tout échangeur de chaleur k installé entre un courant chaud de charge thermique Q_h et un courant froid de charge Q_c, la borne supérieure de la variable X_k associée soit:

$$X_{\text{sup}} = \frac{1}{N_{EC\,\min}}\left[\min\left(Q_h, Q_c\right)\right] \pm r \qquad [5.25]$$

Ainsi, l'algorithme ajustera le coefficient r pour satisfaire le bilan d'énergie (équation 5.11), et minimiser en même temps le terme QU_j. Le mécanisme de correction des bornes supérieures des variables X_k permet de satisfaire le bilan d'énergie des courants. Toutes les contraintes égalité du modèle de programmation linéaire sont alors satisfaites. Il ne reste plus que les contraintes inégalités qui seront prises en charge dans la méthode de la fonction pénalité.

L'équation 5.9 utilisée pour décoder les variables peut alors être réécrite sous la forme suivante, en tenant compte des expressions de la borne inférieure (équation 5.22) et de la borne supérieure (équation 5.25) :

$$X_k = \frac{1}{2^n - 1}\left(\frac{1}{N_{EC\min}}\left[\min(Q_h, Q_c)\right] \pm r\right) X'_k \qquad [5.26]$$

La satisfaction du bilan d'énergie de chaque courant n'implique pas que les contraintes de température soient respectées au niveau de chaque échangeur de chaleur. La méthode de la ***fonction pénalité*** est utilisée pour maintenir un certain nombre de solutions infaisables dans chaque génération de telle manière à amener l'algorithme génétique à chercher la solution optimale aussi bien dans la région des solutions faisables que dans la région des solutions infaisables. Ainsi les solutions infaisables ne sont pas systématiquement rejetées à chaque génération, parce que certaines d'elles peuvent contenir des informations très utiles pour la recherche de l'optimum. L'illustration donnée à la Figure 5.17 montre comment des solutions infaisables sont souvent proches de l'optimale recherchée que celles qui sont faisables. Par exemple, la solution infaisable (e) est plus proche de la solution optimale recherchée (a) que la solution faisable (f) (Voir la Figure 5.17).

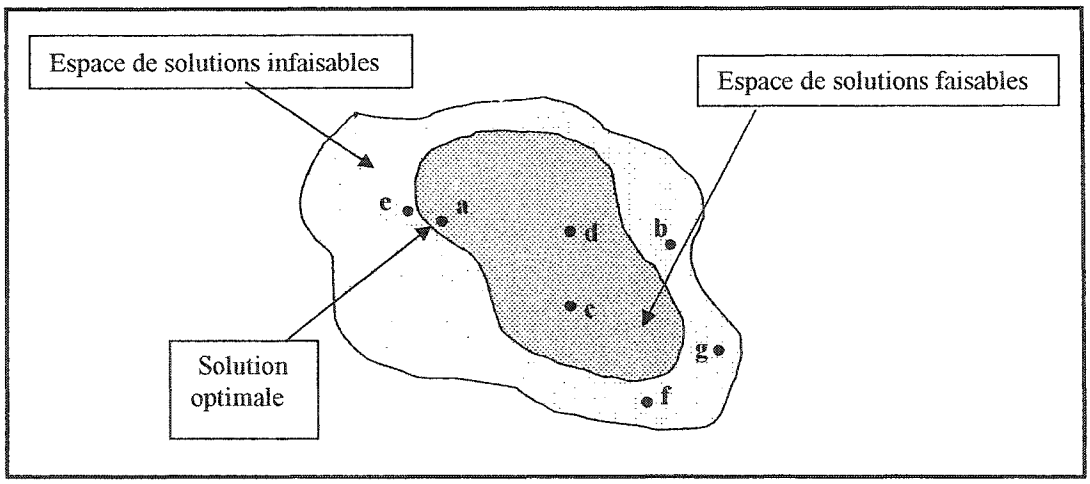


Figure 5.17 : Espaces des solutions faisables et infaisables

Pour permettre à l'algorithme génétique de traiter un problème comportant des contraintes, le problème est converti en un problème sans contraintes par l'association d'une pénalité à la fonction d'évaluation, pour toute violation des contraintes (Yogesh, 1998)

[39]. En d'autres termes, les AG génèrent les solutions potentielles sans considérer les contraintes et ensuite les pénalisent en altérant la qualité de la fonction d'évaluation. Cette approche combine la fonction objectif et les contraintes en une seule fonction qui est traitée comme un problème sans contrainte.

Le modèle résultant est une **fonction composite** F combinant la **fonction objectif** f (équation 5.10) et une **pénalité** P, pour toute violation des contraintes. Cependant, il n'y a pas une méthodologie fixe pour la combinaison de la fonction objectif avec la pénalité (Yogesh, 1998) [39]. Dans le modèle que nous élaborons, nous avons opté pour une simple addition des deux termes telle que la fonction composite résultante est:

$$F = f + P \qquad [5.27]$$

$$\begin{cases} P = 0 & s'il\ n'y\ a\ aucune\ violation\ des\ contraintes \\ P > 0 & si\ les\ contraintes\ sont\ violées \end{cases} \qquad [5.28]$$

Il y a une variété de fonctions pénalités qui peut être utilisée (logarithmique, linéaire, quadratique, constante, etc.). Le problème qui se pose à ce niveau est de savoir quelle forme de pénalisation adoptée pour assurer un équilibre entre la préservation d'un patrimoine génétique (conservation de certaines solutions infaisables) et la pression de sélection (rejet de certaines solutions infaisables) de manière à guider la recherche génétique vers les régions prometteuses de l'espace de solutions. Nous utiliserons dans notre système une fonction pénalité définie telle que:

$$P = Ck \times \sum_{i=1}^{N_{inéq}} \Phi_i \qquad [5.29]$$

$$\begin{cases} \Phi_i = 0 & si \quad \Delta T_i \geq \Delta T_{\min} \\ \Phi_i = 1 - \dfrac{\Delta T_i}{\Delta T_{\min}} & si \quad \Delta T_i < \Delta T_{\min} \end{cases} \qquad [5.30]$$

Ck est un coefficient de pénalité, et Φ_i est la pénalité relative à la i-ième contrainte. $N_{\text{inéq}}$ est le nombre total des contraintes inégalités (contraintes de température). Il faut se

rappeler que les contraintes égalité (conservation d'énergie) ont été satisfaites dans la procédure de réparation.

Le modèle de programmation linéaire précédemment défini (équations 5.19, 5.20 et 5.21) se résume par l'expression suivante :

$$Minimiser \quad F = \sum_{j=1}^{N_{st}} QU_j + Ck \sum_{i=1}^{N_{inéq}} \Phi_i \qquad [5.31]$$

Le modèle mis ainsi sous cette forme est mieux adapté pour être traité par les algorithmes génétiques.

Le coefficient de pénalité *Ck* est un paramètre qui varie le niveau de la pénalisation ***P*** par rapport à la fonction ***f***. Si le coefficient de pénalité *Ck* est pris égal à zéro, les chromosomes ne subissent aucune pénalisation pour toute violation des contraintes, et par conséquent, les contraintes ne sont pas satisfaites. Si la valeur de *Ck* est prise assez large, la moindre violation des contraintes est fortement pénalisée, ce qui entraîne la formation d'une population de chromosomes qui satisfont les contraintes. Cependant, si l'espace de solutions est tel que la production d'individus violant les contraintes est probable, on court le risque de créer une population peu diversifiée qui peut conduire à un optimum local. En adoptant un coefficient de pénalité modéré, l'algorithme peut admettre des individus qui violent les contraintes mais qui sont cependant plus acceptables que certains individus qui ne violent pas ces contraintes. En variant *Ck* nous pouvons varier la vitesse de convergence et l'effet des différentes contraintes sur la solution.

5.5.10 Processus de sélection pour la reproduction

Les algorithmes génétiques manipulent une population de solutions et non une solution unique. À chaque génération, les chromosomes des meilleures solutions doivent être sélectionnés pour la reproduction. Nous constituons ainsi une population intermédiaire ou bassin de reproduction (***'mating pool'***) qui contiendra les chromosomes devant

participer à la production de la génération future. Pour la constitution de ce bassin de reproduction, nous utilisons dans notre algorithme la méthode de la roue biaisée pour laquelle chaque chromosome est sélectionné en fonction de son "fitness". Pratiquement cette procédure se présente comme suit :

1- Évaluer le "fitness" de chaque chromosome $F_j(X_1, X_2, ..., X_k)$, avec j = 1,..., pop (pop est la taille de la population totale).

2- Calculer la somme des "fitness" de toute la population : $som = \sum F_j$ [5.32]

3- Calculer la moyenne des "fitness" de la population : $moy = \frac{som}{pop}$ [5.33]

4- Calculer le taux de sélection de chaque chromosome :

Si le problème d'optimisation est une minimisation, alors : $\tau_j = \frac{moy}{F_j}$ [5.34]

Dans le cas d'une maximisation : $\tau_j = \frac{F_j}{moy}$ [5.35]

La valeur du taux de sélection détermine combien de fois le chromosome sera représenté dans le bassin de reproduction. Le remplissage du bassin de reproduction se fait à partir du meilleur chromosome vers le moins bon, tout en s'assurant de garder la taille de la population constante.

5.6 *Élaboration du programme d'optimisation*

L'algorithme de recherche que nous avons codifié en FORTRAN 90 comporte deux niveaux : Le premier niveau est consacré à la recherche de la structure optimale, tandis que le second niveau assure l'optimisation de la distribution des charges. Les programme sont présentés aux Annexes 1 et 2.

Nous organisons l'algorithme de manière analogue à une chaîne de triage (Figure 5.18)

où nous déterminons dans un premier temps la topologie optimale du REC pour la récupération de chaleur. Une fois la meilleure topologie identifiée, nous procédons à l'optimisation de la distribution des charges.

5.6.1 Recherche de la topologie du REC

L'algorithme commence avec une population initiale arbitraire de REC qui va évoluer vers la topologie optimale, et une population de distribution de charges qui elle ne subit pas de transformation au cours de la recherche de la topologie du REC. Chaque structure de REC positionnée au poste (A) (Figure 5.18), agit comme un tamis au travers duquel vont passer les différentes distributions de charge.

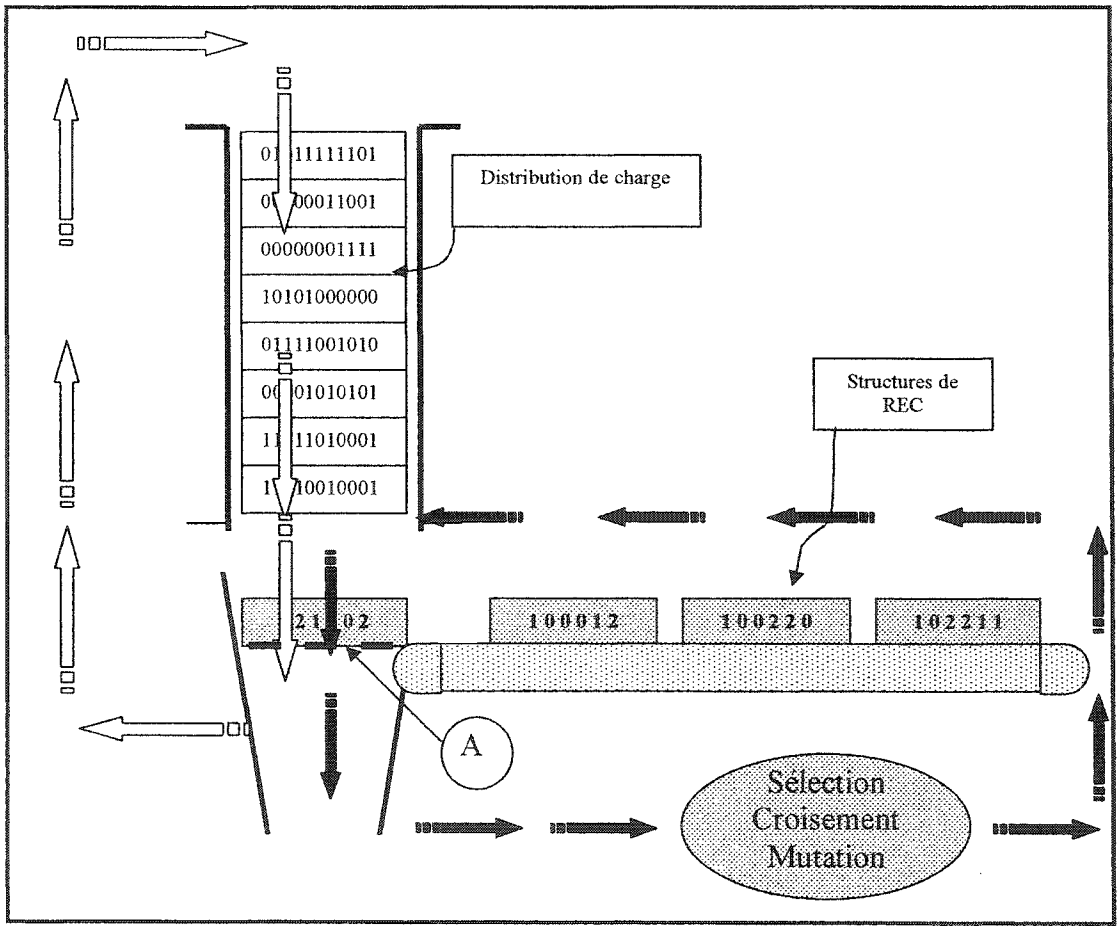


Figure 5.18 : Mécanisme de recherche de la topologie optimale du REC

Plus la topologie du REC est adaptée à une distribution de charges, plus grande sera la puissance maximale récupérée. Ainsi chaque chromosome de REC sera adapté à un chromosome de distribution de charges. Après l'évaluation de toutes les structures de REC par rapport aux distributions des charges, les meilleurs chromosomes de REC sont sélectionnés. Les chromosomes sélectionnés vont subir des opérations de croisement et de mutation pour donner naissance à une nouvelle population. Cette nouvelle population subit le même processus d'évaluation et l'opération se répète ainsi jusqu'à ce que les conditions d'arrêt soient satisfaites.

5.6.2 Optimisation de la distribution des charges

Après avoir retenu la meilleure structure de REC, la distribution des charges des échangeurs de chaleur peut être optimisée.

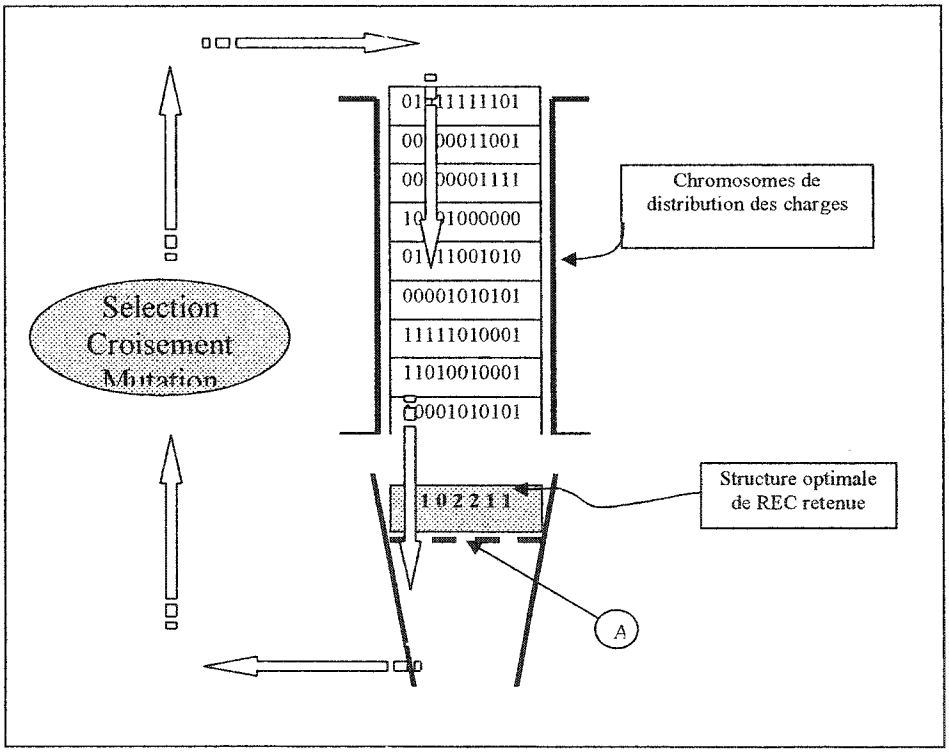


Figure 5.19 : Mécanisme de recherche de la distribution de charge du REC

Le chromosome de la structure optimale retenue est positionnée en (A), et agit comme le

tamis dont nous évaluons la performance par rapport aux chromosomes de distributions des charges. Une population initiale de distributions des charges est générée de façon aléatoire. Pour chacun de ces chromosomes, nous déterminons la puissance maximale récupérée. À chaque cycle, les meilleurs chromosomes de distribution de charge permettant une meilleure récupération de chaleur sont sélectionnés et à travers des opérations de croisement et de mutation, nous faisons évoluer ces distributions vers la solution optimale. Cette phase de l'optimisation de la distribution des charges thermiques est illustrée à la figure 5.19.

5.6.3 Conditions d'arrêt de l'algorithme de recherche

Plusieurs conditions peuvent provoquer l'arrêt du processus de recherche de l'optimum d'une solution. Le processus d'optimisation est arrêté si la solution optimale recherchée est atteinte, ou si le nombre maximal de générations fixé est atteint, ou encore si après un certain nombre de générations, il n'y a pas d'amélioration significative de la solution. La spécificité de l'optimisation des REC est que l'on connaît toujours la puissance maximale à récupérer puisqu'elle peut être déterminée à travers le diagramme des courbes composites ou par une méthode numérique que nous verrons au paragraphe suivant. Cette valeur peut donc servir comme condition d'arrêt si la solution recherchée s'approche ou atteint cette valeur.

5.6.4 Aperçu sur la méthode du 'Problem Table Algorithm'

La construction graphique du diagramme des courbes composites est un excellent moyen pour apprendre la méthode du pincement et comprendre la situation générale mais n'est pas commode dans les méthodes automatiques de synthèse des réseaux d'échangeurs de chaleur. Dans le programme que nous avons élaboré, nous déterminons les performances du systèmes par une procédure numérique appelée 'Problem Table Algorithm' (**PTA**) qui a été développée par Linnhoff et Flower, (1978) [40].

Avec le PTA, les températures d'entrée et de sortie de tous les courants du procédé divisent l'échelle de température en plusieurs intervalles, tout comme dans le cas de la construction des courbes composites. Pour les besoins de l'établissement du diagramme des intervalles de température, on procède à une modification des températures par soustraction de la quantité $\Delta T_{min}/2$ des courants chauds et par addition de la même quantité $\Delta T_{min}/2$ aux courants froids. En nous servant des données du Tableau 4.1 (Chapitre 4) et pour une température minimale d'approche ΔT_{min} = 10 °C, nous établissons le nouveau tableau modifié dont les résultats sont présentés au Tableau 5.2.

Tableau 5.2 : Températures modifiées du PTA

Courants	Température initiale modifiée T^S_{mod} (°C)	Température finale modifiée T^T_{mod} (°C)	Capacité calorifique (MW/°C)
H1	245	35	0.15
H2	195	75	0.25
C1	25	185	0.2
C2	145	235	0.3

Pour chaque intervalle de température dT_i, nous établissons le bilan d'énergie selon les étapes suivantes :

1) Identifier les courants froids en présence.
2) Déterminer la demande d'énergie selon la relation :

$$demande = \sum CP_c dT_i \qquad [5.36]$$

3) Identifier les courants chauds en présence.
4) déterminer l'apport d'énergie selon la relation :

$$apport = \sum CP_h dT_i \qquad [5.37]$$

Dans chaque intervalle de température, le ***surplus d'énergie*** est calculé par la relation :

$$\{surplus\} = \{apport - demande\} \qquad [5.38]$$

Une valeur positive du ***surplus d'énergie*** indique un excédent d'énergie (signe +) et une valeur négative indique un déficit (signe -). En faisant le bilan de puissance dans chaque intervalle de température (figure 5.20), nous constatons que certains intervalles sont déficitaires en énergie, tandis que d'autres sont excédentaires.

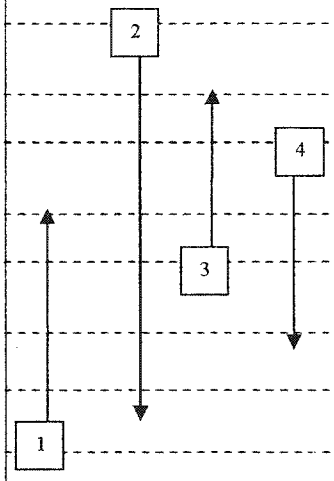
Intervalle de température	Tracé des courants	dT_i	Demande d'énergie $\sum CP_c dT_i$	Apport d'énergie $\sum CP_h dT_i$	Surplus
245°C					
		10	0	1,5	+1,5
235°C		40	12	6	-6
195°C		10	3	4	+1
185°C		40	20	16	-4
145°C		70	14	28	+14
75°C		40	8	6	-2
35°C		10	2	0	-2
25°C					

Figure 5.20 : Bilan d'énergie par intervalle de température

À partir des bilans de puissance établis pour chaque intervalle, nous pouvons construire la table de cascade d'énergie en utilisant la méthode du '***Problem Table Algorithm***'. Elle consiste donc, en allant dans le sens décroissant de l'échelle de température, à transférer tout surplus de chaleur d'un intervalle à l'autre. Ceci est possible car tout excès de chaleur d'un courant chaud dans un intervalle est suffisamment chaud pour combler un déficit d'un courant froid dans l'intervalle suivant. À la Figure 5.21 est présentée la

procédure de cascade d'énergie. Le premier intervalle a un surplus de +1.5. Le second intervalle a un surplus de -6. Si nous réalisons la cascade d'énergie dans le sens décroissant des températures, nous avons donc : Le surplus de l'intervalle1 est ajouté au surplus du second intervalle. Le résultat est -4.5 et est ajouté au surplus du troisième intervalle (+1). En procédant ainsi nous aboutissons à la fin à un surplus de +2.5.

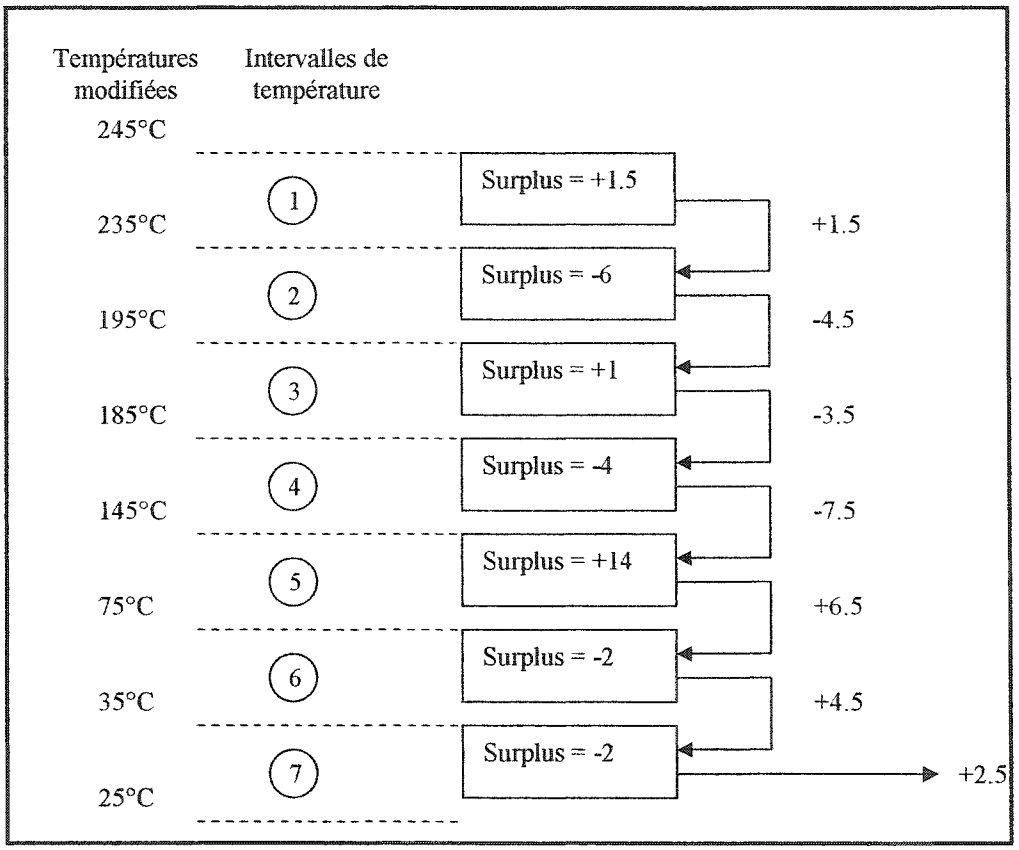


Figure 5.21 : Cascade du surplus d'énergie

La cascade du surplus est {+1.5, -4.5, -3.5, -7.5, +6.5, +4.5, +2.5}. Nous voyons que certaines cascades sont négatives, ce qui est infaisable. La chaleur ne peut pas être transférée dans le sens ascendant de l'échelle des températures. Pour rendre la cascade d'énergie faisable, il faut pourvoir suffisamment de chaleur à partir d'une source chaude

afin de rendre au moins le flux de chaleur nul dans le sens ascendant de l'échelle des températures. La plus petite quantité de chaleur à fournir par une source chaude sera alors égale à la plus large valeur négative du flux de chaleur (Voir la Figure 5.21). La plus large valeur négative que nous avons dans la cascade de surplus d'énergie est -7.5. Si nous ajoutons +7.5 unités de chaleur au sommet du système et réalisons une cascade descendante nous obtenons le résultat de la Figure 5.22. La cascade du surplus d'énergie devient {+9, +3, +4, 0, +14, +12, +10}.

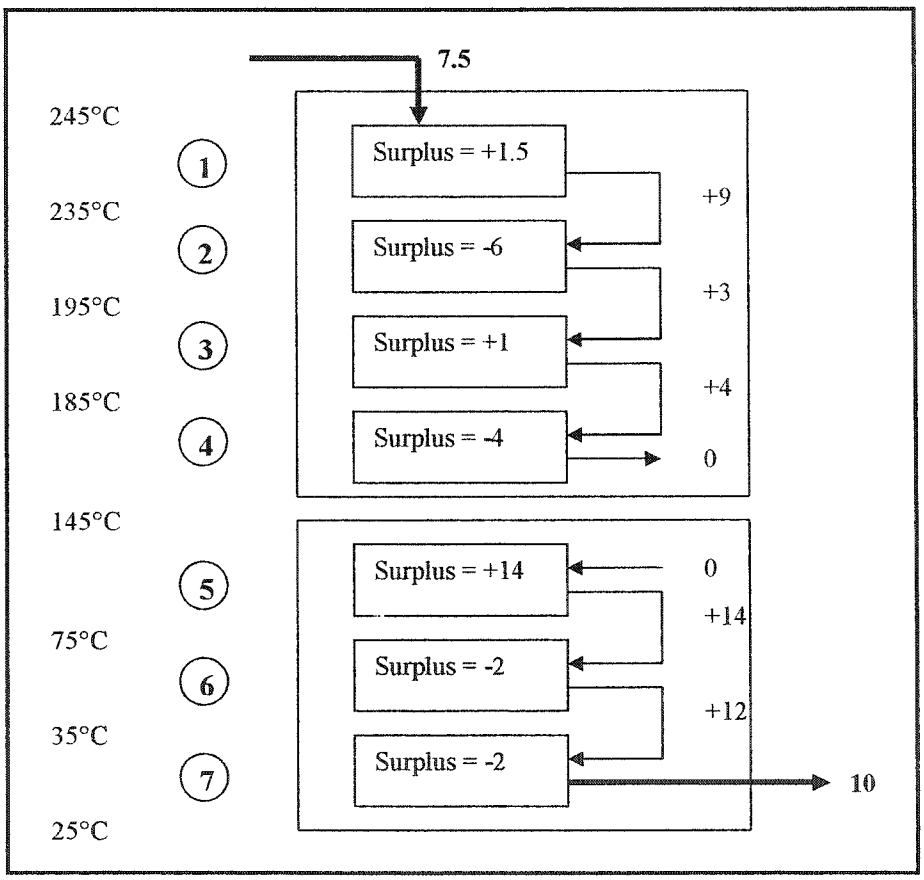


Figure 5.22 : Ajout d'une source chaude au système

Tous les éléments négatifs ont disparus et nous voyons également que l'écoulement de chaleur entre les intervalles 4 et 5 est zéro. Nous avons alors deux sous-systèmes en

équilibre thermique : Le premier sous-système inclut les intervalles 1 à 4 dans lesquels la chaleur demandée par le procédé est satisfaite par les échanges de chaleur entre les courants et par une utilité chaude de puissance égale à 7.5 MW. Le second sous-système englobe les intervalles 5 à 7 dans lesquels les besoins du procédé sont satisfaits par les échanges de chaleur entre les courants et par une utilité froide de puissance 10 MW.

Le point où l'écoulement de chaleur devient zéro correspond au point de pincement. La température de pincement des courants chauds est égale à $145+\Delta T_{min}/2$ =150°C et la température de pincement des courants froids est égale à $145-\Delta T_{min}/2$ =140°C.

5.7 *Optimisation de la distribution de charges par la méthode du simplexe*

Le modèle de programmation linéaire développée pour la simulation de la récupération de chaleur peut être résolu dans une approche mathématique combinée à l'algorithme génétique. Dans cette approche la topologie du réseau est définie par les algorithmes génétiques, l'optimisation de la distribution de charge étant réalisée par la méthode du simplexe. La méthode du simplexe est un algorithme permettant de calculer une chaîne de solutions de base admissibles qui améliorent graduellement la valeur de la fonction objectif. Son principe de fonctionnement se résume ainsi :

1) Déterminer une première solution de base réalisable; cette solution ***initiale*** sert de départ au cheminement vers la solution optimale (si elle existe).
2) Si la solution obtenue en (1) n'est pas optimale, déterminer une autre solution de base réalisable qui permet d'améliorer la fonction objectif.
3) On répète cette procédure itérative jusqu'à ce qu'il ne soit plus possible d'améliorer la fonction objectif. La dernière solution de base réalisable obtenue constitue la solution optimale au programme linéaire.

L'algorithme permet, avec divers critères, de détecter si la solution optimale est unique ou s'il existe des solutions optimales multiples ou encore si le modèle ne permet pas d'obtenir une valeur finie pour la fonction objectif.

Pour établir l'équation de la contrainte de température, considérons le quatrième échangeur du REC présenté à la Figure 5.15 :
À l'extrémité froide de l'échangeur 4, les températures des courants chauds et froids sont respectivement :

$$Th_4^o = T_1^T + \frac{QU_1}{CP_1} + \frac{X_1}{CP_1} \qquad [5.40]$$

$$Tc_4^i = T_3^S + \frac{X_1}{CP_3} + \frac{X_2}{CP_3} \qquad [5.41]$$

À l'extrémité chaude de l'échangeur 4, les températures des courants chauds et froids sont respectivement :

$$Th_4^i = Th_4^o + \frac{X_4}{CP_1} = T_1^T + \frac{QU_1}{CP_1} + \frac{X_1}{CP_1} + \frac{X_4}{CP_1} \qquad [5.42]$$

$$Tc_4^o = Tc_4^i + \frac{X_4}{CP_1} = T_3^S + \frac{X_1}{CP_3} + \frac{X_2}{CP_3} + \frac{X_4}{CP_3} \qquad [5.43]$$

La différence de température entre les courants chaud 1 (H1) et froid 1 (C1) à l'extrémité froide de l'échangeur est :

$$Th_4^o - Tc_4^i = \left(\frac{1}{CP_1} + \frac{1}{CP_3}\right) X_1 - \frac{1}{CP_3} X_2 + \frac{1}{CP_1} QU_1 + \left(T_1^T - T_3^S\right) \geq \Delta T_{\min} \qquad [5.44]$$

et la différence de température à l'extrémité chaude est:

$$Th_4^i - Tc_4^o = \left(\frac{1}{CP_1} - \frac{1}{CP_3}\right) X_1 + \frac{1}{CP_3} X_2 + \left(\frac{1}{CP_1} - \frac{1}{CP_3}\right) X_4 + \frac{1}{CP_1} QU_1 + \left(T_1^T - T_3^S\right) \geq \Delta T_{\min} \qquad [5.45]$$

Dix contraintes inégalités seront ainsi générées pour les 5 échangeurs de chaleur que comprend le réseau.

En remplaçant les notations QU_1, QU_2, QU_3, QU_4 respectivement par X_6, X_7, X_8, X_9 et en réaménageant les équations [5.44] et [5.45] nous pouvons écrire:

$$-\left(\frac{1}{CP_1}-\frac{1}{CP_3}\right)X_1+\frac{1}{CP_3}X_2-\frac{1}{CP_1}X_6\leq-\Delta T_{\min}+\left(T_1^T-T_3^S\right) \qquad [5.46]$$

$$-\left(\frac{1}{CP_1}-\frac{1}{CP_3}\right)X_1-\frac{1}{CP_3}X_2-\left(\frac{1}{CP_1}-\frac{1}{CP3}\right)X_4-\frac{1}{CP_1}X_6\leq-\Delta T_{\min}+\left(T_1^T-T_3^S\right) \qquad [5.47]$$

Les équations [5.46] et [5.47] exprimant la différence de température entre le courant chaud et le courant froid aux extrémités d'un échangeur de chaleur peuvent se mettre sous la forme suivante : $\sum a_{ij}X_j \leq B_i$ [5.48]

Avec $i = 1,\ldots,N_{inéq}$, $j = 1,\ldots,n$

où n est le nombre de variables comprenant les puissances des échangeurs de chaleur et des utilités, $N_{inéq}$ est le nombre de contraintes inégalité.

De la même manière nous pouvons développer l'équation du bilan d'énergie pour chaque courant. Par exemple, pour le courant 1, le bilan d'énergie s'écrit :

$$X_1+X_4+X_5+QU_1=CP_1\left(T_1^S-T_1^T\right) \qquad [5.49]$$

En remplaçant QU_1 par X_6, cette équation devient :

$$X_1+X_4+X_5+X_6=CP_1\left(T_1^S-T_1^T\right) \qquad [5.50]$$

Les équations définissant les bilans d'énergie sur les courants peuvent se mettre sous la forme générale :

$$\sum a_{ij}X_j=B_i \qquad [5.51]$$

avec $i = 1,\ldots,N_{st}$, $j = 1,\ldots,n$

où n est le nombre de variables comprenant les puissances des échangeurs de chaleur et des utilités, N_{st} est le nombre de courants.

Si nous définissons la fonction objectif comme une maximisation de la puissance récupérée par les échangeurs de chaleur, le modèle de programmation linéaire peut être mis
sous la forme générale, donnée par:

$$Maximiser \quad f = \sum_{i=1}^{N_{EC}} c_i X_i \qquad [5.52]$$

liée aux contraintes

$$\left.\begin{aligned} a_{11}X_1 + a_{12}X_2 + \ldots\ldots\ldots\ldots + a_{1n}X_n &= B_1 \\ a_{21}X_1 + a_{22}X_2 + \ldots\ldots\ldots\ldots + a_{2n}X_n &= B_2 \\ \cdot \quad \cdot \quad \cdot \quad \cdot \end{aligned}\right\} \qquad [5.53]$$

$$\left.\begin{aligned} a_{51}X_1 + a_{52}X_2 + \ldots\ldots\ldots\ldots + a_{5n}X_n &\le B_5 \\ a_{61}X_1 + a_{62}X_2 + \ldots\ldots\ldots\ldots + a_{6n}X_n &\le B_6 \\ \cdot \quad \cdot \quad \cdot \\ a_{m1}X_1 + a_{m2}X_2 + \ldots\ldots\ldots\ldots + a_{mn}X_n &\le B_m \end{aligned}\right\} \qquad [5.54]$$

$$X_i \ge 0$$

$c_i = 1$ si la variable X_i est présente, et $c_i = 0$ dans le cas contraire; m est le nombre de contraintes, n est le nombres de variables.

L'équation 5.52 définit la fonction objectif qui est traduite ici par une maximisation de la puissance récupérée par les échangeurs de chaleur. L'équation 5.53 représente les bilans d'énergie des courants. L'équation 5.54 représente les contraintes de température au niveau des échangeurs de chaleur.

Il faut noter que la fonction objectif traduite par une maximisation de la puissance récupérée par les échangeurs de chaleur est équivalente à la fonction objectif de minimisation de la puissance des utilités chaudes et froides.

5.7.1 Forme standard d'un programme linéaire

Pour appliquer la méthode du simplexe, le programme linéaire doit être mis sous une forme standard. La forme standard d'un modèle de programmation linéaire définie par Baillargeon, 1996 [41], doit respecter les trois conditions suivantes:

- Les contraintes sont exprimées sous forme d'équations
- Le second membre des contraintes (les B_i) doit être non négatif
- Toutes les variables sont non négatives.

Une contrainte inégalité peut être convertie en une égalité par l'introduction dans le modèle de nouvelles variables appelées ***variables d'écart.*** Si la contrainte est de la forme $a_{i1}X_1 + a_{i2}X_2 + \ldots\ldots\ldots + a_{in}X_n \leq B_i$, elle est convertie en une contrainte égalité en ajoutant une variable d'écart non négative X_{n+j} tel que :

$$a_{i1}X_1 + a_{i2}X_2 + \ldots\ldots\ldots + a_{in}X_n + X_{n+j} = B_i \qquad [5.55]$$

où $B_i \geq 0$ $\quad X_{n+j} \geq 0$ $\quad j = 1$ à p (p est le nombre de contraintes inégalité)

Si la contrainte est de la forme $a_{i1}X_1 + a_{i2}X_2 + \ldots\ldots\ldots + a_{in}X_n \geq B_i$, elle est convertie en une contrainte égalité en soustrayant une variable d'écart non négative X_{n+j} tel que :

$$a_{i1}X_1 + a_{i2}X_2 + \ldots\ldots\ldots + a_{in}X_n - X_{n+j} = B_i \qquad [5.56]$$

où $X_{n+j} \geq 0$, $B_i \geq 0$

Ainsi, la forme standard du problème défini par les équations [5.52], [5.53] et [5.54] devient :

$$\textit{Maximiser} \quad f = \sum_{i=1}^{N_{EC}} c_i X_i \qquad [5.57]$$

liée aux contraintes

$$\left.\begin{aligned} a_{11}X_1 + a_{12}X_2 + \ldots\ldots\ldots\ldots + a_{1n}X_n &= B_1 \\ a_{21}X_1 + a_{22}X_2 + \ldots\ldots\ldots\ldots + a_{2n}X_n &= B_2 \\ \cdot \quad\quad \cdot \quad\quad \cdot & \end{aligned}\right\} \qquad [5.58]$$

$$\left.\begin{aligned} a_{51}X_1 + a_{52}X_2 + \ldots\ldots\ldots\ldots + a_{5n}X_n + X_{n+1} &= B_5 \\ a_{61}X_1 + a_{62}X_2 + \ldots\ldots\ldots\ldots + a_{6n}X_n + X_{n+2} &= B_6 \\ \cdot \quad\quad \cdot \quad\quad \cdot & \\ a_{m1}X_1 + a_{m2}X_2 + \ldots\ldots\ldots\ldots + a_{mn}X_n + X_{n+p} &= B_m \end{aligned}\right\} \qquad [5.59]$$

Avec $X_n \geq 0, \qquad B_i \geq 0$

Le problème comporte maintenant *m* équations et s variables (s = *n* + *p*). Le programme linéaire mis sous la forme standard forme un système d'équations linéaires que nous résolvons par la méthode du simplexe. Si le programme linéaire ne peut être mis sous la forme standard, une autre méthode (la méthode des deux phases) doit être utilisée. La résolution d'un système de ***m*** équations linéaires et ***s*** variables peut conduire aux situations suivantes :

1- Il n'y a pas de solution

2- Une solution existe et elle est unique

3- Il y a une infinité de solutions

5.7.2 Le Tableau du simplexe

Pour utiliser la méthode du simplexe, le problème est mis sous la forme d'un tableau appelé ***Tableau du simplexe*** (Tableau 5.3). La procédure de résolution que nous utilisons a été décrite par Kolman et Beck (1980) [42].

Tableau 5.3 : Tableau du simplexe

Lignes	Variables							Solution de base
	f	X_1	X_2	..... X_n	X_{n+1}	X_{n+2}	X_{n+q}	
0	1	$-c_1$	$-c_2$	 0	0	0	0	*0*
1	0	a_{11}	a_{12}	..... a_{1n}	0	0	0	B_1
2	0	a_{21}	a_{22}	a_{2n}	0	0	0	B_2
.	.	.	.	.	.	.	.	.
.	.	.	.	.	.	.	.	.
5	0	a_{51}	a_{52}	... a_{5n}	+1	0	0	B_5
6	0	a_{61}	a_{62}	... a_{6n}	0	+1	0	B_6
.	0	.	.	.	.	.	.	.
.	0	.	.	.	.	.	.	.
m	0	a_{m1}	a_{m2}	..... a_{mn}	0	0	+1	B_m

Dans ce tableau, la fonction objectif est réécrite sous la forme suivante:

$$f - \sum_{i=1}^{N_{EC}} c_i X_i = 0 \qquad [5.60]$$

Dans le tableau du simplexe, la ligne 0 correspond à la ***ligne de la fonction objectif*** définie par l'équation [5.60]. Les lignes 1 à m représentent les coefficients des contraintes du programme linéaire mis sous la forme standard (équations [5.58] et [5.59]). La recherche de la solution optimale à l'aide du tableau du simplexe procède comme suit :

1-Établir le tableau initial du simplexe
2-Appliquer le test d'optimalité : Si la ligne de la fonction objectif ne comporte pas de coefficient négatif alors la solution indiquée est optimale; arrêter la recherche, sinon,
3-Trouver la colonne pivot en déterminant la colonne ayant le coefficient le plus négatif sur la ligne de la fonction objectif. S'il y a plusieurs colonnes possibles, en choisir une.
4-Trouver la ligne pivot : Pour cela faire les rapports θ en divisant les coefficients de la colonne des solutions (excepté le coefficient de la ligne de la fonction objectif) par les coefficients correspondants de la colonne pivot en utilisant seulement ceux qui sont positifs. La ligne pivot est la ligne ayant le plus petit rapport. Si deux ou plusieurs rapports sont identiques, choisir une des lignes possibles. Si aucun des coefficients dans la colonne pivot au-dessous de la ligne de la fonction objectif n'est positif, le problème n'a pas d'optimum fini. Arrêter dans ce cas la procédure de recherche, sinon,
5-Obtenir un nouveau tableau par pivot et retourner à l'étape 2.

5.7.3 Solution de base faisable

Le nombre de ***solutions de base*** d'un programme linéaire est défini par:

$$\binom{s}{m} = \frac{s!}{m!(s-m)!} \qquad [5.61]$$

Lorsque la solution de base satisfait à toutes les contraintes de non négativité, la solution est alors appelée ***solution de base faisable.*** Pour commencer la méthode du simplexe, il faut trouver une solution de base faisable. Ceci est possible si les B_i sont positifs.

Une solution de base au système de m équations linéaires $Ax = B$ comprenant s variables peut être obtenue en égalant s-m variables à zéro et en résolvant le système pour les m variables restantes. Les s-m variables mises à zéro sont des ***variables hors base*** alors que les variables restantes sont des ***variables de base***. Dans le tableau du simplexe une ***variable de base*** a les propriétés suivantes :

1) Elle apparaît dans une seule équation et dans cette équation elle a un coefficient de valeur (+1).
2) La colonne correspondante de la variable comporte des zéros, sauf le (+1) dans la ligne de la variable de base.
3) La valeur de la variable de base est la valeur du coefficient dans la colonne des solutions (solution de base).

5.7.4 Méthode des deux phases

L'algorithme du simplexe débute toujours avec une solution de base réalisable. Ceci est facile à obtenir si le programme linéaire comporte des contraintes de type inférieur ou égal (≤) et que les B_i sont tous positifs. Une solution de base initiale faisable s'obtient ainsi facilement en ajoutant des variables d'écart. Ces variables servent de variables de base pour obtenir le Tableau initial du simplexe. Cependant, certains problèmes pratiques contiennent dans leurs modélisations des contraintes de types supérieur ou égal (≥), inférieur ou égal (≤) et des contraintes égalité (=). En plus de cela, les B_i ne sont pas toujours positifs. C'est le cas par exemple du modèle pour la récupération de chaleur. Une solution de base réalisable au départ ne sera pas toujours obtenue aussi facilement dans ce système de contraintes mixtes. Dans ce cas, il faut avoir recours à des ***variables artificielles*** pour obtenir une solution de départ, et appliquer par la suite l'algorithme du simplexe en deux phases appelé ***méthode en deux phases***.

Dans le cas de contraintes de type "≥" ' ou du type "=", on devra, en plus de soustraire une variable d'écart pour les contraintes du type "≥", ajouter une variable artificielle pour ces contraintes.

Soit le programme linéaire mis sous la forme standard :

$$Maximiser \quad f = \sum_{i=1}^{N_{EC}} c_i X_i \qquad [5.62]$$

Liées aux contraintes

$$\sum a_{ij} X_j = B_i \qquad [5.63]$$

$$X_j \geq 0 \qquad i = 1,.....m \qquad j = 1,.....s$$

où les B_i ne sont pas tous positifs.

Pour obtenir une solution de base faisable, nous devons introduire une variable artificielle dans l'équation [5.63]. Le modèle devient alors :

$$Maximiser \quad f = \sum_{i=1}^{N_{EC}} c_i X_i \qquad [5.64]$$

liées aux contraintes

$$\sum a_{ij} X_j + y_i = B_i \qquad [5.65]$$

Avec $X_j \geq 0$, $y_i \geq 0, B_i \geq 0$

$i = 1,2,.....m$ et $j = 1,2,.....s$

Les variables artificielles y_i jouent seulement un rôle utilitaire et servent à obtenir une solution de départ. Il faut noter que l'introduction des variables artificielles modifie de façon importante le problème original. Pour obtenir une solution de base faisable au problème original, il faut que les variables artificielles soient réduites à zéro. C'est ce qui est fait à travers la procédures des deux phases.

Le but de la phase 1 est de déterminer une solution faisable de base au problème original, en éliminant les variables artificielles du système de contraintes. Il s'agit d'abord d'introduire les variables artificielles nécessaires dans le système de contraintes, par la suite, on peut appliquer les critères associés à un problème de minimisation en optimisant la fonction objectif auxiliaire suivante :

$$Minimiser\ f^* = \sum y_i \qquad [5.66]$$

liées aux contraintes

$$\sum a_{ij} X_j + y_i = B_i \qquad [5.67]$$

Avec :

$X_j \geq 0$, $y_i \geq 0$, $B_i \geq 0$

$i = 1, 2, \ldots\ldots m$ et $j = 1, 2, \ldots\ldots s$

où m est le nombre de contraintes. Si le critère d'optimalité est atteint et que $f^* > 0$, alors il y a au moins une variable artificielle dans la base : le programme linéaire n'admet pas de solution faisable. Par contre si $f^* = 0$, le programme linéaire admet des solutions faisables : la phase 1 est terminée (il n'est pas nécessaire de satisfaire au critère d'optimalité). On passe à la phase 2.

La phase 2 consiste à optimiser la fonction objectif ne comportant que des variables légitimes (c'est-à-dire les variables autres que les variables artificielles), en appliquant l'algorithme du simplexe. La solution optimale obtenue à la phase 1 est utilisée pour obtenir une solution de base initiale faisable pour le problème original. Dans le cas où les variables artificielles ont été éliminées de la base (phase 1), on débute la phase 2 avec un tableau réduit, éliminant les colonnes associées aux variables auxiliaires. L'application de la méthode du simplexe pour l'optimisation de la puissance récupérée dans les réseaux d'échangeurs de chaleur est démontrée au chapitre 6.

5.8 Conclusion

Dans ce chapitre nous avons présenté une procédure d'optimisation des réseaux d'échangeurs de chaleur par les algorithmes génétiques. Les AG sont bien adaptés pour traiter des problèmes d'optimisation intégrant simultanément des variables discrètes de décision et des variables continues. Ne requérant aucune dérivation, ils sont beaucoup plus souples et permettent de résoudre tous les types de problèmes. Cependant les algorithmes génétiques comportent des paramètres qui doivent être réglés par expérimentations pour chaque cas de problème donné.

L'optimisation paramétrique des REC par les algorithmes génétiques a un avantage par rapport à l'optimisation par la méthode du simplexe en ce sens que l'algorithme du simplexe requiert toujours une solution initiale faisable à partir de laquelle la recherche de l'optimum est exécutée. Si cette solution initiale n'existe pas, le chromosome est simplement rejeté, ce qui n'est pas le cas de l'optimisation paramétrique par les AG qui pénalisent le chromosome au lieu de le rejeter systématiquement. Le chromosome pénalisé peut contenir des informations utiles pour la future génération.

CHAPITRE VI

6 EXEMPLES D' APPLICATION

Nous présentons dans ce chapitre des résultats d'expérimentations effectuées sur des problèmes d'optimisation qui ont traités dans la littérature. Nous avons appliqué les algorithmes génétiques à la synthèse de réseaux d'échangeurs en utilisant les deux approches suivantes:

- L'approche génétique/simplexe (G/S) pour désigner l'optimisation de la topologie par les algorithmes génétiques et l'optimisation de la distribution de charges par le simplexe (programme linéaire). Nous présenterons cette approche uniquement avec le premier exemple d'application.
- L'approche génétique/génétique (G/G) pour désigner l'optimisation de la topologie du REC par les algorithme génétiques et l'optimisation de la distribution de charges également par les algorithmes génétiques.

Ces deux approches ont été codées dans le langage de programmation FORTRAN 90. Les calculs ont été effectués sur un processeur Pentium IV – 1.6 MHz.

6.1 Étude de cas 1

Le premier problème que nous utilisons pour vérifier l'applicabilité de notre algorithme est le problème qui a servi à la démonstration de la méthode du pincement (Smith,. 1995) [27]. Le procédé comporte deux courants chauds (H1, H2), et deux courants froids (C1, C2). Les caractéristiques (températures initiale et finale, capacité calorifique) ainsi que les besoins énergétiques pour chaque courant sont consignées dans le Tableau 6.1.

Tableau 6.1 : Caractéristiques des courants : Étude de cas 1

Fluide	Température initiale T^S (°C)	Température finale T^T(°C)	Charge thermique (MW)	Capacité calorifique CP (MW/°C)
H1	250	40	31.5	0.15
H2	200	80	30	0.25
C1	20	180	32	0.2
C2	140	230	27	0.3

La température minimale d'approche utilisée est ΔT_{min} = 10°C.

Cette température d'approche détermine la puissance à récupérer ainsi que les besoins suivants :

Puissance maximale récupérable : Q_{REC} = 51.5 MW

Besoin minimum en utilité chaude : QH_{min} = 7.5 MW

Besoin minimum en utilité froide : QC_{min} = 10 MW

6.1.1 Optimisation par l'approche Génétique/Simplexe

Les paramètres utilisés pour l'algorithme de recherche sont les suivants:

Taille de la population : 60

Nombre de générations : 200

Probabilité de croisement : 70%

Probabilité de mutation : 10%

Nombre de niveaux: 3

Nous présentons à la Figure 6.1 les courbes d'évolution de la puissance maximale du meilleur chromosome, et de la puissance moyenne de la population en fonction du nombre de générations. On peut constater sur cette figure que l'amélioration de la population est très rapide au début (recherche globale) et devient de plus en plus lente au fil des générations (recherche locale). Le bruit dans la moyenne est essentiellement dû aux mutations. La valeur moyenne des puissances au cours des générations a tendance à

se rapprocher de celle de l'individu le plus adapté (puissance maximale). Cela correspond à une uniformisation croissante de la population. La forme en 'escalier' de la courbe de puissance maximale (Figure 6.1) est due au fait que nous avons utilisé le concept de l'élitisme dans la procédure de recherche de la solution optimale. À chaque génération, le patrimoine génétique du meilleur chromosome est préservé pour être utilisé dans la génération suivante.

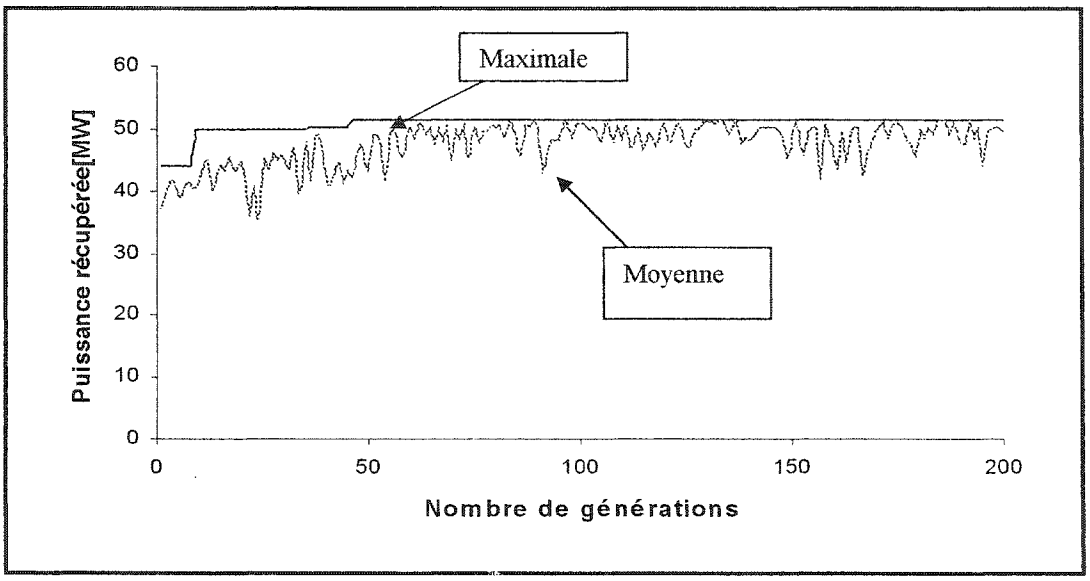


Figure 6.1 : Convergence de la solution avec élitisme

La Figure 6.2 ci-dessous illustre un cas de recherche génétique où l'élitisme n'est pas pris en compte. On peut donc voir comment au cours des générations, la meilleure performance évolue de façon aléatoire. D'une génération à l'autre, la puissance maximale du meilleur chromosome peut évoluer indifféremment, dans le sens croissant ou décroissant.

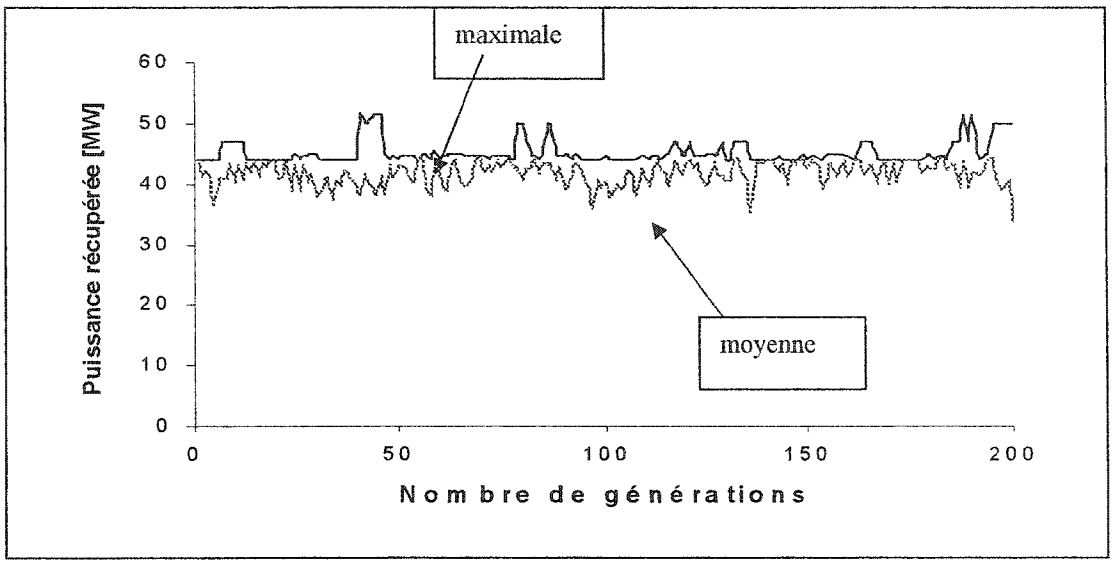


Figure 6.2 : Convergence de la solution sans élitisme

La meilleure structure de REC générée par l'algorithme est le chromosome codé par la séquence de gènes **[1 0 2 2 1 1]**. Le décodage de ce chromosome donne la topologie du réseau présentée à la Figure 6.3 (partie encadrée).

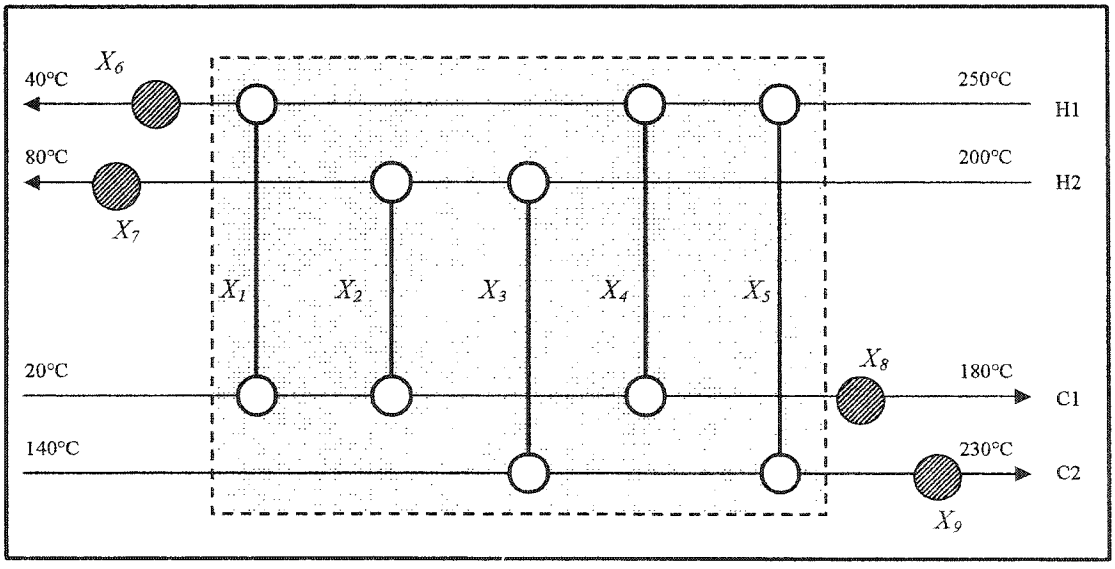


Figure 6.3 : Solution pour la meilleure structure de REC (étude de cas 1)

Le schéma complet est obtenu après ajout des utilités définies par l'optimisation paramétrique. Sur la Figure 6.3, et pour toutes les REC que nous présentons dans ce document, l'ordre de notation des échangeurs de chaleur est de gauche à droite. La charge du premier échangeur est notée X_1, le suivant X_2 et ainsi de suite jusqu'au dernier échangeur de chaleur. La notation des charges des utilités commence à partir de l'utilité du premier courant chaud vers le dernier courant froid, à la suite du numéro du dernier échangeur de chaleur. Même si l'utilité sur un courant n'existe pas, elle reçoit néanmoins un numéro.

Le Tableau 6.2 est le tableau final obtenu par la méthode du simplexe. Ce tableau donne la distribution optimale des puissances. Selon les conditions d'optimalité du simplexe, l'optimum de la fonction est atteint quand les coefficients de la ligne de la fonction objectif (la ligne 0) sont tous positifs ou nuls.

Tableau 6.2 : Tableau final du simplexe (1e solution)

Ligne	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	X_{16}	X_{17}	X_{18}	X_{19}	Solution
0	0	0	0	0	0	0	0	0	0	0	0	0.2	0.15	0	0	0.05	0	0	0	51.5
1	0	0	0	0	0	0	0	0	0	0	0	-0.4	0	-0.5	0	0	0	0	1	29.17
2	0	0	0	0	0	1	1	0	0	0	0	-0.2	-0.2	0	0	-0.1	0	0	0	10
3	0	0	0	0	0	0	0	1	0	0	0	-0.1	-0.2	0.15	0	-0.1	0	0	0	1.75
4	0	0	1	0	0	0	0	0	0	0	0	0.25	0	0	0	0	0	0	0	12.5
5	0	0	0	0	0	-6.7	0	0	0	1	0	0	0	0	0	0	0	0	0	10
6	1	0	0	0	0	1	0	0	0	0	0	-0.2	-0.2	0	0	0.15	0	0	0	16.5
7	0	0	0	0	0	0	0	0	0	0	0	-0.8	0	0	0	0	1	0	0	8.33
8	0	0	0	0	0	-1	0	0	0	0	1	-0.1	0.15	0	0	-1	0	0	0	7.5
9	0	0	0	1	0	0	0	0	0	0	0	0.28	0.15	-0.2	0	-0.2	0	0	0	6.25
10	0	0	0	0	0	-5	0	0	0	0	0	-0.3	-0.3	0	1	0.25	0	0	0	37.5
11	0	1	0	0	0	-1	0	0	0	0	0	-0.1	0.15	0	0	0.05	0	0	0	7.5
12	0	0	0	0	0	0	0	0	1	0	0	-0.1	0	-0.2	0	0	0	0	0	5.75
13	0	0	0	0	0	0	0	0	0	0	0	0.46	-0.8	-0.3	0	-0.3	0	1	0	10.42
14	0	0	0	0	1	0	0	0	0	0	0	-0.1	0	0.15	0	0	0	0	0	8.75

La solution optimale (puissance maximale récupérée Q_{REC}) est lue dans la colonne des solutions, sur la ligne 0 (Q_{REC} = 51.5 MW). Les valeurs obtenues pour les variables de base X_1, X_2, X_3, X_4, X_5, X_7, X_8, X_9 (Tableau 6.2) constituent les valeurs optimales pour assurer la récupération maximale de chaleur. Ces variables sont les paramètres du problème. Les utilités chaudes sont constituées de X_6 (QU_1) et X_7 (QU_2). L'utilité $X_6 = 0$ car elle n'est pas une variable de base , alors $QH_{min} = X_6 + X_7 = X_7 = 10$ MW. Les utilités froides sont constituées de X_8 (QU_3) et X_9 (QU_4), alors $QC_{min} = X_8 + X_9 = 7.5$ MW. Les variables $X_1 = 16.5$, $X_2 = 7.5$, $X_3 = 12.5$, $X_4 = 6.25$ et $X_5 = 8.75$ sont les puissances optimales des échangeurs de chaleur. Les variables X_{10} à X_{19} ne sont pas des paramètres du problème. Ce sont les variables auxiliaires qui ont juste servi à la résolution du problème.

Le Tableau 6.2 nous renseigne qu'il est possible d'obtenir d'autres solutions alternatives pour ce problème, du point de vue de la distribution des charges. En effet, les variables X_6 et X_{14} qui sont des variables hors base peuvent être transformées en variables de base. La transformation de la variable hors base X_6 en une variable de base donne une deuxième solution alternative (Tableau 6.3): Les valeurs nulles des coefficients situés à l'intersection de la ligne de la fonction objectif (ligne 0) et des colonnes des variables X_6 et X_{14} indiquent que l'on peut accroître les valeurs de ces variables, sans modifier la valeur optimale de la fonction objectif. Ceci implique nécessairement que certaines variables de la base seront modifiées (pouvant même être exclues de la base). L'introduction des variables X_6 et X_{14} dans la base va donc donner lieu à une autre distribution des charges thermiques. La puissance maximale récupérée reste inchangée (Q_{REC} = 51.5 MW), seule la distribution des charges au niveau des échangeurs change. Les variables de base dans ce cas sont X_1, X_2, X_3, X_4, X_5, X_6, X_8, X_9 . L'autre alternative consiste à faire entrer la variable X_{14} dans la base (à partir du Tableau 6.2). La transformation de X_{14} en une variable de base définit une troisième solution alternative de la distribution des charges (Tableau 6.4). Les variables de base de cette solution alternative sont X_1, X_2, X_3, X_4, X_5 , qui sont les charges des échangeurs de chaleur, et X_7, X_9 pour les

charges des utilités. Cette solution comporte moins d'utilités que les deux solutions précédentes.

Tableau 6.3 : Tableau final du simplexe (2e solution alternative)

Ligne	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	X_{16}	X_{17}	X_{18}	X_{19}	Solution
0	0	0	0	0	0	0	0	0	0	0	0	0.2	0.15	0	0	0.05	0	0	0	51.5
1	0	0	0	0	0	0	0	0	0	0	0	-0.4	0	-0.5	0	0	0	0	1	29.17
2	0	0	0	0	0	1	1	0	0	0	0	-0.2	-0.2	0	0	-0.1	0	0	0	10
3	0	0	0	0	0	0	0	1	0	0	0	-0.1	-0.2	0.15	0	-0.1	0	0	0	1.75
4	0	0	1	0	0	0	0	0	0	0	0	0.25	0	0	0	0	0	0	0	12.5
5	0	0	0	0	0	0	6.67	0	0	1	0	-1.3	-1	0	0	-0.3	0	0	0	76.67
6	1	0	0	0	0	0	-1	0	0	0	0	0.05	0	0	0	0.2	0	0	0	6.5
7	0	0	0	0	0	0	0	0	0	0	0	-0.8	0	0	0	0	1	0	0	8.33
8	0	0	0	0	0	0	1	0	0	0	1	-0.3	0	0	0	-1	0	0	0	17.5
9	0	0	0	1	0	0	0	0	0	0	0	0.28	0.15	-0.2	0	-0.2	0	0	0	6.25
10	0	0	0	0	0	0	5	0	0	0	0	-1.3	-1	0	1	0	0	0	0	87.5
11	0	1	0	0	0	0	1	0	0	0	0	-0.3	0	0	0	0	0	0	0	17.5
12	0	0	0	0	0	0	0	0	1	0	0	-0.1	0	-0.2	0	0	0	0	0	5.75
13	0	0	0	0	0	0	0	0	0	0	0	0.46	-0.8	-0.3	0	-0.3	0	1	0	10.42
14	0	0	0	0	1	0	0	0	0	0	0	-0.1	0	0.15	0	0	0	0	0	8.75

Tableau 6.4 : Tableau final du simplexe (3e solution alternative)

Ligne	X_1	X_2	X_3	X_4	X_5	X_6	X_7	X_8	X_9	X_{10}	X_{11}	X_{12}	X_{13}	X_{14}	X_{15}	X_{16}	X_{17}	X_{18}	X_{19}	Solution
0	0	0	0	0	0	0	0	0	0	0	0	0.2	0.15	0	0	0.05	0	0	0	51.5
1	0	0	0	0	0	0	0	3.33	0	0	0	-0.7	-0.5	0	0	-0.2	0	0	1	35
2	0	0	0	0	0	1	1	0	0	0	0	-0.2	-0.2	0	0	-0.1	0	0	0	10
3	0	0	0	0	0	0	0	6.67	0	0	0	-0.5	-1	1	0	-0.3	0	0	0	11.67
4	0	0	1	0	0	0	0	0	0	0	0	0.25	0	0	0	0	0	0	0	12.5
5	0	0	0	0	0	-6.7	0	0	0	1	0	0	0	0	0	0	0	0	0	10
6	1	0	0	0	0	1	0	0	0	0	0	-0.2	-0.2	0	0	0.15	0	0	0	16.5
7	0	0	0	0	0	0	0	0	0	0	0	-0.8	0	0	0	0	1	0	0	8.33
8	0	0	0	0	0	-1	0	0	0	0	1	-0.1	0.15	0	0	-1	0	0	0	7.5
9	0	0	0	1	0	0	0	1	0	0	0	0.2	0	0	0	-0.2	0	0	0	8
10	0	0	0	0	0	-5	0	0	0	0	0	-0.3	-0.3	0	1	0.25	0	0	0	37.5
11	0	1	0	0	0	-1	0	0	0	0	0	-0.1	0.15	0	0	0.05	0	0	0	7.5
12	0	0	0	0	0	0	0	1	1	0	0	-0.2	-0.2	0	0	-0.1	0	0	0	7.5
13	0	0	0	0	0	0	0	1.67	0	0	0	0.33	-1	0	0	-0.3	0	1	0	13.33
14	0	0	0	0	1	0	0	-1	0	0	0	-0.1	0.15	0	0	0.05	0	0	0	7

Les solutions pour l'optimisation paramétrique du REC contenues dans les Tableaux 6.2, 6.3, et 6.4 sont résumées dans le Tableau 6.5 que l'on peut comparer avec la solution de référence (Smith, 1995) [27]. Bien que les distributions de charges soient différentes, le bilan global (QH_{min}, QC_{min}, Q_{REC}) est satisfait. Le choix d'une solution ou l'autre se fera en fonction du nombre d'unités et de la surface totale des échangeurs de chaleur.

Tableau 6.5 : Solutions par l'approche G/S

Variables		Solution G/S (MW)			Solution de référence (MW)
		Solution 1	Solution 2	Solution 3	
Échangeurs de chaleur	X_1	16.5	6.5	16.5	6.5
	X_2	7.5	17.5	7.5	17.5
	X_3	12.5	12.5	12.5	12.5
	X_4	6.25	6.25	8	8
	X_5	8.75	8.75	7	7
Utilités froides	X_6	0	10	0	10
	X_7	10	0	10	0
Utilités chaudes	X_8	1.75	1.75	0	0
	X_9	5.75	5.75	7.5	7.5
Puissance récupérée (Q_{REC}) (MW)		51.5	51.5	51.5	51.5

Le bilan d'énergie est identique pour toutes ces solutions. Le réseau construit récupère 100% de la puissance maximale récupérable. La différence que l'on peut observer est la répartition des charges. Si nous comparons la solution 3 du Tableau 6.5 avec la solution de référence, nous constatons que l'utilité froide (10MW) installée sur le courant chaud 2 se retrouve par contre sur le courant chaud 1 (solution de référence). Il y a une réorganisation de la distribution des charges des échangeurs de chaleur 1 et 2.

Les schémas des Figures 6.4.a et 6.4.b représentent respectivement les solutions de l'AG (solution 3) et la solution de référence. Sur ces figurent sont indiquées les températures aux différents niveaux des courants. Les puissances des échangeurs de chaleur et des utilités (nombres soulignés) sont également représentées.

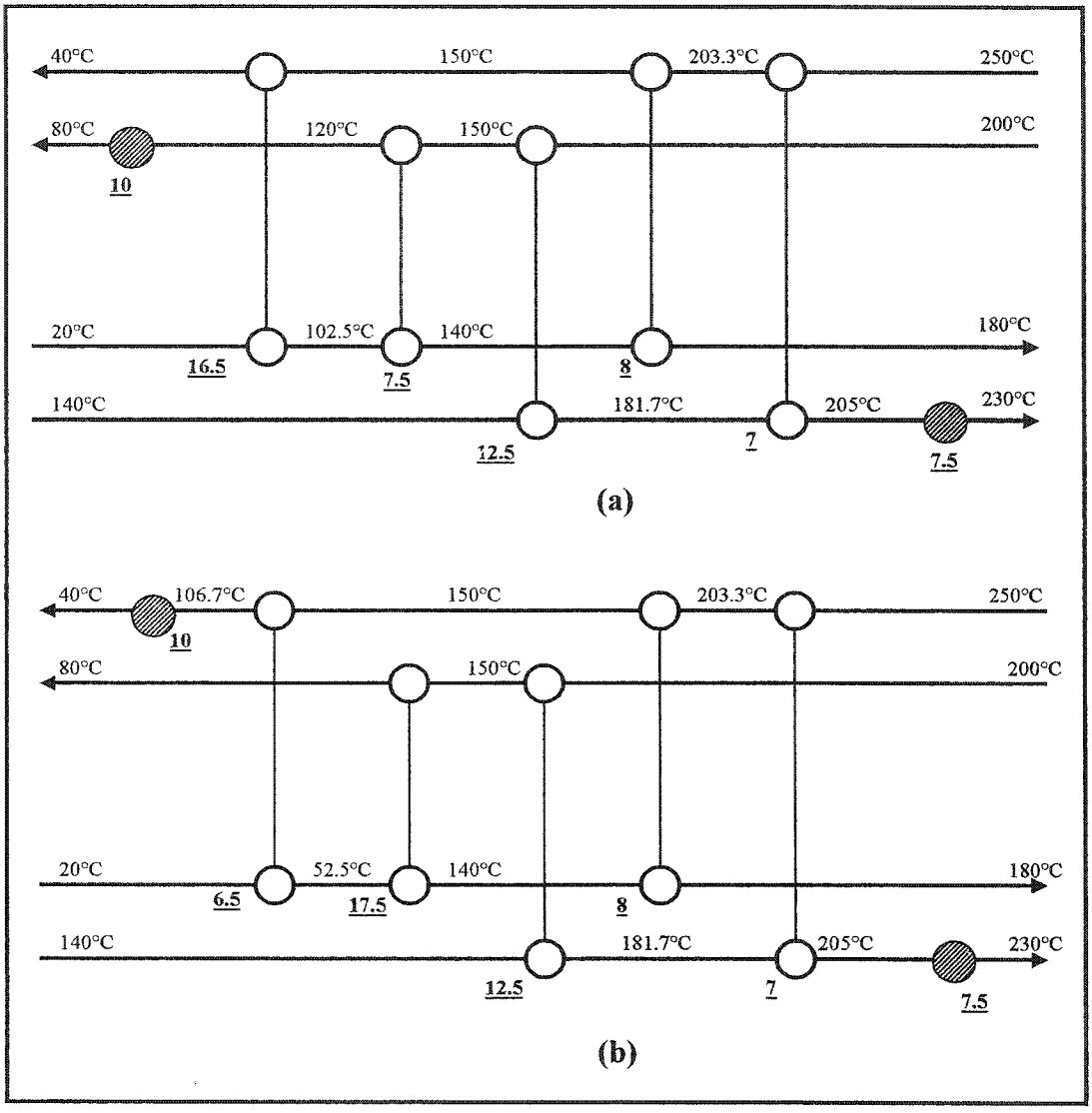


Figure 6.4 : Réseau d'échangeurs de chaleur (solution G/S)

6.1.2 Optimisation par l'approche Génétique/Génétique

Dans cette procédure que nous appelons approche G/G, l'optimisation au niveau de la topologie et celle de la distribution des charges sont assurées exclusivement par les

algorithmes génétiques. Nous manipulons ici deux types de chromosomes : le premier codé par des nombres entiers, détermine la topologie du REC tandis que le second chromosome est celui de la distribution des charges et est codé en binaire. Les paramètres utilisés pour l'algorithme de recherche sont :

Taille de la population : 100

Nombre de générations : 200

Probabilité de croisement : 70%

Probabilité de mutation : 10%

Coefficient de pénalisation $Ck = 1000$

Nombre de niveaux du REC : 3

À la Figure 6.5 sont présentées les courbes d'évolution de la puissance récupérée par le meilleur chromosome et la moyenne de la puissance récupérée par la population. La grande disparité au départ entre les individus de la population s'amoindrit au fil des générations, ce qui traduit une uniformisation de la population.

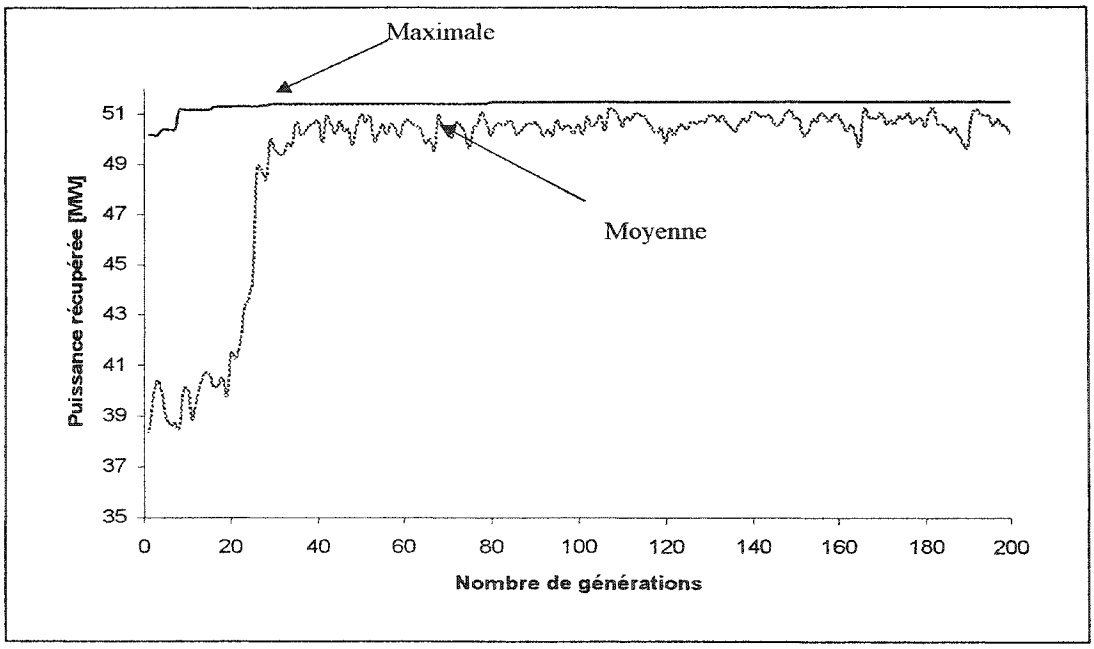


Figure 6.5 : Convergence de la solution (approche G/G)

Pour analyser l'influence du coefficient de pénalité Ck sur le comportement de l'algorithme, nous rappelons ici le modèle que nous avons établi pour simuler la minimisation des besoins en utilités (Voir l'équation 5.31):

$$\textit{Minimiser} \quad F = \sum_{j=1}^{N_{st}} QU_j + Ck \sum_{i=1}^{N_{inéq}} \Phi_i \,,$$

où $\sum_{j=1}^{N_{st}} QU_j$ représente la somme des utilités chaudes et froides, $Ck \times \sum_{i=1}^{N_{inéq}} \Phi_i$ est le terme de pénalisation pour toute violation de la contrainte de température minimale d'approche.

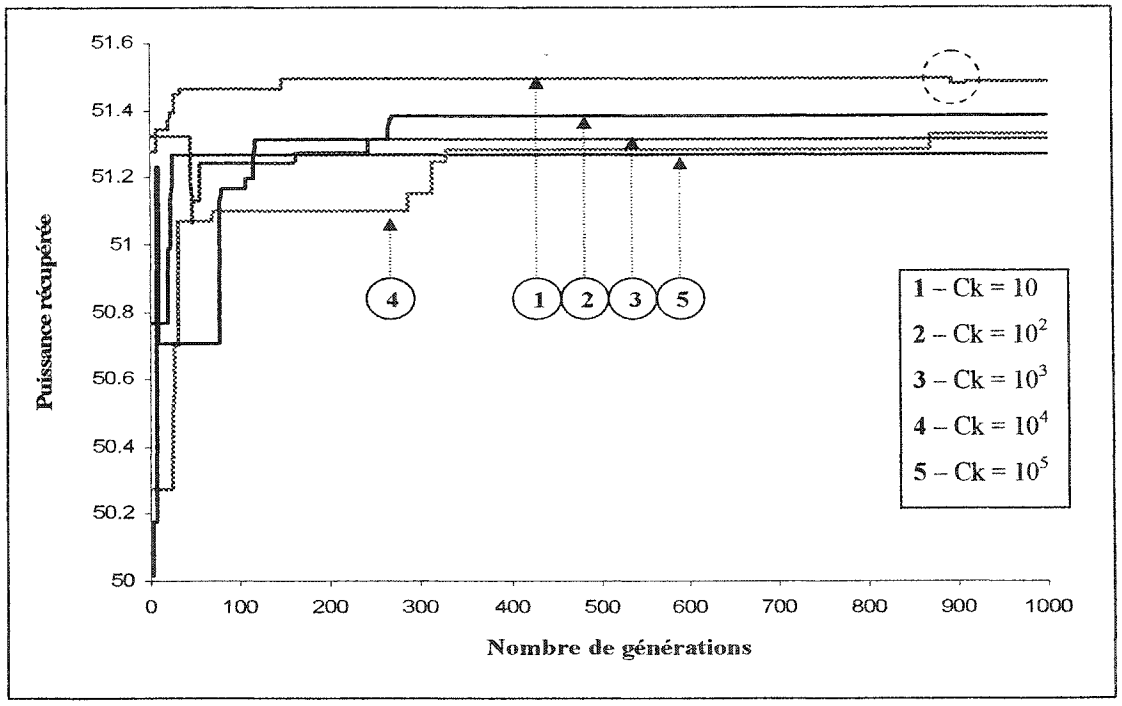


Figure 6.6 : Influence du coefficient de pénalisation sur la convergence

Une observation des courbes de la Figure 6.6. nous renseigne que pour de fortes valeurs de Ck, la convergence vers l'optimum est lente. Il faudra alors plusieurs générations avant que la courbe pour $Ck = 10^5$ s'élève au même niveau que la courbe pour $Ck = 10$. Cela est essentiellement dû au fait que la forte pénalisation limite l'exploration de

l'espace de recherche. Cependant, des faibles valeurs de Ck peuvent entraîner une convergence prématurée de la solution. Nous pouvons le constater avec la courbe pour Ck = 10. Cette courbe évolue rapidement vers un maximum et reste constante pendant plusieurs générations puis baisse subitement (partie encerclée) après 800 générations, ce qui ne devrait pas être le cas, étant donné que nous utilisons l'élitisme dans notre algorithme. Le maximum initialement indiqué était donc obtenu avec une violation de la contrainte de température minimale d'approche. Une valeur de Ck supérieure ou égale à 1000 permet d'avoir des résultats sans violation de la contrainte de température pour ce problème.

Tous les lancements effectués pour ce problème ont permis de trouver la même structure du REC. Le chromosome optimal obtenu est codé par **[1 0 2 2 1 1].** Le réseau d'échangeurs de chaleur équivalent correspond à celui présenté à la Figure 6.3. Nous présentons au Tableau 6.6 les résultats de la distribution des charges de trois lancements de l'algorithme de recherche. La solution de référence est également présentée dans ce tableau.

Tableau 6.6 : Solutions par l'approche G/G

Variables		Solution (G/G) (MW)			Solution de Référence (MW)
		1er lancement	2ème lancement	3ème lancement	
Échangeurs de chaleur	X_1	11.88	7.15	6.78	6.5
	X_2	12.12	16.83	17.17	17.5
	X_3	12.5	12.5	12.5	12.5
	X_4	7.05	7.74	7.5	8
	X_5	7.95	7.27	7.54	7
Utilités froides	X_6	4.62	9.34	9.70	10
	X_7	5.38	0.67	0.33	0
Utilités chaudes	X_8	0.95	0.27	0.55	0
	X_9	6.55	7.23	6.97	7.5
Puissance récupérée (Q_{REC}) (MW)		51.5	51.49	51.49	51.5

Les puissances récupérées dans les trois lancements sont comparables à la solution de référence. Le taux de récupération est supérieur à 98 %, ce qui est une solution satisfaisante.

Deux paramètres importants caractérisent les algorithmes génétiques. Il s'agit des probabilités de croisement et de mutation. Les valeurs de ces paramètres doivent être fixées par l'opérateur de manière à assurer aussi bien l'exploitation des meilleurs solutions que l'exploration de l'espace des solutions. Nous présentons à la Figure 6.7, des courbes de variation de la valeur moyenne de la pénalisation, illustrant l'effet de la mutation sur la dégradation de la qualité des chromosomes. Nous rappelons que la pénalisation du chromosome pour chaque violation de la contrainte de température minimale d'approche a été définie par : $P = Ck \times \sum_{i=1}^{N_{inéq}} \Phi_i$ (voir l'équation 4.23). Les courbes de la Figure 6.7 obtenues pour la même probabilité de croisement (Pc = 70%), mais avec des probabilités de mutation différentes, montrent que le niveau de pénalisation est plus faible pour une faible mutation (probabilité de mutation Pm = 5%). Par contre pour une forte probabilité de mutation (Pm = 30%), la pénalisation est amplifiée.

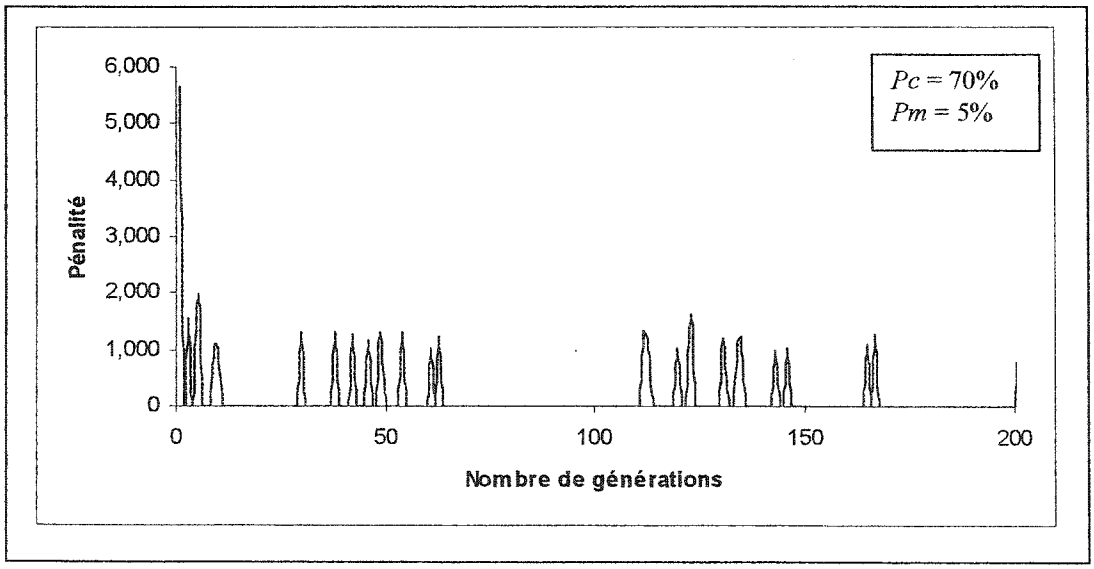


Figure 6.7 :Effets de la mutation sur le niveau de pénalisation

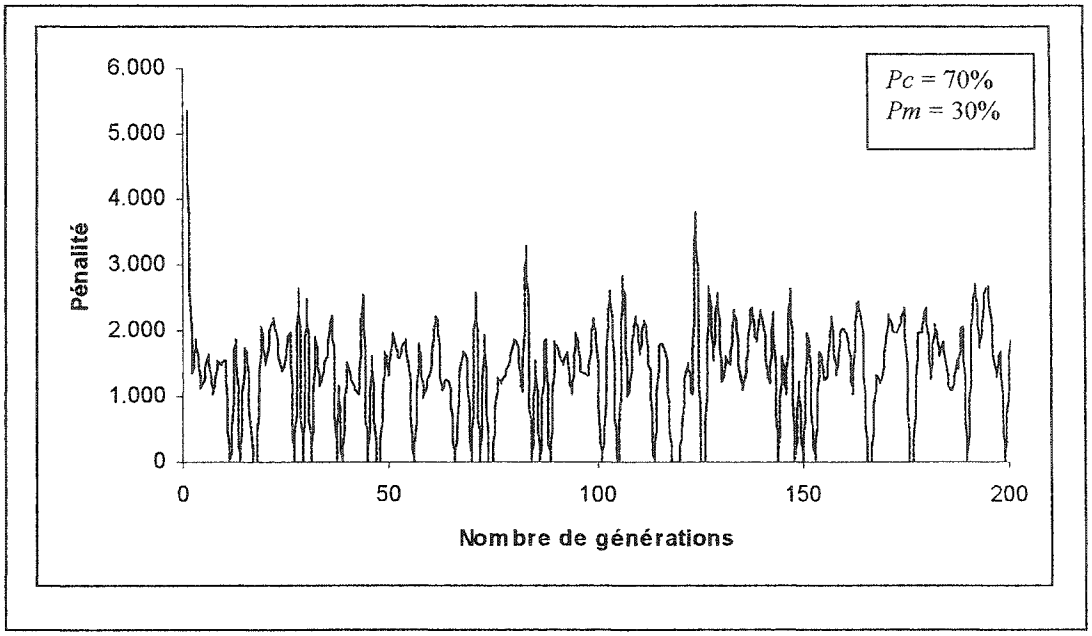


Figure 6.7 (suite) : Effets de la mutation sur le niveau de pénalisation

En général, la probabilité de mutation doit être fixée à une faible valeur, sinon la recherche risque de devenir aléatoire. Différents essais nous ont permis de constater qu'une valeur comprise entre 1% et 10% donne des résultats satisfaisants.

Les courbes de pénalisation de la Figure 6.8 illustrent l'effet du croisement sur les chromosomes. Pour la même probabilité de mutation,(Pm = 30%), les niveaux de pénalisation obtenus avec une probabilité de croisement Pc = 5% et Pc = 70% sont à peu près similaires. Les variations de la probabilité de croisement n'affectent donc pas substantiellement les chromosomes.

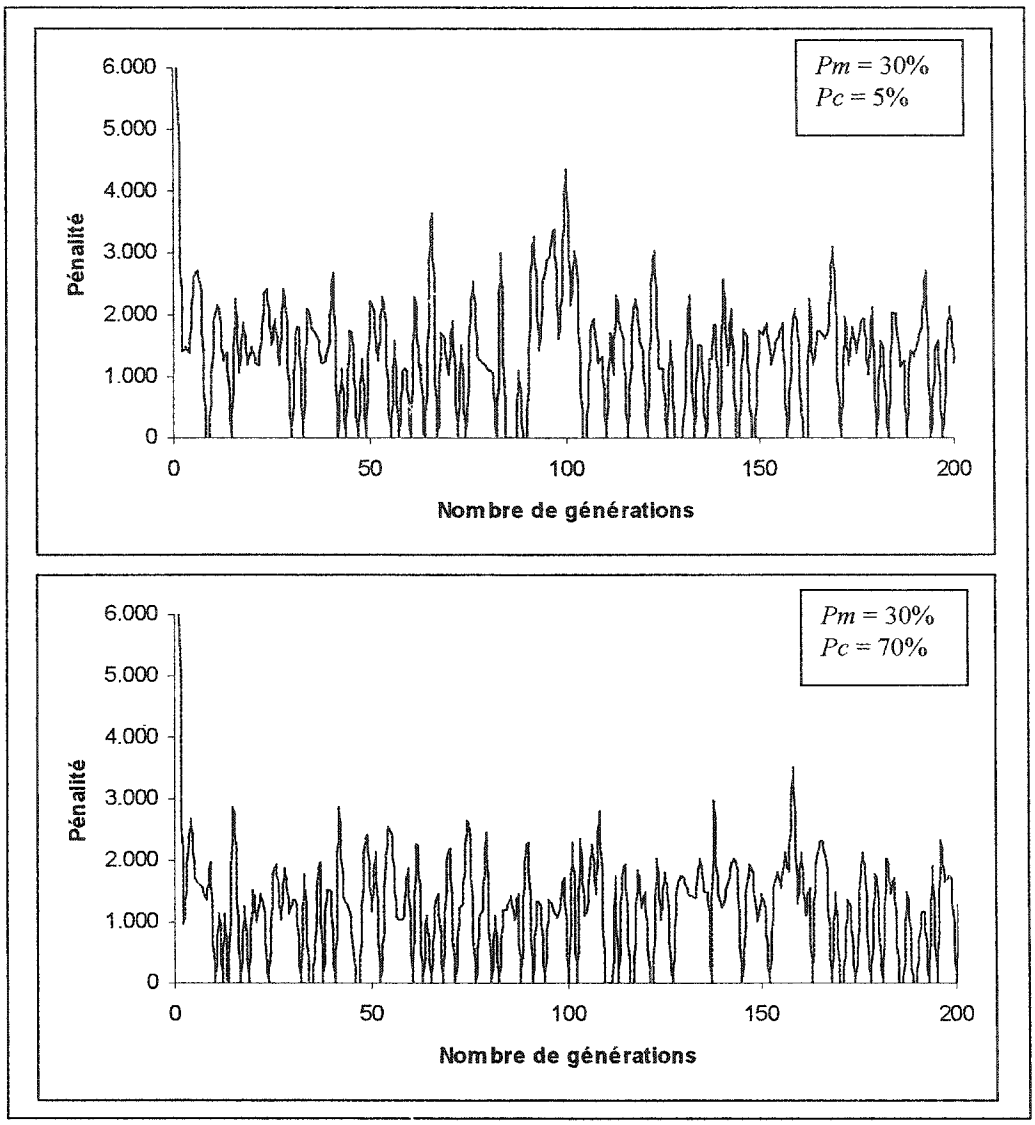


Figure 6.8 : Effets du croisement sur le niveau de pénalisation

6.2 *Étude de cas 2*

Le second exemple que nous traitons est le problème bien connu de l'usine d'aromates qui a été expérimenté dans les travaux de Lewin et al.(1998.a). Le problème comporte 4 courants chauds et 5 courants froids dont les caractéristiques sont rapportées dans le Tableau 6.7.

Tableau 6.7 : Caractéristiques des courants : Étude de cas 2

Courants	Température Initiale T^S (°C)	Température finale T^T (°C)	Charge thermique (MW)	Capacité calorifique CP (MW/°C)
H1	327	40	28.7	0.1
H2	220	160	9.6	0.16
H3	220	60	9.6	0.06
H4	160	45	46	0.4
C1	100	300	20	0.1
C2	35	164	9.03	0.07
C3	85	138	18.55	0.35
C4	60	170	6.6	0.06
C5	140	300	32	0.2

La température minimale d'approche utilisée est ΔT_{min} = 26°C. Basée sur la technique du pincement, la puissance maximale récupérée est Q_{REC} = 61.14 MW, réalisée par la division du courant chaud 4 (Linnhoff et Ahmad., 1990) [43]. Cela requiert 11 échangeurs de chaleur, 3 refroidisseurs et 3 réchauffeurs.

Les paramètres utilisés pour le lancement de l'algorithme génétique sont les suivants :

Taille de la population : 100

Nombre de générations : 1000

Probabilité de croisement : 70%

Probabilité de mutation : 10%

Coefficient de pénalisation Ck = 1000.

6.2.1 Résultats de test pour un niveau

En fixant le nombre de niveaux à 1 ($N_v = 1$), la distribution des charges obtenue est présentée au Tableau 6.8. La puissance récupérée Q_{REC} = 56.545 MW, ce qui représente 92% de la puissance disponible (61.14MW).

Tableau 6.8 : Solutions par l'approche G/G ($N_v = 1$)

Variables		Solution G/G (MW)	Solution de référence (MW)
Échangeurs de chaleur	X_1	9.398	9.4
	X_2	7.468	7.35
	X_3	17.149	17.15
	X_4	6.598	6.6
	X_5	15.931	16.1
Utilités froides	X_6	5.301	5.25
	X_7	0.202	0.2
	X_8	3.002	3.0
	X_9	28.851	28.85
Utilités chaudes	X_{10}	10.602	10.6
	X_{11}	1.562	1.68
	X_{12}	1.401	1.4
	X_{13}	0.002	0.0
	X_{14}	16.069	15.9
Puissance récupérée (Q_{REC}) (MW)		56.545	56.6

Le chromosome optimal du REC généré par l'algorithme est codé par **[2 1 4 3 1]**. Le décodage de ce chromosome donne la structure du REC présenté à la Figure 6.9. Cette structure est identique à celle obtenue par Lewin et al. (1998a) [4], pour le même nombre de niveaux N_v = 1. Sur cette figure, les puissances sont indiquées par les nombres soulignés.

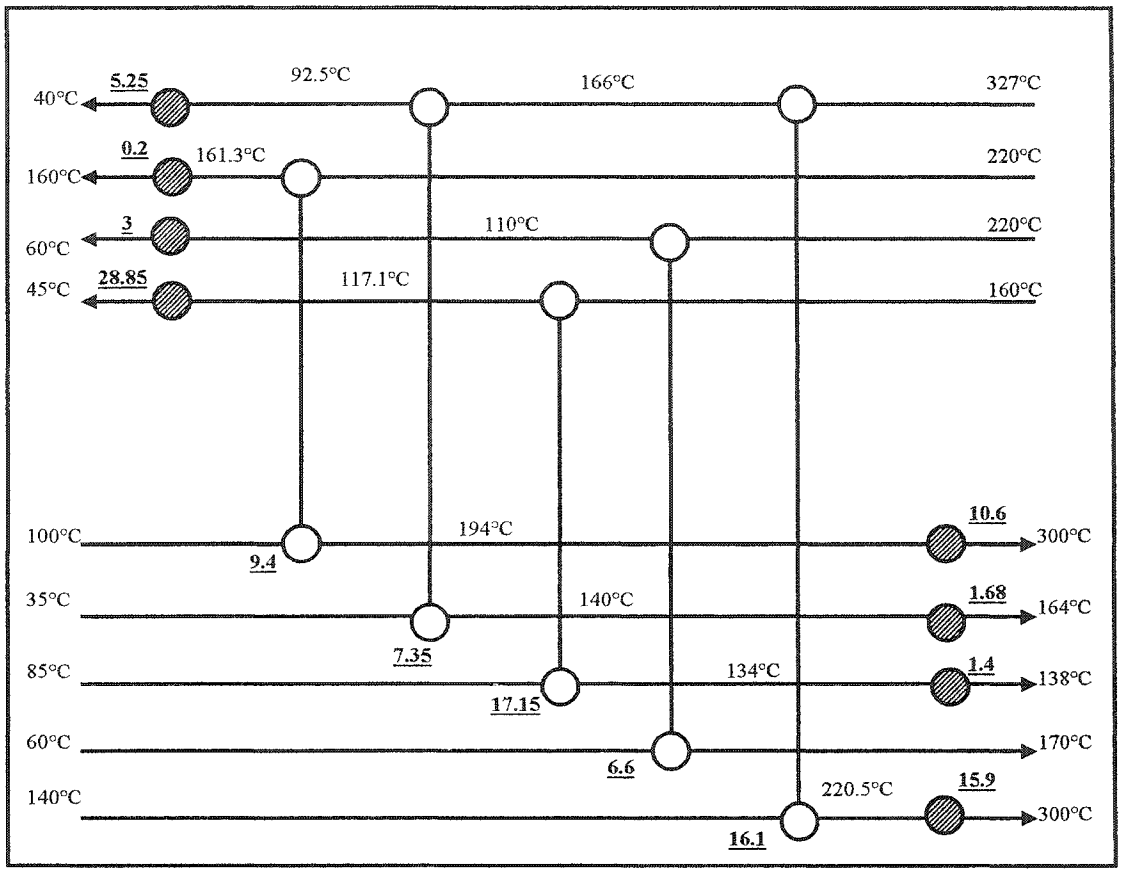


Figure 6.9 : Configuration du meilleur REC pour $N_v = 1$ (étude de cas 2)

6.2.2 Résultats de test pour trois niveaux

Pour améliorer la récupération de chaleur, il est nécessaire d'augmenter le nombre de niveaux. En fixant le nombre de niveaux à 3, un chromosome de REC aura une longueur de 15 gènes c'est-à-dire un maximum de 15 échangeurs de chaleur admissibles dans le réseau. Le meilleur chromosome de REC que nous avons obtenu est codé par :

[0 1 4 3 2 1 0 4 1 2 1 3 0 0 2]. Cela représente un total de 11 échangeurs de chaleur qui a été utilisé. Les résultats de la distribution de charges obtenus sont présentés dans le Tableau 6.9.

Tableau 6.9 : Solutions par l'approche G/G ($N_v = 3$)

Variables		Solution (G/G) (MW)	Solution de référence (MW)
Échangeurs de chaleur	X_1	5.333	4.55
	X_2	10.956	9.2
	X_3	4.584	5.93
	X_4	3.698	19.43
	X_5	8.530	0.96
	X_6	6.187	6.71
	X_7	1.262	2.11
	X_8	2.032	1.41
	X_9	9.186	2.64
	X_{10}	3.456	0.67
	X_{11}	2.909	6.96
Utilités froides	X_{12}	4.388	4.05
	X_{13}	0.960	0.0
	X_{14}	1.560	1.56
	X_{15}	28.858	27.71
Utilités chaudes	X_{16}	2.284	0.57
	X_{17}	0.241	2.11
	X_{18}	1.408	0.0
	X_{19}	0.754	0.0
	X_{20}	23.360	22.93
Puissance récupérée ($QREC$) (MW)		58.134	60.58

La puissance récupérée est Q_{REC} = 58.134 MW, soit un taux de récupération de 95%.

La topologie que nous avons obtenue est différente de celle présentée dans les travaux de Lewin et al.,(1998a) [4], où le meilleur chromosome est codé selon la séquence des nombres suivants : **[0 1 4 3 0 1 4 4 0 3 0 4 2 1 2]**. La puissance totale récupérée dans ce cas est Q_{REC} = 60.58 MW . La Figure 6.10 présente la configuration du réseau d'échangeurs de chaleur obtenu.

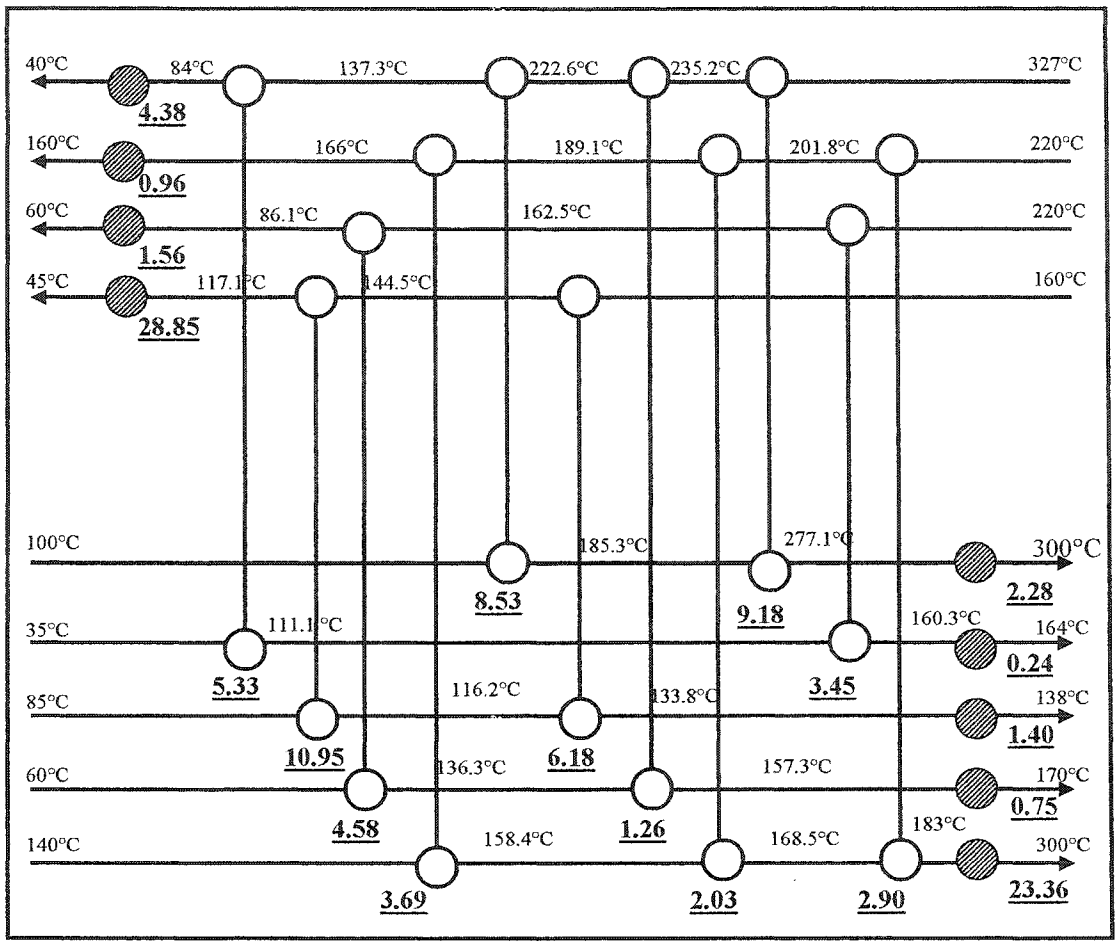


Figure 6.10 : Configuration du meilleur REC pour $N_v = 3$ (étude de cas 2)

6.3 Étude de cas 3

Le troisième exemple que nous traitons est un problème qui comporte 3 courants chauds et 4 courants froids. Ce problème est traité par la méthode du pincement (Site Internet de 'Environmental Technology Network') [29], avec une température minimale d'approche fixée à ΔT_{min} = 10°C. Pour cette température minimale d'approche, la puissance maximale disponible pour la récupération est Q_{REC} = 2364.5 KW, le besoin minimum en utilités chaudes est QH_{min} = 1841.5 KW et le besoin minimum en refroidissement est QC_{min} = 766 KW.

Les caractéristiques du problème sont reportées au Tableau 6.10.

Tableau 6.10 : Caractéristiques des courants : Étude de cas 3

Fluide	Température initiale T^S (°C)	Température finale T^T (°C)	Charge thermique (KW)	Capacité calorifique CP (KW/°C)
H1	200	40	2288	14.3
H2	85	75	457	45.7
H3	140	125	385.5	25.7
C1	20	105	986	11.6
C2	50	75	560	224
C3	75	200	312.5	2.5
C4	145	220	2347.5	31.3

Les paramètres suivants ont été utilisés pour le lancement de l'algorithme :

- Nombre de niveaux N_v = 2
- Probabilité de croisement Pc = 70%
- Probabilité de mutation Pm = 10%
- Coefficient de pénalisation Ck = 1000
- Taille de la population = 100

Les résultats de la distribution des charges sont reportés dans le Tableau 6.11. Cette distribution des charges est obtenue pour un REC codé par le chromosome **[0 3 0 0 1 2 1 1]**. Ce codage est identique à celui de la solution de référence. La structure du réseau d'échangeurs de chaleur équivalente est présentée à la Figure 6.11. Elle comporte 5 échangeurs de chaleur, 3 utilités froides et 2 utilités chaudes.

Tableau 6.11 : Solutions par l'approche G/G ($N_v = 2$)

Variables		Solution G/G (KW)	Solution de référence (KW)
Échangeurs de chaleur	X_1	348.255	385.5
	X_2	985.933	986
	X_3	211.736	174.5
	X_4	175.043	175
	X_5	643.106	643.5
Utilités froides	X_6	483.919	483.5
	X_7	245.264	282.5
	X_8	37.245	0
Utilités chaudes	X_9	0	0
	X_{10}	0	0
	X_{11}	137.457	137.5
	X_{12}	1704.394	1704
Puissance récupéré (Q_{REC}) (KW)		2364.072	2364.5

La puissance récupérée Q_{REC} = 2364.072 KW, soit un taux de récupération presque 100%. Dans ce cas, la puissance totale de l'utilité chaude QH_{min} = 1841.851 KW (contre 1841.5 kW requise) et la puissance totale de l'utilité froide QC_{min} = 766.428 KW (contre 766 KW requise).

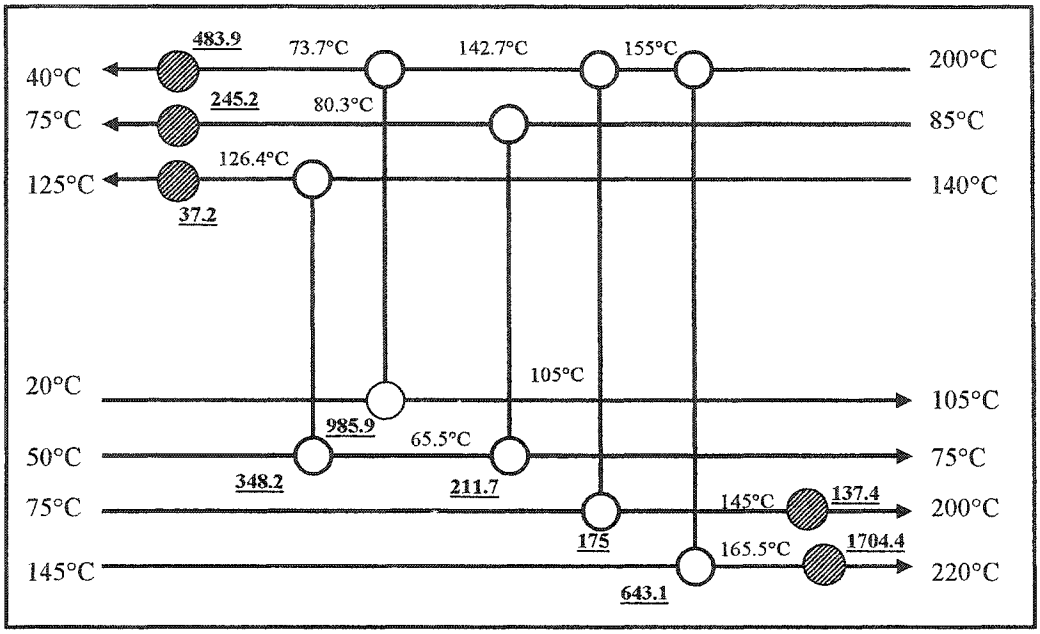


Figure 6.11 : Structure optimale du REC (étude de cas 3)

6.4 Conclusion

Nous avons présenté dans ce chapitre quelques exemples de problèmes pour lesquels la récupération de chaleur a été optimisée. Les solutions obtenues sont comparables à celles présentées dans la littérature. Pour ces problèmes, nous avons adopté des valeurs de coefficient de pénalité, telles que les contraintes de température soient respectées aux bornes de tous les échangeurs de chaleur. Par un choix judicieux du coefficient de pénalité, il est possible de contrôler la violation de la contrainte de température minimale d'approche pour améliorer la récupération de chaleur.

CHAPITRE VII

7 CONCLUSION

Dans cet ouvrage, l'application des algorithmes génétiques à la synthèse de réseaux d'échangeurs de chaleur a été démontrée sur différents problèmes. Nous avons utilisé une procédure, que nous appelons approche G/G. Dans cette procédure, la détermination de la structure optimale du réseau d'échangeurs de chaleur ainsi que l'optimisation des charges des échangeurs de chaleur du réseau ont été réalisées par l'application stricte des algorithmes génétiques. Les résultats obtenus de l'application de la méthode sur quelques problèmes de test témoignent de la validité de la méthode. Nous avons également expérimenté une approche mixte combinant les algorithmes génétiques et la méthode du simplexe que nous avons appelée approche G/S. Dans cette procédure, la structure du réseau est déterminée suivant le principe des algorithmes génétiques, tandis que l'optimisation de la distribution des charges est assurée par la méthode du simplexe. Cette méthode a été appliquée au problème de l'étude de cas 1, pour démontrer son principe de fonctionnement.

7.1 Les performances de la méthode

Les différents essais qui ont été réalisés ont permis d'apprécier la validité de l'approche G/G pour des réseaux de taille modérée. L'AG est capable de trouver très rapidement les régions les plus prometteuses de l'espace des solutions et, dans la majorité des cas, la meilleure topologie du REC a été identifiée avec une distribution optimale de la charge. Pour un même problème, chaque lancement de l'algorithme permet de trouver une configuration de REC, différente de celles des autres lancement ,ou identique, avec des distributions de charge variant d'un lancement à l'autre. Ceci est un avantage, car les

AG peuvent servir à générer un nombre important de structures initiales de REC qui optimisent la récupération de chaleur. Ces structures peuvent être évaluées par la suite par rapport au nombre d'unités, à la surface totale d'échange, ou à l'énergie récupérée afin de retenir la structure la plus économique.

Les algorithmes génétiques présentent une grande flexibilité de manipulation, et diverses contraintes peuvent être incluses dans le modèle de simulation. Des restrictions d'appariement peuvent donc être prises en compte, qu'il s'agisse d'appariements forcés ou interdits. Par la méthode de la fonction pénalité, nous transformons un problème avec des contraintes, en un problème sans contraintes par l'incorporation d'une fonction de pénalisation dans le modèle. Cette possibilité est un avantage pour les algorithmes génétiques par rapport à la méthode du simplexe qui requiert la manipulation de tableaux de grandes dimensions. De plus, la nature combinatoire des problèmes de synthèse des réseaux d'échangeurs de chaleur, est beaucoup plus difficile à traiter par la méthode du simplexe qui exige au moins une solution faisable dans la population initiale. Ceci n'est pas le cas de notre approche (l'approche G/G) qui, à partir d'une population initiale infaisable, peut toujours évoluer pour la recherche d'une solution optimale en amélioration la population.

Par un choix judicieux du coefficient de pénalité Ck, il est possible d'autoriser, si l'on souhaite, une légère violation de la température minimale d'approche , pour permettre de récupérer le maximum de chaleur.

7.2 Le réglage des paramètres de l'algorithme génétique

Plusieurs facteurs influencent les performances de l'algorithme génétique : il s'agit notamment des probabilités de croisement et de mutation, du coefficient de pénalisation (méthode de la fonction pénalité), de la taille de la population. Les essais sur plusieurs exemples ont montré qu'une probabilité de croisement de 70 à 80% et une mutation de 5 à 10% étaient de bons compromis pour assurer une bonne convergence de la solution.

Même s'il n'existe pas de règles strictes définissant la taille de la population, nous pouvons constater qu'elle doit avant tout tenir compte de la taille du chromosome. La précision des résultats dans un codage binaire dépendant de la taille des gènes (donc du chromosome), l'obtention d'une solution avec une bonne précision nécessitera un chromosome de taille élevée, impliquant un espace de recherche large. Dans ce cas, il faudra choisir la taille de la population assez grande pour explorer suffisamment l'espace des solutions. Cependant, il faut noter que la taille de la population n'influence plus sensiblement les solutions au-delà d'une certaine valeur. Il faut alors trouver une taille optimale utilisant peu de ressources informatiques, pour assurer une bonne performance de l'algorithme.

Le réglage du coefficient de pénalisation (*Ck*) utilisé dans la fonction pénalité doit être optimisé pour chaque type de problème traité, ce qui constitue une part importante du travail de l'utilisateur. Le constat que nous faisons est que ce paramètre est adapté à la topologie du REC, et donc de la complexité du problème. Dans la phase de recherche de la topologie optimale du REC, le coefficient de pénalisation *Ck* doit être fixé à une valeur faible (pression de sélection moins rigoureuse) pour permettre l'exploration de tout l'espace de recherche. En général, la valeur optimale de *Ck* comprise entre 10 et 20 a permis d'avoir une topologie satisfaisante pour la plupart des problèmes que nous avons traités . Par contre, lors de la phase de recherche de la distribution optimale des charges, la valeur de *Ck* doit être adaptée à chaque problème donné. Pour les problèmes que nous avons traités, une valeur de Ck supérieure à 10^3 donne une marge assez suffisante pour éviter de violer la contrainte de température minimale d'approche. Cependant, celle-ci ne doit pas dépasser 10^5 au risque de ralentir considérablement la vitesse de convergence.

7.3 Insuffisances de la méthode

Le modèle que nous avons utilisé permet d'optimiser la récupération de chaleur, mais n'inclut pas le contrôle du nombre d'utilités utilisé. Des besoins résiduels, souvent très infimes, apparaissent sur les courants et doivent être satisfaits par des utilités. Les structures générées par notre algorithme contiennent ainsi généralement une utilité sur chaque courant. Elles doivent donc être considérées comme des solutions initiales qui nécessitent d'être raffinées pour limiter le nombre d'utilités.

Le codage binaire de la distribution des charges peut être également un facteur de limitation des performances de l'algorithme, surtout pour des chromosomes comportant plusieurs gènes (problème à plusieurs variables). Si une bonne précision est requise pour les solutions, la taille du chromosome utilisé sera grande. Or dans un codage binaire, l'espace de solutions est d'autant plus large que la taille du chromosome est grande. Pour un problème comportant plusieurs échangeurs de chaleur et pour lequel une bonne précision des solutions est exigée, l'AG peut faillir à trouver des solutions acceptables si la population est insuffisante ou si le nombre de générations est faible. Beaucoup d'espaces mémoire sont requises pour traiter des problèmes complexes, donc exigent un outil informatique performant.

7.4 Propositions d'amélioration

Le modèle que nous avons développé pour la synthèse du REC ne prend pas en compte la division des courants. Il serait intéressant d'investiguer dans ce sens pour permettre à l'algorithme de tenir compte de ces cas, en vue d'améliorer la récupération de chaleur. D'autres investigations doivent être également menées pour déterminer un type de codage offrant des facilités de maniabilité comme le codage binaire, mais qui réduisent la taille de l'espace de recherche.

RÉFÉRENCES

[1] BERNARD, P. A. (1995). *L'audit énergétique*. Paris : Les éditions d'organisation, pp.15-18

[2] LINNHOFF, B. et HINDMARSH, E. (1983). *The Pinch Design Method for Heat Exchanger Networks*. Chem. Engng. Sci., vol.38, no. 5, pp 745-763.

[3] HOLLAND, J. H. (1975). *Adaptation in Natural and Artificial Systems: an Introduction Analysis with Applications to Biology, Control and Artificial Intelligence*. Ann Arbor: University of Michigan Press, 183p.

[4] LEWIN, D. R., WANG, H. et SHALEV, O. (1998a). *A Generalized Method for HEN Synthesis using Stochastic Optimization.(I) General Framework and MER Optimal Synthesis*. Comput. Chem. Engng., vol. 22, no.10, pp. 1503-1513.

[5] GUNDERSEN, T. et NAESS, L. (1988). *The Synthesis of Cost Optimal Heat Exchanger Network- an industrial review of the state of the art*. Comput. Chem. Engng., vol. 12, no.6, pp. 503-530.

[6] UMEDA, T., ITOH, J. et SHIROKO, K. (1978). *Heat Exchange System Synthesis*. Chem. Engng. Progr., vol. 74, pp. 70-76.

[7] LINNHOFF, B. et VREDEVELD D. R. (1984). *Pinch Technology has come of Age*. Chem. Engng. Progr., vol. 80, pp. 33-40.

[8] HUANG, F. et ELSHOUT, R. V. (1976). *Optimizing the Heat Recovery of Crude Units*. Chem. Engng. Prog. vol. 72, pp. 68-74.

[9] LINNHOFF, B. (1979). Thermodynamic Analysis in the Design of Process Networks. Comput. Chem. Engng., vol. 3, no.1-4, pp. 283-291.

[10] LINNHOFF, B. et al. (1982). *User Guide on Process Integration for the Efficient Use of Energy*. 1st edition, Oxford; New York: Pergamon Press, 247p.

[11] HOHMANN, E. C. (1971) *Optimum Networks for Heat Exchange*. PhD. Thesis, Department of Chemical Engineering, University of Southern California.

[12] BOLAND, D. et LINNHOFF, B. (1979). *The Preliminary Design of Networks for Heat Exchange by Systematic Methods*. Chem. Engr, no. 343, pp. 222-228.

[13] AHMAD, S. et SMITH, R. (1989). *Targets and Design for Minimum Number of Shells in Heat Exchanger Networks*. Chem. Engng. Res. Des., vol. 67, pp. 481-494.

[14] NISHIDA, N., STEPHANOPOULOS, G. et WESTERBERG, A. W. (1981). *A Review of Process Synthesis*. AIChE Jl, vol. 27, no. 3, pp. 321-351.

[15] TOWNSEND, D. W. et LINNHOFF, B. (1984). *Surface Area Targets for Heat Exchanger Networks*. IChemE 11th Annual Research Meeting on Heat Transfer, Bath, UK.

[16] TJOE, T. N. et LINNHOFF, B. (1986). *Using Pinch technology for Process Retrofits*. Chem. Engr., vol. 93, no. 8, pp. 47-60.

[17] YEE, T. F. et GROSSMANN, I. E. (1990). *Simultaneous Optimization Models for Heat Integration- II. Heat Exchanger Networks Synthesis*. Comput. Chem. Engng., vol. 14, no. 10, pp. 1165-1184.

[18] CIRIC, A. R. et FLOUDAS, C. A. (1991). *Heat Exchanger Network Synthesis without Decomposition*. Comput. Chem. Engng., vol. 15, no. 6, pp. 385-396.

[19] ZHU, X. X., O'NEILL, B. K., ROACH, J. R. et WOOD, R. M. (1995). *A Method for Automated Heat Exchanger Network Synthesis Using Block Decomposition and Non-linear Optimization*. Chem. Engng. Res. Des., Part A, vol. 73, no. 11, pp. 919-930.

[20] GOLDBERG, D. E. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Reading, Mass: Addison-Wesley, 412p.

[21] ANDROULAKIS, I. P. et VENKATASUBRAMANIAN, V. (1991). *A Genetic Algorithm Framework for Process Design and Optimization*. Comput. Chem. Engng, vol. 15, no. 4, pp. 217-228.

[22] STAIR, C. et FRAGA, E. S. (1995). *Optimization of Unit Operating Conditions for Heat Integrated Process Using Genetic Algorithms*. Proceedings of the 1995 IChemE Research Event, pp. 95-97.

[23] LEWIN, D. R. (1998b). *A Generalized Method for HEN Synthesis using Stochastic Optimization: (II) The Synthesis of Cost-Optimal Networks*. Comput, Chem. Engng., vol. 22, no. 10, pp. 1387-1405.

[24] SADIK, K., HONGTAN, L. (2002). *Heat Exchangers: selection, rating, and thermal design*. 2nd ed., Boca Raton: CRC Press, 501p.

[25] INCROPERA, F. P. et DEWITT, D. P. (2001). *Fundamentals of Heat and Mass Transfer*. 5th ed., New York: Willey, 981p.

[26] FLOWER, J. R. et LINNHOFF, B. (1980). *A Thermodynamic Combinatorial Approach to the Design of Optimum Heat Exchanger Networks.* AIChE Jl, vol. 26, pp. 1- 9.

[27] SMITH, R. (1995). *Chemical Process Design.* New York: McGraw-Hill, 459 p.

[28] THE CHEMICAL ENGINEERS' RESSOURCE PAGE. Pinch Technology: Basics for the Beginners. In Site de *The Chemical Engineers' Resource Page*. [En ligne] http://www.cheresources.com/pinchtech3.shtml . (Page consultée le 9 Octobre 2002)

[29] THE ENVIRONMENTAL TECHNOLOGY NETWORK. PinchTechnology. In Site de The Environmental Technology Network. [En ligne]. http://www.envirotechnet.com/pinchtechnology.com. (Page consultée le 10 Novembre 2002)

[30] GRIMES, L. E. (1980). *The Synthesis and Evolution of Networks of Heat Exchanger that feature the Minimum Number of Units.* M.Sc. Thesis, Carnegie-Mellon University, Pittsburgh.

[31] HOHMANN, E. C. (1971). *Optimum Networks for Heat Exchange.* Ph.D. Thesis, University of Southern California.

[32] LINNHOFF, B., Mason, D. R. et Wardle, I. (1979). *Understanding Heat Exchanger Networks.* Comput. Chem. Engng, vol. 3, pp. 295-302.

[33] POLYTECH'LILLE. Ressources en ligne : Optimisation et algorithmes génétiques. In *Site de POLYTECH'LILLE.* [En ligne] http://www.polytech-lille.fr/~vmagnin/coursag/algo_evo.html (Page consultée le 15 Juillet 2003)

[34] WIKIPÉDIA. Encyclopédie : Sciences naturelles et mathématiques. In Site de *WIKIPÉDIA*. [En ligne]. http://fr.wikipedia.org/wiki/S%E9lection_naturelle. (Page consultée le 9 Avril 2003).

[35] MICHALEWICZ, Z. (1996). *Genetic Algorithms + data structures = Evolution Programs*. 3rd rev. and extended ed., Berlin: Springer, 387p.

[36] MELANIE, M. (1996). *An Introduction to Genetic Algorithms*. Cambridge, Mass: MIT Press, 205p.

[37] BÄCK, T. (1996). *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. New York: Oxford University Press, 314p.

[38] MITSUO, G., CHENG, R. (1997). *Genetic algorithms and Engineering Design*. New York: Wiley, 411p.

[39] YOGESH, J. (1998). *Design and Optimization of Thermal System*. Montréal: McGraw-Hill Co., 626 p.

[40] LINNHOFF, B. et FLOWER, J. R. (1978). *Synthesis of Heat Exchanger Networks: I. Systematic Generation of Energy Optimal Networks*'. AIChE Jl., vol. 24, pp. 633-642.

[41] BAILLARGEON, G. (1996). *Programmation Linéaire Appliquée: Outil d'Optimisation et d'Aide à la Décision*. Trois-Rivières, Québec : Ed. SMG, 431p.

[42] KOLMAN, B. et BECK, R. E. (1980). *Elementary Linear Programming with Applications*. New York: Academic Press, 399p.

[43] LINNHOFF, B. et AHMAD, S. (1990). *Cost Optimum Heat Exchanger Networks. 1: Minimum Energy and Capital using Models for Capital Cost*. Comput. Chem. Engng., vol. 14, no 7, pp. 729-750.

ANNEXES

A-1 Description des sous programmes

Programme PRINCIPAL
Ce programme principal gère l'ensemble des sous-programmes, chaque sous-programme étant affecté à une tâche précise dans le processus de recherche de l'optimum des solutions par les algorithmes génétiques.

Sous-programme VARIABLES
Il enregistre les différentes données de l'algorithmes qui sont : le nombre de générations, la taille de la population, le nombre de courants chauds et froids, le nombre de niveaux pour le codage de la structure du réseau d'échangeurs de chaleur.

Sous-programme PARAMETRES
Il enregistre les caractéristiques des courants : La température minimale d'approche, les capacités calorifiques, les températures initiales et finales et le choix de l'élitisme ou non.

Sous-programme COMPOSITES
Il utilise la méthode du PTA pour déterminer la puissance maximale récupérable et les puissances des utilités chaudes et froides. Il détermine également les températures de pincement des courants chauds et froids.

Sous-programme POPULATION
Il génère la population initiale des réseaux d'échangeurs de chaleur.

Sous-programme MATRIX
Il établit une matrice de connexions des échangeurs de chaleur sur les différents courants.

Sous-programme DISTRIBUTION
Il génère la population initiale des distributions de charges.

Sous-programmes PUISSANCE ou PUISSANCE2
Il décode le chromosome de distribution de charge et calcule la valeur réelle de la puissance de chaque échangeur de chaleur. Un coefficient de réparation r permet d'ajuster la borne supérieure des variables de sorte à respecter le principe de conservation d'énergie sur chaque courant.

Sous-programme TEMPÉRATURE ou TEMPÉRATURE2
Il calcule la température à l'entrée et à la sortie de chaque échangeur de chaleur pour vérifier le respect de la température minimale d'approche.

Sous-programme PÉNALITÉ ou PENALITÉ2
Il calcule la pénalisation pour toute violation de la contrainte de température minimale d'approche aux bornes de chaque échangeur de chaleur.

Sous-programme RESTRICTION
Il permet de réduire l'espace à explorer lorsque la solution s'approche de l'optimum.

Sous-programmes SELECTION ou SÉLECTION2
Il crée le bassin de reproduction (mating pool), c'est-à-dire la population intermédiaire (constituée de chromosomes assez bons) à partir de laquelle seront appliqués les opérateurs de croisement et de mutation pour reproduire la future génération.

Sous-programme ÉLITISME ou ÉLITISME2
Il sélectionne les deux meilleurs chromosomes de la population de chaque génération pour les reporter directement dans la future génération et permet ainsi de préserver les meilleurs gènes qui pourraient être dégradés lors des opérations de croisement et de mutation.

Sous-programme CROISEMENT ou CROISEMENT2
Il assure les opérations de croisement et de mutation des chromosomes sélectionnés de façon aléatoire dans la population.

Sous-programme CONVERSION
Il convertit en base 10, le codage binaire du chromosome de distribution de charge.

Sous-programme PINCEMENT
Il divise le système en deux parties : au-dessus et au-dessous du pincement, en indiquant les courants qui figurent dans chaque section et leurs caractéristiques.

Sous-programme MATRICE
Il établit le tableau initial du simplexe, comprenant les coefficients de la fonction objectif, les contraintes égalité et les contraintes inégalité.

Sous-programme PHASE2
Il permet de trouver une solution initiale faisable pour la procédure du simplexe dans le cas où il n'est pas possible d'exécuter directement le simplexe.

Sous-programme SIMPLEX
Il utilise la procédure du simplexe pour déterminer la valeur optimale de la fonction objectif ainsi que les valeurs des paramètres à optimiser.

Sous-programme SOLUTION
Il donne les résultats obtenus par l'algorithme d'optimisation. Ces résultats concernent la structure optimale, la distribution des charges, la puissance récupérée, les besoins en utilités chaudes et froides, les différences de températures aux extrémités des échangeurs de chaleur.

A-2 Principaux paramètres utilisés

Nkey	: Nombre de courants chauds
Key	: Nombre de courants froids
Level	: Nombre de niveaux
Nunit	: Nombre total d'unités
HEN	: Taille de la population de réseaux d'échangeurs de chaleur
Iteration	: Nombre de générations
NExch	: Nombre d'échangeurs de chaleur admissible dans le réseau
Equ	: Nombre total des courants chauds et froids
deltaT	: Température minimale d'approche
Cp	: Capacité calorifique des courants
Te	: Température initiale du courant
Ts	: Température final du courant
RHS	: Charge thermique du courant
Tpinh	: Température de pincement des courants chauds
Tpinc	: Température de pincement des courants froids
QHneed	: Besoins en utilités chaudes
QCneed	: Besoins en utilités froides
Qmax	: Puissance maximale récupérable
QREC	: Puissance réellement récupérée
QHmin	: Puissance
Epsin	: Différence de température entre les courants chaud et froid à l'extrémité chaude de l'échangeur de chaleur
Epso	: Différence de température entre les courants chaud et froid à l'extrémité froide de l'échangeur de chaleur
X	: Charge thermique d'un échangeur de chaleur ou d'une utilité
Pc	: Probabilité de croisement
Pm	: Probabilité de mutation

ANNEXE 1 : Code de l'algorithme d'optimisation (approche G/G)

```
************************************
*       Programme PRINCIPAL        *
************************************

PROGRAM PRINCIPAL
IMPLICIT NONE
REAL,DIMENSION(:),ALLOCATABLE      ::opdTo,opdTi,Fitessmod,eXi,OBJi
REAL,DIMENSION(:),ALLOCATABLE      ::Q,OBJ,epso,epsin,OptiX,Fitness
REAL,DIMENSION(:),ALLOCATABLE      ::Fitmod,Recup,Qi,QREC
REAL,DIMENSION(:),ALLOCATABLE      ::RHS,Te,Ts,Cp,Heatmin,BdTo,BdTi
REAL,DIMENSION(:,:),ALLOCATABLE    ::X,Y,Xi,Yi,dTo,dTi,BH,BC
REAL                               ::deltaT,som,Targ,somFit,cumulFit
REAL                               ::somQrec,moyQrec,OptiFit,OptiRecup
REAL                               ::cumulpenalty,BestFit
REAL                               ::QHneed,QCneed,Qmax,Tpinh,Tpinc
REAL                               ::penalty,Ck1,Ck2,Ck1ref,moypenalty
REAL                               ::QHmin,Goodfit,GoodRecup,GoodQrec
INTEGER                            ::J,k,L,M,I,Ia,T,ajust,NExch,equ
INTEGER                            ::TotGen,EchGen,Iter,Level,Nkey,Key
INTEGER                            ::cnt,El1,drap,Iteration,Nconex,cas
INTEGER                            ::HEN,Nbrota,rot,Nunit
INTEGER,DIMENSION(1)               ::N,N1,N2,N4,N5
INTEGER,DIMENSION(:),ALLOCATABLE   ::strH,strC,decichromo,chromBinR
INTEGER,DIMENSION(:),ALLOCATABLE   ::OptiBinchromo
INTEGER,DIMENSION(:),ALLOCATABLE   ::Parent,Binchromo,OptIntchromo
INTEGER,DIMENSION(:,:),ALLOCATABLE::Intchromo,Elite,EliteBin,Child
INTEGER,DIMENSION(:,:),ALLOCATABLE::Father,chromoBin
INTEGER,DIMENSION(:,:),ALLOCATABLE::Baby,Digichromo,chromBinT,Mother
CHARACTER(LEN=32)                  ::choix


CALL VARIABLES(Nkey,Key,Level,NExch,equ,Nunit,&
                    TotGen,EchGen,HEN,Iteration)

Iter=20
Nbrota=1000
Ck1=10.
Ck2=10000.
ALLOCATE()
ALLOCATE()
ALLOCATE(epso(1:NExch))
ALLOCATE(epsin(1:NExch))
ALLOCATE(Xi(1:Nunit,1:HEN),Y(1:Nunit,1:Iter),X(1:Nunit,1:Iter))
```

```
ALLOCATE(OptiX(1:Nunit),eXi(1:Nunit),Yi(1:Nunit,1:HEN))
ALLOCATE(Heatmin(1:HEN),QREC(1:HEN),Fitness(1:HEN))
ALLOCATE(Recup(1:HEN),Fitmod(1:4*HEN),OBJi(1:HEN))
ALLOCATE(Fitessmod(1:4*HEN),Parent(1:NExch),Q(1:Iter))
ALLOCATE(dTo(1:NExch,1:HEN),OBJ(1:Iter),Qi(1:HEN))
ALLOCATE(BdTo(1:NExch),BdTi(1:NExch),dTi(1:NExch,1:HEN))
ALLOCATE(strH(1:NExch),strC(1:NExch),opdTi(1:NExch),opdTo(1:NExch))
ALLOCATE(decichromo(1:NExch),BC(1:NExch,1:Key),RHS(1:equ))
ALLOCATE(Intchromo(1:NExch,1:HEN),OptIntchromo(1:NExch))
ALLOCATE(Elite(1:NExch,1:2),EliteBin(1:TotGen,1:2))
ALLOCATE(Child(1:NExch,1:HEN),Father(1:NExch,1:(4*HEN)))
ALLOCATE(Binchromo(1:TotGen),chromoBin(1:TotGen,1:Iter))
ALLOCATE(chromBinT(1:TotGen,1:HEN),chromBinR(1:TotGen))
ALLOCATE(OptiBinchromo(1:TotGen),Digichromo(1:TotGen,1:HEN))
ALLOCATE(Mother(1:TotGen,1:(4*HEN)),Baby(1:TotGen,1:HEN))
ALLOCATE(BH(1:NExch,1:Nkey),Te(1:equ),Ts(1:equ),Cp(1:equ),)

OPEN(1,FILE='QREC.TXT')
OPEN(2,FILE='RESULT.TXT')

CALL PARAMETRES(Nkey,equ,deltaT,Cp,Te,Ts,RHS,cas)
CALL COMPOSITES(deltaT,equ,Nkey,Cp,Te,Ts,Tpinh,&
                        Tpinc,QHneed,QCneed,Qmax)
CALL POPULATION(HEN,NExch,Nkey,Intchromo)

       PRINT*,'  Recherche en cours du meilleur réseau....'
     Goodfit=100E9
     GoodRecup=0.
     Parent(:)=1
*************************************************************************
!Recherche de la meilleure structure de réseau d'échangeur de chaleur!
*************************************************************************

Rotation:DO rot=1,Nbrota
     somFit=0.

 TITI:DO T=1,HEN
 Nconex=0
 DO k=1,NExch
    IF(Intchromo(k,T).NE.0)THEN
       Nconex=Nconex+1
    END IF
 END DO
 CALL MATRICE(T,Intchromo,HEN,NExch,Level,Nkey,key,strH,strC,BH,BC)
 cnt=0

   TIT0:DO I=1,Iter
  CALL DISTRIBUTION(cnt,Nunit,TotGen,EchGen,NExch,decichromo,Binchromo)
   cnt=cnt+1
  CALL PUISSANCE(BH,BC,I,Iter,T,NExch,HEN,Nunit,equ,Key,Nkey,&
                    EchGen,Qmax,RHS,decichromo,strH,strC,Q,Targ,X)
```

```
   chromoBin(:,I)=Binchromo
   Y(:,I)=X(:,I)
   CALL TEMPERATURE(I,Iter,T,strH,strC,HEN,NExch,BH,BC,Nkey,key,Nunit,&
                equ,Cp,Te,Ts,X,epso,epsin)
   CALL PENALITE(Nconex,T,HEN,I,X,Nunit,Iter,deltaT,&
                      epso,epsin,NExch,Penalty)
   OBJ(I)=Targ+Ck1*Penalty
   END DO TIT0

 N=MINLOC(OBJ)
 Fitness(T)=OBJ(N(1))
 Recup(T)=Q(N(1))
 somFit=somFit+Fitness(T)
 chromBinT(:,T)=chromoBin(:,N(1))
 END DO TITI

       N1=MINLOC(Fitness(:))
       IF(Fitness(N1(1)).LT.Goodfit)THEN
          Goodfit=Fitness(N1(1))
          GoodRecup=Recup(N1(1))
          chromBinR(:)=chromBinT(:,N1(1))
          Parent(:)=Intchromo(:,N1(1))
       END IF
       IF(rot.EQ.Nbrota)EXIT Rotation
 CALL RESTRICTION(rot,Nbrota,GoodRecup,Qmax,drap)
 CALL SELECTION(drap,rot,Nbrota,HEN,NExch,Fitness,Intchromo,&
                       Father,Fitessmod)
     Intchromo(:,1:HEN)=Father(:,1:HEN)
       Fitness(1:HEN)=Fitessmod(1:HEN)
 CALL ELISTISME(Nbrota,rot,T,Goodfit,Fitness,NExch,HEN,Intchromo,Elite)
 CALL CROISEMENT(Elite,Intchromo,rot,Nbrota,NExch,HEN,Nkey,&
                   somFit,Fitness,Child)
     Intchromo=Child
END DO Rotation

**************************************************************************
!Recherche de la meilleure distribution de charge!
**************************************************************************
       OptiFit=Goodfit
       OptiRecup=GoodRecup
       OptIntchromo(:)=Parent(:)
       OptiBinchromo(:)=chromBinR(:)
     DEALLOCATE(Fitness)
     DEALLOCATE(chromBinT)
     DEALLOCATE(Recup)
       WRITE(*,40)OptIntchromo
  40 FORMAT('   Meilleur reseau :',20(I2))
     PRINT*,''
     PRINT*,'   Recherche de la distribution de Puissance en cours...'
       J=1
       Intchromo(:,J)=OptIntchromo(:)
```

```
      Nconex=0

        DO k=1,NExch
          IF(Intchromo(k,J).NE.0)THEN
          Nconex=Nconex+1
          END IF
        END DO

      CALL MATRICE(J,Intchromo,HEN,NExch,Level,Nkey,key,strH,strC,BH,BC)
      cnt=0

    composition:DO T=1,HEN
      CALL
DISTRIBUTION(cnt,Nunit,TotGen,EchGen,NExch,decichromo,Binchromo)
        Digichromo(:,T)=Binchromo(:)
      cnt=cnt+1
    END DO composition

      Digichromo(:,1)=OptiBinchromo(:)
        BestFit=100E9
        GoodQrec=0.
        eXi(:)=0.
        BdTo(:)=1E7
        BdTi(:)=1E7

Research:   DO I=1,Iteration
        cumulFit=0.
        somQrec=0.
        cumulpenalty=0.
        moypenalty=0.
 Evaluation:DO T=1,HEN
 CALL CONVERSION(T,HEN,TotGen,EchGen,NExch,Digichromo,decichromo)
 CALL PUISSANCE2(Cp,BH,BC,I,Iteration,T,NExch,HEN,Nunit,equ,Key,&
                  Nkey,EchGen,Qmax,RHS,decichromo,strH,strC,Qi,Targ,Xi)
 CALL TEMPERATURE2(I,Iteration,T,strH,strC,HEN,NExch,BH,BC,Nkey,key,&
              Nunit,equ,Cp,Te,Ts,Xi,epso,epsin)
 dTo(:,T)=epso(:)
 dTi(:,T)=epsin(:)
 CALL PENALITE2(Nconex,T,HEN,Nkey,I,Xi,Nunit,Iteration,deltaT,&
                       epso,epsin,NExch,Penalty)
 Yi(:,T)=Xi(:,T)
 QREC(T)=Qi(T)
 OBJi(T)=Targ+Ck2*Penalty
 cumulfit=cumulfit+OBJi(T)
 somQrec=somQrec+QREC(T)
 cumulpenalty=cumulpenalty+penalty
 END DO Evaluation

        N4=MINLOC(OBJi(:))
        IF(OBJi(N4(1)).LT.BestFit)THEN
        BestFit=OBJi(N4(1))
```

```
          GoodQrec=QREC(N4(1))
          eXi(:)=Yi(:,N4(1))
          BdTo(:)=dTo(:,N4(1))
          BdTi(:)=dTi(:,N4(1))
        END IF
        moyQrec=somQrec/HEN
        moypenalty=cumulpenalty/HEN

        IF(I.EQ.Iteration)EXIT Research
 CALL RESTRICTION(I,Iteration,GoodQrec,Qmax,drap)
 CALL SELECTION2(drap,I,Iteration,HEN,TotGen,OBJi,&
                 Digichromo,Mother,Fitmod)
        Digichromo(:,1:HEN)=Mother(:,1:HEN)
        OBJi(1:HEN)=Fitmod(1:HEN)
 CALL
ELITISME2(Iteration,I,BestFit,OBJi,TotGen,HEN,Digichromo,EliteBin)
 CALL CROISEMENT2(cas,EliteBin,Digichromo,I,Iteration,TotGen,HEN,Nkey,&
                  NExch,EchGen,cumulFit,OBJi,Baby)
        Digichromo=Baby
END DO Research

************************************************************************
!Resultats de l'optimisation!
************************************************************************
        OptiX(:)=eXi(:)
        OptiRecup=GoodQrec
        OptiFit=BestFit
        QHmin=SUM(OptiX((NExch+Nkey+1):Nunit))
        OpdTo(:)=BdTo(:)
        OpdTi(:)=BdTi(:)
        PRINT*,'   RESULTATS:'
        PRINT*,''
      write(*,300)OptiRecup
      write(*,310)QHmin
      write(*,320)OptIntchromo(:)
        write(2,300)OptiRecup
      write(2,310)QHmin
      write(2,320)OptIntchromo(:)
 300 format('Puissance Recuperée :',F15.3)
 310 format('Utilité chaude      :',F15.3)
 320 format('Meilleur réseau     :'20(I2))
      print*,''
        WRITE(2,*)'Distribution des charges'
        DO k=1,Nunit
        IF(k.EQ.NExch+1)THEN
        WRITE(2,*)'Utilités froides et chaudes'
        END IF
      write(*,400)k,OptiX(k)
        write(2,400)k,OptiX(k)
 400 format(2x,'X(',I2,')=',F12.3)
      END DO
```

```
       write(*,420)(OptiRecup/Qmax)*100
       write(2,420)(OptiRecup/Qmax)*100
 420 format(2x,'Taux de recupération.=',F7.3,1x,'%')
       write(2,*)''
       write(*,*)''
       do k=1,NExch
       write(*,500)OpdTo(k),OpdTi(k)
       write(2,500)OpdTo(k),OpdTi(k)
 500 format(F9.3,2x,F9.3)
     end do

END   PROGRAM PRINCIPAL



              ************************************
              *     Sous-programme VARIABLES      *
              ************************************


SUBROUTINE VARIABLES(Nkey,Key,Level,NExch,equ,Nunit,TotGen,&
                     EchGen,HEN,Iteration)
IMPLICIT NONE
INTEGER               ::Nkey,Key,Level,NExch,equ,EchGen,TotGen
INTEGER               ::Nunit,HEN,Iteration

PRINT*,' ***********************************************'
PRINT*,' * SYNTHESE DE RESEAU D'ECHANGEURS OPTIMAL POUR *'
PRINT*,' * LA RECUPERATIONDE CHALEUR                    *'
PRINT*,' ***********************************************'
PRINT*,''
PRINT*,'   FIXER LE NOMBRE D''ITERATIONS :'
READ*,Iteration
10   PRINT*,'   FIXER LA TAILLE DE LA POPULATION: '
READ*,HEN

IF(HEN.GT.2)THEN
    IF((MOD(HEN,2)).GT.0.0)THEN
      CALL BEEPQQ(1500,200)
      PRINT*,'!!!Entrer un nombre Pair!!!'
      PRINT*,''
      GOTO 10
    ELSE
    END IF
ELSE
    CALL BEEPQQ(1500,200)
    PRINT*,'Entrer un nombre superieur a 2'
    PRINT*,''
    GOTO 10
```

```
END IF

PRINT*,''
PRINT*,' ****************************************'
PRINT*,'   CARACTERISTIQUES DE L''INSTALLATION   '
PRINT*,' ****************************************'
PRINT*,''
PRINT*,'   NOMBRE DE COURANTS CHAUDS:'
READ*,Nkey
PRINT*,'   NOMBRE DE COURANTS FROIDS:'
READ*,key
PRINT*,'   NOMBRE MAXIMAL D''ECHANGEURS PAR COURANT FROID :'
READ*,Level

NExch=Key*Level
equ=Nkey+Key
EchGen=12
Nunit=NExch+equ
TotGen=NExch*EchGen

END SUBROUTINE VARIABLES



              ************************************
              *     Sous-programme PARAMETRES     *
              ************************************



SUBROUTINE PARAMETRES(NkAy,ega,deltaT,Cp,Te,Ts,RHS,cas)
IMPLICIT NONE
INTEGER                              ::J,Nkay,ega,cas
REAL                                 ::deltaT
REAL,DIMENSION(1:ega)                ::Cp,Te,Ts
REAL,DIMENSION(1:ega)                ::RHS
CHARACTER(LEN=32)                    ::choix

     PRINT*,'      TEMPERATURE DE PINCEMENT dTmin:'
     READ*,deltaT
     PRINT*,'      CAPACITES CALORIFIQUES    :'
     DO J=1,ega
       IF(J.LE.Nkay)THEN
            PRINT 15,J
  15      FORMAT('       - courant chaud(',I2,')?')
       ELSE
            PRINT 25,(J-Nkay)
  25      FORMAT('       - courant froid(',I2,')?')
       END IF
       READ*,Cp(J)
```

```
      END DO
        PRINT*,'      TEMPERATURES D''ENTREE    : '
      DO J=1,ega
        IF(J.LE.Nkay)THEN
             PRINT 26,J
   26      FORMAT('        - courant chaud(',I2,')?')
        ELSE
           PRINT 36,(J-Nkay)
   36      FORMAT('        - courant froid(',I2,')?')
        END IF
        READ*,Te(J)
      END DO
        PRINT*,'      TEMPERATURES DE SORTIE    : '
      DO J=1,ega
        IF(J.LE.Nkay)THEN
             PRINT 37,J
   37      FORMAT('        - courant chaud(',I2,')?')
        ELSE
           PRINT 47,(J-Nkay)
   47      FORMAT('        - courant froid(',I2,')?')
        END IF
        READ*,Ts(J)
      END DO
      PRINT*,'       ELITISME (O/N)?:'
        READ '(A)',choix
        Performance:&
        SELECT CASE (choix)
        CASE ('o','O') Performance
        cas=2
      CASE ('n','N') Performance
        cas=0
        END SELECT Performance
      DO J=1,ega
        IF(J.LE.Nkay)THEN
        RHS(J)=Cp(J)*(Te(J)-Ts(J))
        ELSE
        RHS(J)=Cp(J)*(Ts(J)-Te(J))
        END IF
      END DO

END SUBROUTINE PARAMETRES
```

****

```
*      Sous-programme COMPOSITES      *
***************************************


SUBROUTINE COMPOSITES(daltaT,ega,Nkay,Cap,Ten,Tso,&
                      Tpinh,Tpinc,QHneed,QCneed,Qmax)
IMPLICIT NONE
INTEGER                          ::ct,J,k,ega,Nkay,cont,incr
INTEGER,DIMENSION(1)             ::N,Zero
REAL                             ::petit,smallneed,QHneed,QCneed,hotQ
REAL                             ::daltaT,Tpinh,Tpinc,Qmax
REAL,DIMENSION(1:2*ega)          ::allTemp,NtervdT,NtervdH,NtervCp,hotcurv
REAL,DIMENSION(0:2*ega)          ::casPow
REAL,DIMENSION(1:2*ega)          ::comp
REAL,DIMENSION(0:2*ega)          ::Ordre
REAL,DIMENSION(1:ega)            ::Ten,Tso,cap,Tin,Tou
REAL,DIMENSION(1:ega,1:2*ega)    ::Cpop

      ****************************************
        Températures modifiées
      ****************************************

      DO J=1,ega
           IF(J.LE.Nkay)THEN
              Tin(J)=Ten(J)-daltaT/2
              Tou(J)=Tso(J)-daltaT/2
           ELSE
            Tin(J)=Ten(J)+daltaT/2
              Tou(J)=Tso(J)+daltaT/2
           END IF
        END DO

        Do k=1,(2*ega)
           IF(k.LE.ega)THEN
               allTemp(k)=Tin(k)
           ELSE
               AllTemp(k)=Tou(k-ega)
           END IF
      END DO
      ****************************************************
      Classement par ordre decroissant des
        intervalles de temperature
      ****************************************************

        Ordre(0)=MAXVAL(AllTemp)
        petit=MINVAL(allTemp)
        WHERE(AllTemp.EQ.Ordre(0))
        AllTemp=0.
        ELSEWHERE
        END WHERE
```

```
    ct=1

 50 DO k=1,(2*ega)
      comp(k)=Ordre(ct-1)-allTemp(k)
    END DO
      N=MINLOC(comp,MASK=comp.GT.0.)
      Ordre(ct)=AllTemp(N(1))
    IF(Ordre(ct).EQ.petit) GOTO 100
    WHERE(AllTemp.EQ.Ordre(ct))
      allTemp=0.
      ELSEWHERE
      END WHERE
      ct=ct+1
      GOTO 50
100 cont=ct
    ****************************************
      Intervalles des Cp,dT,dH
    ****************************************

      DO k=1,cont
      NtervdT(k)=Ordre(k-1)-Ordre(k)
      END DO

    DO k=1,ega
         DO J=1,cont
              IF(k.LE.Nkay)THEN
                   IF(Ordre(J).GE.Tin(k).OR.Ordre(J).LT.Tou(k))THEN
                        Cpop(k,J)=0.
                   ELSE
                        Cpop(k,J)=Cap(k)
                   END IF
              ELSE
         IF(Ordre(J).GE.Tou(k).OR.Ordre(J).LT.Tin(k))THEN
              Cpop(k,J)=0.
           ELSE
              Cpop(k,J)=Cap(k)
           END IF
              END IF
         END DO
      END DO

      DO J=1,cont
         NtervCp(J)=-1*(SUM(Cpop(1:Nkay,J)))+SUM(Cpop(Nkay+1:ega,J))
       NtervdH(J)=NtervCp(J)*NtervdT(J)
         hotcurv(J)=SUM(Cpop(1:Nkay,J))*NtervdT(J)
    END DO
       hotQ=SUM(hotcurv(1:cont))
    ************************************************************
    cascade des surplus d'énergie intervalle par intervalle
    ************************************************************
    casPow(0)=0.
```

```
    incr=0
200 DO J=1,cont
    casPow(J)=casPow(J-1)-NtervdH(J)
      END DO

 Need: DO J=1,cont
      IF(casPow(J).LT.(-1E-3))EXIT Need
      IF(J.EQ.cont)GOTO 300
 END DO Need
    smallneed=MINVAL(casPow)
      casPow(0)=ABS(smallneed)
      incr=incr+1
      IF(incr.LE.100)GOTO 200
 CALL BEEPQQ(1500,1000)
      PRINT*,'     ERROR!!SVP ENTREZ LES DONNEES SUIVANTES:'
      PRINT*,''
      PRINT*,'     Besoins en source chaude QHneed?'
      READ*,QHneed
      PRINT*,'     Besoins en source froide QCneed?'
      READ*,QCneed
      PRINT*,'     Puissance maximale recuperable Qmax?'
      READ*,Qmax
      PRINT*,'     Temperature au point de pincement chaud Tpinh?'
      READ*,Tpinh
      PRINT*,'     Temperature au point de pincement froid Tpinc?'
      READ*,Tpinc
      GOTO 400
300 QHneed=casPow(0)
    QCneed=casPow(cont)
    Qmax=hotQ-QCneed
    ************************************************
      Recherche temperature de pinch des courants
    ************************************************
      Zero=MINLOC(casPow(1:cont-1))
      IF(casPow(Zero(1)).LE.1E-2)THEN
            Tpinh=Ordre(Zero(1))+daltaT/2
            Tpinc=Ordre(Zero(1))-daltaT/2
            GOTO 400
      ELSE
            CALL BEEPQQ(1500,1000)
            PRINT*,'    TEMPERATURE D''APPROCHE INADMISSIBLE!!!'
            PRINT*,''
          PRINT*,'     SVP ENTREZ LES DONNEES SUIVANTES:'
            PRINT*,''
            PRINT*,'     Besoins en source chaude QHneed?'
            READ*,QHneed
            PRINT*,'     Besoins en source froide QCneed?'
            READ*,QCneed
            PRINT*,'     Puissance maximale recuperable Qmax?'
            READ*,Qmax
            PRINT*,'Temperature au point de pincement chaud Tpinh?'
```

```
            READ*,Tpinh
            PRINT*,'Temperature au point de pincement froid Tpinc?'
            READ*,Tpinc
        END IF

400     END SUBROUTINE COMPOSITES
```

```
            ************************************
            *     Sous-programme POPULATION     *
            ************************************
```

```
SUBROUTINE POPULATION(HAN,NAxch,Nkay,Intchromo)
IMPLICIT NONE
INTEGER ,DIMENSION(1)                :: Seed
INTEGER                              :: k,T,Nkay,NAxch
INTEGER                              :: HAN,ct,count
REAL ,DIMENSION(1:NAxch)             :: R
INTEGER,DIMENSION(1:NAxch,1:HAN)     :: Intchromo

DO T=1,HAN
     ct=0
 10  CALL SYSTEM_CLOCK(Count)
     Seed = Count+T+ct
     CALL RANDOM_SEED( PUT = Seed )
     CALL RANDOM_NUMBER(R)

     DO k=1,NAxch
       Intchromo(k,T)=AINT((Nkay+0.5)*R(k))
     END DO

       ct=0
       DO k=1,NAxch
         IF(Intchromo(k,T).EQ.0)THEN
            ct=ct+1
         END IF
       END DO

       IF(ct.EQ.NAxch)GOTO 10
END DO
END SUBROUTINE POPULATION
```

```
            ************************************
```

```
              *      Sous-programme MATRIX          *
              ************************************


SUBROUTINE MATRIX(Ti,chrama,HAN,NAxch,Laval,Nkay,kay,strH,strC,BH,BC)
IMPLICIT NONE
INTEGER    ::Nkay,kay,HAN,NAxch,Laval,N,J,k,Ti,cnt,L
INTEGER,DIMENSION(1:NAxch,1:HAN)   ::chrama
INTEGER,DIMENSION(1:NAxch)         ::strH,strC
REAL,DIMENSION(1:NAxch,1:Nkay)     ::BH
REAL,DIMENSION(1:NAxch,1:Kay)      ::BC

     *********************************
     Courants chauds
     *********************************
StreamH:DO J=1,Nkay
       k=1
          DO WHILE(k.LE.NAxch)
               IF(chrama(k,Ti).EQ.J) THEN
                   BH(k,J)=1.
               ELSE
                   BH(k,J)=0.
               END IF
                   k=k+1
         END DO
ENDDO StreamH
     *********************************
       Courants froids
       *********************************
StreamC: DO J=1,Kay
     k=J
       N=1
          DO WHILE(N.LE.NAxch)
               IF(N.EQ.k) THEN
                   IF(chrama(k,Ti).NE.0)THEN
                       BC(k,J)=1.
                   ELSE
                       BC(k,J)=0.
                   END IF
                       k=k+Kay
             ELSE
                BC(N,J)=0.
               END IF
       N=N+1
          END DO
 ENDDO StreamC
       ***************************************************
       Recherche  courant chaud  et froid  de l'echangeur k
       ***************************************************
       cnt=0
```

```
Ligne2: DO L=1,Laval
           DO k=1,Kay
             cnt=cnt+1
                IF(chrama(cnt,Ti).NE.0)THEN
                       strH(cnt)=chrama(cnt,Ti)
                       strC(cnt)=k
                  ELSE
                       strH(cnt)=0
                   strC(cnt)=0
                END IF
          END DO
END DO Ligne2

END  SUBROUTINE MATRIX



              ************************************
              *     Sous-programme DISTRIBUTION   *
              ************************************


SUBROUTINE DISTRIBUTION(ct,Nunyt,TotalGene,ExchGene,NAxch,&
                        decichromo,Binchromo)
IMPLICIT NONE
INTEGER                              ::Count,slack,chromo_deci,Nunyt,ct
INTEGER                              ::TotalGene,ExchGene,NAxch
INTEGER,DIMENSION(1)                 ::Seed
INTEGER,DIMENSION(1:TotalGene)       ::Binchromo
INTEGER,DIMENSION(1:NAxch)           ::decichromo
INTEGER,DIMENSION(1:ExchGene,1:NAxch) ::Echgchromo
INTEGER,DIMENSION(1:TotalGene)       ::B
INTEGER                              ::I,k,L,T
REAL, DIMENSION(1:TotalGene)         ::R

     slack=1
     CALL SYSTEM_CLOCK(Count)
     Seed = Count+ct
     CALL RANDOM_SEED( PUT = Seed )
     CALL RANDOM_NUMBER(R)
     DO  I=1,TotalGene

       B(I)=ANINT(R(I))
         Binchromo(I)=B(I)
     ENDDO

TIT2: DO T=1,NAxch
       k=0
  TIT3: DO L=slack,TotalGene
```

```
          k=k+1
          Echgchromo(k,T)=Binchromo(L)
           IF(k.EQ.ExchGene)THEN
                slack=L+1
                EXIT TIT3
             END IF
  END DO TIT3
          chromo_deci=0
          DO I=ExchGene,1,-1
          chromo_deci=chromo_deci+Echgchromo(I,T)*2**(ExchGene-I)
          END DO
          decichromo(T)=chromo_deci
END DO TIT2

END SUBROUTINE DISTRIBUTION
```

```
              ************************************
              *     Sous-programme PUISSANCE      *
              ************************************
```

```
SUBROUTINE PUISSANCE(BaH,BaC,Ie,ateration,Ti,NAxch,HAN,Nunyt,ega,&

Kay,Nkay,ExchGene,Qmox,LHS,decichreme,strmH,strmC,Q,Targ,X)
IMPLICIT NONE
INTEGER                        ::Kay,Nkay,NAxch,HAN,ega,ExchGene,Nunyt,Ti
INTEGER                        ::L,J,k,ateration,Ie,scrute
REAL                           ::som,Qmox,Targ,Minload,conec,r
INTEGER,DIMENSION(1:NAxch)     ::decichreme
INTEGER,DIMENSION(1:NAxch)     ::strmH,strmC
REAL,DIMENSION(1:2)            ::Load
INTEGER,DIMENSION(1)           ::N
REAL,DIMENSION(1:NAxch)        ::binf,bsup,coef
REAL,DIMENSION(1:Nunyt,1:ateration)::X
REAL,DIMENSION(1:ega)                 ::LHS
REAL,DIMENSION(1:NAxch,1:Nkay)        ::BaH
REAL,DIMENSION(1:NAxch,1:Kay)         ::BaC
REAL,DIMENSION(1:ateration)           ::Q
```

```
     ****************************************
```

```
      Recherche des bornes des variables
        ************************************
        r=0.2
10 Bornes1:DO k=1,NAxch
             IF(strmH(k).EQ.0)THEN
                X(k,Ie)=0.
                CYCLE Bornes1
             END IF
      Minload=MIN(LHS(strmH(k)),LHS(strmC(k)+Nkay))
      Load(1)=LHS(strmH(k))
      Load(2)=LHS(strmC(k)+Nkay)
      N=MINLOC(Load)
        Scrute=N(1)
        conec=0.
        IF(scrute.EQ.1)THEN
             DO L=1,NAxch
                  IF(BaH(L,strmH(k)).NE.0.)THEN
                  conec=conec+1.
                  END IF
           END DO
      ELSE
             DO L=1,NAxch
                IF(BaC(L,strmC(k)).NE.0.)THEN
                  conec=conec+1.
                  END IF
             END DO
      END IF
        binf(k)=0.
      bsup(k)=Minload*(1/conec+r)
        coef(k)=(bsup(k)-binf(k))/(2**ExchGene-1)
        X(k,Ie)=binf(k)+coef(k)*decichreme(k)
END DO Bornes1

      ****************************
      Charge total Q
        ****************************
      Q(Ie)=SUM(X(1:NAxch,Ie))
        IF(Q(Ie).GT.Qmox)THEN
            r=r-0.025
            GOTO 10
        END IF
      ****************************
      Utilités chaudes et froides
        ****************************
DO J=1,ega
        som=0.
              IF(J.LE.Nkay)THEN
                    DO k=1,NAxch
                         som=som+BaH(k,J)*X(k,Ie)
                         X((NAxch+J),Ie)=LHS(J)-som
                    END DO
```

```
              ELSE
                    DO k=1,NAxch
                        som=som+BaC(k,(J-Nkay))*X(k,Ie)
                        X((NAxch+J),Ie)=LHS(J)-som
                    END DO
              END IF
        IF(X((NAxch+J),Ie).LT.0.)THEN
              r=r-0.05
             GOTO 10
        END IF
END DO
        Targ=0.
        DO k=NAxch+1,Nunyt
        Targ=Targ+X(k,Ie)
        END DO
END  SUBROUTINE PUISSANCE



              ************************************
              *      Sous-programme TEMPERATURE    *
              ************************************


SUBROUTINE TEMPERATURE(Ie,Iteration,h,strmH,strmC,HAN,NAxch,BaH,BaC,&
                  Nkay,kay,Nunyt,ega,Cap,Tie,Tos,eX,epso,epsin)
IMPLICIT NONE
INTEGER                              ::Nkay,kay,NAxch,HAN,Nunyt,ega
INTEGER                              ::J,k,L,cnt,h,Ie,Iteration
INTEGER,DIMENSION(1:NAxch)           ::strmH,strmC
REAL,DIMENSION(1:NAxch,1:Nkay)       ::BaH
REAL,DIMENSION(1:NAxch,1:kay)        ::BaC
REAL ,DIMENSION(1:ega)               ::Cap,Tie,Tos
REAL ,DIMENSION(1:NAxch)             ::Tsoh,Tenh,Tsoc,Tenc,epso,epsin
REAL ,DIMENSION(1:Nunyt,1:Iteration) ::eX
REAL,DIMENSION(1:NAxch,1:NAxch)      ::dTh,dTc


        ************************************************************
      Calcul des temperatures entrée et sortie  de l'echangeur k
        ************************************************************
      Inequation1: DO k=1,NAxch
      IF(strmH(k).EQ.0)THEN
        epso(k)=1E3
        epsin(k)=1E3
        CYCLE Inequation1
        END IF

        DO L=1,k
```

```
        dTh(L,k)=BaH(L,strmH(k))*eX(L,Ie)/Cap(strmH(k))
        dTc(L,k)=BaC(L,strmC(k))*eX(L,Ie)/Cap(strmC(k)+Nkay)
        END DO

      Tsoh(k)=Tos(strmH(k))+SUM(dTh(1:(K-1),K))+&
        eX((NAxch+strmH(k)),Ie)/Cap(strmH(k))

      Tenh(k)= Tsoh(k)+dTh(k,k)
      Tenc(k)=Tie(strmC(k)+Nkay)+SUM(dTc(1:(K-1),k))
      Tsoc(k)=Tenc(k)+dTc(k,k)
      epso(k)=Tsoh(k)-Tenc(k)
        epsin(k)=Tenh(k)-Tsoc(k)
      END DO Inequation1

END SUBROUTINE TEMPERATURE



              ************************************
              *      Sous-programme PÉNALITÉ       *
              ************************************


SUBROUTINE PENALITE(Nconax,Ti,HAN,Ie,eX,Nunyt,ateration,&
                            daltaT,epsso,epsen,NAxch,Penalty)
IMPLICIT NONE
INTEGER                          ::NAxch,HAN,Nunyt,ateration
INTEGER                          ::k,Ie,Ti,Nconax
REAL                             ::daltaT,penalty
REAL                             ::Psotot,Pentot
REAL,DIMENSION(1:NAxch)          ::epsso,epsen,pso,pen
REAL ,DIMENSION(1:Nunyt,1:ateration) ::eX

      Psotot=0.
        Pentot=0.
        DO k=1,NAxch
               IF(epsso(k).GT.daltaT)THEN
                  pso(k)=0.
               ELSE
                    IF(epsso(k).LT.0.)THEN
                        pso(k)=Nconax*(1-epsso(k)/daltaT)
                    ELSE
                         IF(epsso(k).GE.0.0.AND.epsso(k).LE.daltaT)THEN
                             pso(k)=(1-epsso(k)/daltaT)
                         END IF
                    END IF
               END IF
        Psotot=Psotot+pso(k)
        END DO
```

```
        DO k=1,NAxch
              IF(epsen(k).GT.daltaT)THEN
                 pen(k)=0.
              ELSE
                   IF(epsen(k).LT.0.)THEN
                       pen(k)=Nconax*(1-epsen(k)/daltaT)
                   ELSE
                        IF(epsen(k).GE.0.0.AND.epsen(k).LE.daltaT)THEN
                             pen(k)=(1-epsen(k)/daltaT)
                        END IF
                   END IF
             END IF
        Pentot=Pentot+pen(k)
        END DO
        penalty=(Pentot+Psotot)

END SUBROUTINE PENALITE
```

```
*************************************
*     Sous-programme RESTRICTION    *
*************************************
```

```
SUBROUTINE RESTRICTION(rat,Nbrot,GaadRecup,Qmox,drap)
IMPLICIT NONE
INTEGER                              ::drap,rat,Nbrot
REAL                                 ::Qmox
REAL                                ::GaadRecup


IF( GaadRecup.LE.(0.90*Qmox))drap=1
IF( GaadRecup.GT.(0.90*Qmox).AND.GaadRecup.LT.(0.95*Qmox))drap=2
IF( GaadRecup.GE.(0.95*Qmox))drap=3

END SUBROUTINE RESTRICTION
```

```
*************************************
```

```
*     Sous-programme SÉLECTION      *
************************************


SUBROUTINE SELECTION(drop,rat,Nbrot,HAN,NAxch,Fitess,&
                     chrama,Father,Fitessmod)
IMPLICIT NONE
INTEGER                ::T,I,k,rat,Nbrot,HAN,NAxch,Totindexe,drop
REAL                   ::somfit,moyfit,adjust
INTEGER,DIMENSION(1)   ::Elit,N
INTEGER,DIMENSION(1:HAN) ::Indexe
INTEGER,DIMENSION(1:NAxch,1:HAN)     ::chrama,chromsort
INTEGER,DIMENSION(1:NAxch,1:(4*HAN))::Father
REAL,DIMENSION(1:HAN)                ::fitcount
INTEGER,DIMENSION(0:HAN)             ::som
REAL,DIMENSION(1:HAN)                ::Fitess,Fitsort
REAL,DIMENSION(1:4*HAN)              ::Fitessmod


     IF(drop.EQ.1) adjust=0.15
       IF(drop.EQ.2) adjust=0.20
       IF(drop.EQ.3) adjust=0.40

Classement:DO T=1,HAN
       N=MINLOC(Fitess(:))
       chromsort(:,T)=chrama(:,N(1))
       Fitsort(T)=Fitess(N(1))
       Fitess(N(1))=1E7
END DO classement

     DO T=1,HAN
       chrama(:,T)=chromsort(:,T)
       Fitess(T)=Fitsort(T)
     END DO

     ********************************************
       Determination de la representativité(Indexe)
       de chaque chromosome en fonction de son fitness
       ********************************************
       somfit=0
       DO k=1,HAN
       somfit=somfit+Fitess(k)
       END DO

       moyfit=somfit/HAN

DO k=1,HAN
       fitcount(k)=(moyfit/Fitess(k))-adjust
            IF((1-fitcount(k)).GT.0.AND.(1-fitcount(k)).LE.0.1)THEN
                fitcount(k)=1.
```

```
            END IF
      Indexe(k)=ANINT(fitcount(k))
        IF(Indexe(k).GT.4)THEN
        Indexe(k)=4
        END IF
END DO

      ************************************************
      Selection des chromosomes en fonction de leur
        indexe
        ************************************************
      som(0)=0
        DO k=1,HAN
        som(k)=som(k-1)+Indexe(k)
        END DO

Placement:DO I=1,HAN
      DO k=(som(I-1)+1),som(I)
            IF(Indexe(I).EQ.0) CYCLE Placement
            Father(:,k)=chrama(:,I)
            Fitessmod(k)=Fitess(I)
        END DO
END DO Placement

        Elit=MINLOC(Fitess(:))
        Totindexe=0
        DO k=1,HAN
        Totindexe=Totindexe+Indexe(k)
        END DO
           IF(Totindexe.LT.HAN) THEN
               DO k=(Totindexe+1),HAN
                     Father(:,k)=chrama(:,Elit(1))
                     Fitessmod(k)=Fitess(Elit(1))
                 END DO
             END IF
END SUBROUTINE SELECTION
```

****

```
*     Sous-programme ÉLITISME       *
************************************


SUBROUTINE ELITISME(Nbrot,rat,Ti,Gaadfit,Fitess,&
                    NAxch,HAN,chrama,Elite)
IMPLICIT NONE
INTEGER
::k,rat,Ti,NAxch,HAN,Nbrot
INTEGER                          ::El1
INTEGER,DIMENSION(1)                 ::El2
INTEGER,DIMENSION(1:NAxch,1:2)       ::Elite
INTEGER,DIMENSION(1:NAxch,1:HAN)     ::chrama
REAL ,DIMENSION(1:HAN)               ::comp
REAL                                 ::Gaadfit
REAL,DIMENSION(1:HAN)                ::Fitess

Elitism1:DO Ti=1,HAN
     IF(Fitess(Ti).EQ.Gaadfit)THEN
            El1=Ti
            EXIT Elitism1
       END IF
            IF(Ti.EQ.HAN) El1=HAN
END DO Elitisme1
       Elite(:,1)=chrama(:,El1)

Elitism2:DO Ti=1,HAN
       IF(Ti.EQ.El1)CYCLE Elitism2
            comp(Ti)=Fitess(Ti)-Fitess(El1)
END DO Elitisme2
       comp(El1)=1E6
       El2=MINLOC(comp)
       Elite(:,2)=chrama(:,El2(1))

END SUBROUTINE ELITISME


************************************
```

```
*      Sous-programme CROISEMENT      *
***************************************


SUBROUTINE CROISEMENT(Elute,chrama,rat,Nbrota,NAxch,HAN,Nkay,&
                          somFitt,Fitess,Child)

IMPLICIT NONE
INTEGER
::H,m,rat,J,k,T,Ti,range1,range2,HAN,NAxch
INTEGER
::Pos1,Pos2,ct,Pos,Nbrota,Nkay,count
INTEGER,DIMENSION(1)      ::Seed
INTEGER,DIMENSION(1:2)    ::N
INTEGER,DIMENSION(1:HAN)              ::Discrim
INTEGER,DIMENSION(1:NAxch,1:HAN)::chrama,child,childm
INTEGER,DIMENSION(1:NAxch,1:2)   ::Elute
REAL                              ::somal,som,somFitt,Site,Pcro,Pmu
REAL,DIMENSION(1:HAN)             ::R
REAL,DIMENSION(1:2*HAN)           ::alter
REAL,DIMENSION(1:HAN)             ::w1
REAL,DIMENSION(1:HAN)             ::Fitess
REAL,DIMENSION(1:NAxch)           ::fantom

        Pcro=0.90
        Pmu=0.4
CALL SYSTEM_CLOCK(Count)
     Seed = Count
CALL RANDOM_SEED( PUT = Seed )
CALL RANDOM_NUMBER(R)

CALL SYSTEM_CLOCK(Count)
     Seed = Count+2
CALL RANDOM_SEED( PUT = Seed )
CALL RANDOM_NUMBER(alter)

CALL SYSTEM_CLOCK(Count)
     Seed = Count+4
CALL RANDOM_SEED( PUT = Seed )
CALL RANDOM_NUMBER(w1)

     range2=2
       ct=0
     k=1
     N(:)=0
search1: DO Ti=1,(HAN-2)/2
     J=1
 100 somal=R(k)*somFitt
     som=0.
   search2:DO T=1,HAN
```

```
          som=som+Fitess(T)
          IF(som.GT.somal)EXIT search2
    END DO search2
        k=k+1
      N(J)=T-1
      IF(N(J).LT.1) N(J)=1
            IF(J.LT.2)THEN
          J=J+1
            GOTO 100
           END IF
      ************************************
      Choix du site de croisement
        ************************************
      IF(R(Ti).GT.Pcro) THEN
             Pos=0
             GOTO 200
        END IF

        Site=NAxch*R(Ti)
        Pos=AINT(site)

      ************************************
      Croisement des chromosomes
        ************************************
 200  range1=range2+1
      DO m=NAxch,1,-1
              IF(m.GT.Pos) THEN
                 childm(m,range1)=chrama(m,N(1))
                 child(m,range1)=chrama(m,N(1))
              ELSE
                 childm(m,range1)=chrama(m,N(2))
                 child(m,range1)=chrama(m,N(2))
              END IF
        END DO

        range2=range1+1
      DO m=NAxch,1,-1
              IF(m.GT.Pos) THEN
                childm(m,range2)=chrama(m,N(2))
                child(m,range2)=chrama(m,N(2))
              ELSE
                childm(m,range2)=chrama(m,N(1))
                child(m,range2)=chrama(m,N(1))
              END IF
        END DO
        ********************************
        Mutation des chromosomes
        ********************************
CALL SYSTEM_CLOCK(Count)
      Seed = Count+Ti
CALL RANDOM_SEED( PUT = Seed )
```

```
CALL RANDOM_NUMBER(fantom)

       ct=ct+1
     IF(w1(range1).LE.Pmu)THEN
             Pos1=AINT(NAxch*alter(ct))+1
             ct=ct+1
             Pos2=AINT(NAxch*alter(ct))+1
        IF((MOD(rat,2)).GT.0.0)THEN
             child(Pos1,range1)=AINT((Nkay+0.5)*fantom(Pos1))
          ELSE
             child(Pos1,range1)=childm(Pos2,range1)
             child(Pos2,range1)=childm(Pos1,range1)
          END IF
     END IF

       ct=ct+1
     IF(w1(range2).LE.Pmu)THEN
           Pos1=AINT(NAxch*alter(ct))+1
           ct=ct+1
           Pos2=AINT(NAxch*alter(ct))+1
           IF((MOD(rat,2)).GT.0.0)THEN
                child(Pos1,range2)=AINT((Nkay+0.5)*fantom(Pos1))
             ELSE
                child(Pos1,range2)=childm(Pos2,range2)
                child(Pos2,range2)=childm(Pos1,range2)
             END IF
       END IF
END DO search1

     DO T=2+1,HAN
       Discrim(T)=SUM(child(:,T))
            IF(Discrim(T).EQ.0)THEN
               DO k=1,NAxch,2
                 child(k,T)=AINT((Nkay+0.5)*alter(k))
               ENDDO
            ELSE
            END IF
     END DO

     child(:,1)=Elute(:,1)
     child(:,2)=Elute(:,2)
END  SUBROUTINE CROISEMENT
```

****

```
*      Sous-programme PUISSANCE2      *
***************************************


SUBROUTINE PUISSANCE2(Cap,BaH,BaC,Ie,ateration,Ti,NAxch,HAN,Nunyt,&
      ega,Kay,Nkay,ExchGene,Qmox,LHS,decichreme,strmH,strmC,Qi,Targ,Xi)
IMPLICIT NONE
INTEGER                    ::Kay,Nkay,NAxch,HAN,ega,ExchGene,Nunyt,Ti
INTEGER                    ::J,k,L,ateration,Ie,scrute
REAL                       ::som,Qmox,Targ,acroi
REAL                       ::r,Minload,conec
INTEGER,DIMENSION(1:NAxch) ::decichreme
INTEGER,DIMENSION(1:NAxch)     ::strmH,strmC
INTEGER,DIMENSION(1)           ::N
REAL,DIMENSION(1:2)            ::Load
REAL,DIMENSION(1:NAxch)        ::binf,bsup,coef
REAL,DIMENSION(1:Nunyt,1:HAN)  ::Xi
REAL,DIMENSION(1:ega)          ::LHS,Cap
REAL,DIMENSION(1:NAxch,1:Nkay) ::BaH
REAL,DIMENSION(1:NAxch,1:Kay)  ::BaC
REAL,DIMENSION(1:HAN)          ::Qi

       r=0.20
10 Bornes1:DO k=1,NAxch
         IF(strmH(k).EQ.0)THEN
            Xi(k,Ti)=0.
            CYCLE Bornes1
         END IF
     Minload=MIN(LHS(strmH(k)),LHS(strmC(k)+Nkay))
     Load(1)=LHS(strmH(k))
     Load(2)=LHS(strmC(k)+Nkay)
     N=MINLOC(Load)
       Scrute=N(1)
       conec=0.
       IF(scrute.EQ.1)THEN
             DO L=1,NAxch
                  IF(BaH(L,strmH(k)).NE.0.)THEN
                      conec=conec+1.
                  END IF
            END DO
     ELSE
             DO L=1,NAxch
                  IF(BaC(L,strmC(k)).NE.0.)THEN
                     conec=conec+1.
                  END IF
             END DO
     END IF
       binf(k)=0.
     bsup(k)=Minload*(1/conec+r)
       coef(k)=(bsup(k)-binf(k))/(2**ExchGene-1)
```

```
        Xi(k,Ti)=binf(k)+coef(k)*decichreme(k)
END DO Bornes1

      *********************************
      Charge total Q
        *********************************
      Qi(Ti)=SUM(Xi(1:NAxch,Ti))
        IF(Qi(Ti).GT.Qmox)THEN
        r=r-0.020
        GOTO 10
        END IF

        *********************************
      Utilités chaudes et froides
        *********************************
        DO J=1,ega
        som=0.
              IF(J.LE.Nkay)THEN
                  DO k=1,NAxch
                     som=som+BaH(k,J)*Xi(k,Ti)
                     Xi((NAxch+J),Ti)=LHS(J)-som
                  END DO
              ELSE
                  DO k=1,NAxch
                     som=som+BaC(k,(J-Nkay))*Xi(k,Ti)
                     Xi((NAxch+J),Ti)=LHS(J)-som
                  END DO
              END IF
           IF(Xi((NAxch+J),Ti).LT.0.)THEN
              r=r-0.05
              GOTO 10
           END IF
        END DO

        DO J=1,ega
        acroi=0.
        IF(Xi((NAxch+J),Ti)/Cap(J).LT.0.01)THEN
                  acroi=Xi((NAxch+J),Ti)
                  Xi((NAxch+J),Ti)=0.
Reglage:DO k=NAxch,1,-1
          IF(J.LE.Nkay)THEN
               IF(BaH(k,J).GT.0.)THEN
                   Xi(k,Ti)=Xi(k,Ti)+acroi
               END IF
          ELSE
               IF(BaC(k,(J-Nkay)).NE.0.)THEN
                   Xi(k,Ti)=Xi(k,Ti)+acroi
               END IF
          END IF
        EXIT Reglage
END DO Reglage
```

```
      END IF
      END DO
            Targ=0.
            DO k=NAxch+1,Nunyt
            Targ=Targ+Xi(k,Ti)
          END DO
END   SUBROUTINE PUISSANCE2
```

```
                ************************************
                *     Sous-programme CONVERSION      *
                ************************************
```

```
SUBROUTINE CONVERSION(Ti,HAN,TotalGene,ExchGene,NAxch,&
                    Digitchromo,decichromo)
IMPLICIT NONE
INTEGER                      ::k,slack,I,L,J,Ti,HAN,ExchGene,NAxch
INTEGER                                  ::chromo_deci,TotalGene
INTEGER,DIMENSION(1:NAxch)               ::decichromo
INTEGER,DIMENSION(1:TotalGene,1:HAN)     ::Digitchromo
INTEGER,DIMENSION(1:ExchGene,1:NAxch)    ::Echgchromo

       slack=1
TIT2:  DO J=1,NAxch
       k=0
       TIT3:  DO L=slack,TotalGene
              k=k+1
              Echgchromo(k,J)=Digitchromo(L,Ti)
                  IF(k.EQ.ExchGene)THEN
                      slack=L+1
                      EXIT TIT3
                  END IF
       END DO TIT3
       chromo_deci=0
            DO I=ExchGene,1,-1
            chromo_deci=chromo_deci+Echgchromo(I,J)*2**(ExchGene-I)
            END DO
       decichromo(J)=chromo_deci
END DO TIT2

END SUBROUTINE CONVERSION
```

```
                ************************************
```

```
*      Sous-programme TEMPERATURE2    *
***************************************

SUBROUTINE
TEMPERATURE2(Ie,ateration,h,strmH,strmC,HAN,NAxch,BaH,BaC,Nkay,&
                  kay,Nunyt,ega,Cap,Tie,Tos,eX,epso,epsin)
IMPLICIT NONE
INTEGER                                 ::Nkay,kay,NAxch,HAN,Nunyt,ega
INTEGER                                 ::J,k,L,cnt,h,Ie,ateration
INTEGER,DIMENSION(1:NAxch)              ::strmH,strmC
REAL,DIMENSION(1:NAxch,1:Nkay)          ::BaH
REAL,DIMENSION(1:NAxch,1:kay)           ::BaC
REAL ,DIMENSION(1:ega)                  ::Cap,Tie,Tos
REAL ,DIMENSION(1:NAxch)                ::Tsoh,Tenh,Tsoc,Tenc,epso,epsin
REAL ,DIMENSION(1:Nunyt,1:HAN)          ::eX
REAL,DIMENSION(1:NAxch,1:NAxch)         ::dTh,dTc


  ************************************************************
  Calcul des temperatures entrée et sortie  de l'echangeur k
  ************************************************************
Inequation1: DO k=1,NAxch
  IF(strmH(k).EQ.0)THEN
     epso(k)=1E3
     epsin(k)=1E3
     CYCLE Inequation1
    END IF

    DO L=1,k
    dTh(L,k)=BaH(L,strmH(k))*eX(L,h)/Cap(strmH(k))
    dTc(L,k)=BaC(L,strmC(k))*eX(L,h)/Cap(strmC(k)+Nkay)
    END DO

  Tsoh(k)=Tos(strmH(k))+SUM(dTh(1:(K-1),K))+&
    eX((NAxch+strmH(k)),h)/Cap(strmH(k))

  Tenh(k)= Tsoh(k)+dTh(k,k)
  Tenc(k)=Tie(strmC(k)+Nkay)+SUM(dTc(1:(K-1),k))
  Tsoc(k)=Tenc(k)+dTc(k,k)
  epso(k)=Tsoh(k)-Tenc(k)
    epsin(k)=Tenh(k)-Tsoc(k)
  END DO Inequation1

END SUBROUTINE TEMPERATURE2


***************************************
```

```
*      Sous-programme PÉNALITÉ2      *
************************************

SUBROUTINE PENALITE2(Nconax,Ti,HAN,Nkay,Ie,eX,Nunyt,ateration,&
                              daltaT,epsso,epsen,NAxch,Penalty)
IMPLICIT NONE
INTEGER                          ::NAxch,HAN,Nunyt,ateration,Nkay
INTEGER                          ::k,Ie,Ti,Nconax
REAL                             ::daltaT,penalty
REAL                             ::Psotot,Pentot
REAL,DIMENSION(1:NAxch)          ::epsso,epsen,pso,pen
REAL ,DIMENSION(1:Nunyt,1:HAN)::eX

       Psotot=0.
       Pentot=0.
       DO k=1,NAxch
            IF(epsso(k).GT.daltaT)THEN
                pso(k)=0.
            ELSE
                 IF(epsso(k).LT.0.)THEN
                     pso(k)=Nconax*(1-epsso(k)/daltaT)
                 ELSE
                    IF(epsso(k).GE.0.0.AND.epsso(k).LE.daltaT)THEN
                           pso(k)=1-epsso(k)/daltaT
                      END IF
                 END IF
            END IF
       Psotot=Psotot+pso(k)
       END DO

       DO k=1,NAxch
            IF(epsen(k).GT.daltaT)THEN
                pen(k)=0.
            ELSE
                 IF(epsen(k).LT.0.)THEN
                      pen(k)=Nconax*(1-epsen(k)/daltaT)
                 ELSE
                    IF(epsen(k).GE.0.0.AND.epsen(k).LE.daltaT)THEN
                         pen(k)=1-epsen(k)/daltaT
                      END IF
                END IF
            END IF
       Pentot=Pentot+pen(k)
       END DO
       Penalty=(Pentot+Psotot)

END SUBROUTINE PENALITE2

************************************
```

```
*      Sous-programme SÉLECTION2      *
***************************************


SUBROUTINE SELECTION2(drop,Ia,ateration,HAN,TotalGene,Fitess,&
                      Digitchromo,Mother,Fitmod)
IMPLICIT NONE
INTEGER         ::T,I,k,Ia,ateration,HAN,TotalGene,Totindexe,drop
REAL            ::somfit,moyfit,adjust
INTEGER,DIMENSION(1)              ::Elit,N
INTEGER,DIMENSION(1:HAN)          ::Indexe
INTEGER,DIMENSION(1:TotalGene,1:HAN)     ::Digitchromo,chromsort
INTEGER,DIMENSION(1:TotalGene,1:(4*HAN)) ::Mother
REAL,DIMENSION(1:HAN)                    ::fitcount
INTEGER,DIMENSION(0:HAN)                 ::som
REAL,DIMENSION(1:HAN)                    ::Fitess,Fitsort
REAL,DIMENSION(1:4*HAN)                  ::Fitmod



       IF(drop.EQ.1) adjust=0.00
       IF(drop.EQ.2) adjust=0.00
       IF(drop.EQ.3) adjust=0.00

Classement:DO T=1,HAN
       N=MINLOC(Fitess(:))
       chromsort(:,T)=Digitchromo(:,N(1))
       Fitsort(T)=Fitess(N(1))
       Fitess(N(1))=1E7
END DO classement
     DO T=1,HAN
       Digitchromo(:,T)=chromsort(:,T)
       Fitess(T)=Fitsort(T)
     END DO

     ***********************************************
       Determination de la representativité(Indexe)
       de chaque chromosome en fonction de son fitness
       ***********************************************
       somfit=0
       DO T=1,HAN
       somfit=somfit+Fitess(T)
       END DO

       moyfit=somfit/HAN

DO T=1,HAN
       fitcount(T)=(moyfit/Fitess(T))-adjust
       IF((1-fitcount(T)).GT.0.AND.(1-fitcount(T)).LE.0.1)THEN
       fitcount(T)=1.
```

```
        END IF
      Indexe(T)=ANINT(fitcount(T))
        IF(Indexe(T).GT.4)THEN
        Indexe(T)=4
        END IF
END DO

      ******************************************
      Selection des chromosomes en fonction de leur
        indexe
        ******************************************
      som(0)=0
        DO T=1,HAN
          som(T)=som(T-1)+Indexe(T)
        END DO

Placement:DO T=1,HAN
      DO k=(som(T-1)+1),som(T)
         IF(Indexe(T).EQ.0) CYCLE Placement
         Mother(:,k)=Digitchromo(:,T)
         Fitmod(k)=Fitess(T)
        END DO
END DO Placement

        Elit=MINLOC(Fitess(:))

        Totindexe=0
        DO T=1,HAN
          Totindexe=Totindexe+Indexe(T)
        END DO
         IF(Totindexe.LT.HAN) THEN
              DO T=(Totindexe+1),HAN
                    Mother(:,T)=Digitchromo(:,Elit(1))
                    Fitmod(T)=Fitess(Elit(1))
                END DO
           END IF

END SUBROUTINE SELECTION2
```

****

```
*      Sous-programme ÉLITISME2       *
************************************

SUBROUTINE ELITISME2(ateration,Ia,Bastfit,ABJi,TotalGene,&
                     HAN,Digitchromo,EliteBin)
IMPLICIT NONE
INTEGER                          ::k,Ia,Ti,TotalGene,HAN,ateration
INTEGER                  ::El1
INTEGER,DIMENSION(1)                      ::El2
INTEGER,DIMENSION(1:TotalGene,1:2)    ::EliteBin
INTEGER,DIMENSION(1:TotalGene,1:HAN)  ::Digitchromo
REAL ,DIMENSION(1:HAN)                    ::comp
REAL                                          ::BastFit
REAL,DIMENSION(1:HAN)                     ::ABJi

Elitism1:DO Ti=1,HAN
         IF(ABJi(Ti).EQ.BastFit)THEN
             El1=Ti
             EXIT Elitism1
         END IF
END DO Elitism1
       EliteBin(:,1)=Digitchromo(:,El1)
Elitism2:DO Ti=1,HAN
       IF(Ti.EQ.El1)CYCLE Elitism2
       comp(Ti)=ABJi(Ti)-ABJi(El1)
END DO Elitism2
       comp(El1)=1E6
       El2=MINLOC(comp)
       EliteBin(:,2)=Digitchromo(:,El2(1))
END SUBROUTINE ELITISME2




************************************
*      Sous-programme CROISEMENT2     *
************************************


SUBROUTINE CROISEMENT2(cass,Elute,chrama,Ia,ateration,TotalGene,&
                           HAN,NkayNAxch,EchGene,CumulFitt,Fitess,Baby)

IMPLICIT NONE
INTEGER
::L,H,m,Ia,J,k,T,Ti,range1,range2,Npos,NAxch
INTEGER
::Pos1,Pos2,ct,Pos,ateration,Nkay,cass,EchGene
```

```
INTEGER
::HAN,TotalGene,count,ret,Site1,Site2,proPos
INTEGER,DIMENSION(1:TotalGEne)       ::Pom
INTEGER,DIMENSION(1:2)                 ::N
INTEGER,DIMENSION(1)                   ::Seed
INTEGER,DIMENSION(1:HAN)                 ::Discrim
INTEGER,DIMENSION(1:TotalGene,1:HAN)::chrama,Baby,Babym
INTEGER,DIMENSION(1:TotalGene,1:2)   ::Elute
REAL                                 ::somal,som,CumulFitt,Pcro,Pmu
REAL,DIMENSION(1:HAN)                ::R
REAL,DIMENSION(1:TotalGene)            ::alter1,alter2
REAL,DIMENSION(1:HAN)                ::alter
REAL,DIMENSION(1:HAN)                ::w1
REAL,DIMENSION(1:HAN+2)              ::cros1,cros2
REAL,DIMENSION(1:HAN)                ::Fitess

      Pcro=0.10
        Pmu=0.03
        IF(Ia.LE.0.70*ateration)THEN
      proPos=NAxch*EchGene
        ELSE
      proPos=0.1*NAxch*EchGene
      END IF

CALL SYSTEM_CLOCK(Count)
      Seed = Count+Ia
CALL RANDOM_SEED( PUT = Seed )
CALL RANDOM_NUMBER(alter)

CALL SYSTEM_CLOCK(Count)
      Seed = Count+Ia+1
CALL RANDOM_SEED( PUT = Seed )
CALL RANDOM_NUMBER(cros1)

CALL SYSTEM_CLOCK(Count)
      Seed = Count+Ia+2
CALL RANDOM_SEED( PUT = Seed )
CALL RANDOM_NUMBER(cros2)

CALL RANDOM_SEED(  )
CALL RANDOM_NUMBER( R )

      range2=cass
        ct=0
      k=1
        N(:)=0
search1: DO Ti=1,(HAN-cass)/2
      J=1
 100  somal=R(k)*CumulFitt
      som=0.
   search2:DO T=1,HAN
```

```
      som=som+Fitess(T)
        IF(som.GT.somal)THEN
              ret=T
            EXIT search2
          END IF
  END DO search2
      k=k+1
    N(J)=ret-1
      IF(N(J).LT.1) N(J)=1
         IF(J.LT.2)THEN
             J=J+1
             GOTO 100
         END IF

    ************************************
      Croisement des chromosomes
    ************************************
    IF(R(Ti).GT.Pcro) THEN
         Pos=0
         GOTO 200
      END IF
      ************************************
      Choix du site de croisement
      ************************************
      Site1=AINT(TotalGene*cros1(Ti)+1)
      Site2=AINT(TotalGene*cros2(Ti+1)+1)
      Pos1=MIN(Site1,Site2)
      Pos2=MAX(site1,site2)


200 range1=range2+1
    DO m=1,TotalGene
           IF(m.LE.Pos1.OR.m.GE.Pos2) THEN
                Babym(m,range1)=chrama(m,N(1))
                Baby(m,range1)=chrama(m,N(1))
           ELSE
                Babym(m,range1)=chrama(m,N(2))
                Baby(m,range1)=chrama(m,N(2))
           END IF
      END DO

      range2=range1+1
    DO m=1,TotalGene
           IF(m.LE.Pos1.OR.m.GE.Pos2) THEN
                Babym(m,range2)=chrama(m,N(2))
                Baby(m,range2)=chrama(m,N(2))
           ELSE
                Babym(m,range2)=chrama(m,N(1))
              Baby(m,range2)=chrama(m,N(1))
           END IF
      END DO
```

```
      ************************************************
      Mutation des chromosomes  pom=position de mutation
      ************************************************
IF(alter(Ti).LE.Pmu)THEN
  CALL SYSTEM_CLOCK(Count)
     Seed = Count+Ti
  CALL RANDOM_SEED( PUT = Seed )
  CALL RANDOM_NUMBER(w1)

     Npos=AINT(proPos*w1(Ti)+1.)

  CALL SYSTEM_CLOCK(Count)
     seed=count+range1
  CALL RANDOM_SEED(PUT=seed  )
  CALL RANDOM_NUMBER( alter1 )
     DO L=1,Npos
          Pom(L)=AINT(TotalGene*alter1(L)+1.)
        Baby(Pom(L),range1)=1-Babym(Pom(L),range1)
       END DO


  CALL SYSTEM_CLOCK(Count)
       seed=count+range2
  CALL RANDOM_SEED(PUT=seed  )
  CALL RANDOM_NUMBER( alter2 )
     DO L=1,Npos
          Pom(L)=AINT(TotalGene*alter2(L)+1)
        Baby(Pom(L),range2)=1-Babym(Pom(L),range2)
       END DO
END IF
END DO search1

     DO T=cass+1,HAN
         Discrim(T)=SUM(Baby(:,T))
           IF(Discrim(T).EQ.0)THEN
                DO k=1,TotalGene,2
                   Baby(k,T)=AINT((1+0.5)*alter(k))
                ENDDO
           ELSE
           END IF
     END DO
     IF(cass.EQ.0)GOTO 400
     Baby(:,1)=Elute(:,1)
     Baby(:,2)=Elute(:,2)

 400 END  SUBROUTINE CROISEMENT2
```

ANNEXE 2 : Code de l'algorithme d'optimisation (approche G/S)

```
************************************
*      Programme PRINCIPAL         *
************************************


Program PRINCIPAL
IMPLICIT NONE
REAL ,DIMENSION(:),ALLOCATABLE     ::RHS,Cp,X,Te,Ts,Tea,Tsa,Teb,Tsb
REAL ,DIMENSION(:),ALLOCATABLE     ::Goodfit,comp,Qmoy
REAL ,DIMENSION(:,:,:),ALLOCATABLE::Tih,Toh,Tic,Toc
REAL ,DIMENSION(:,:),ALLOCATABLE   ::Akj,AkjH,Bkj,Q
REAL,DIMENSION(:,:,:),ALLOCATABLE  ::duties
REAL                               ::Z,deltaT,somqrec,Pc,Pmut,Tpinh
REAL                               ::QHneed,QCneed,Qmax,QREC,Tpinc
INTEGER,DIMENSION(:),ALLOCATABLE   ::Discrim,Option,Refcolopt
INTEGER                            ::Nkey,key,Level,HEN,NExch,Ncol
INTEGER                            ::k,I,J,Ki,T,Iter,ineq,equ,rot
INTEGER                            ::NuC,NuH,colopt,Exces,Ncolopt
INTEGER                            ::Util,mtot,ct,Ex,Quit,mut,contVoX
INTEGER                            ::cntV,cntH,Resopt,incr,Ess
INTEGER,DIMENSION(:,:),ALLOCATABLE::chromo,child,chreme,OptiHEN
INTEGER,DIMENSION(:,:),ALLOCATABLE::BH,BC,calapt
INTEGER,DIMENSION(:,:,:),ALLOCATABLE::chromRef,Aption
INTEGER,DIMENSION(:,:),ALLOCATABLE  ::Elite
CHARACTER(LEN=32)                   ::choix

!!!"""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""""
!!!Parametres du Reseau Temperature,capacité calorifique
CALL VARIABLES(Nkey,Key,Level,NExch,equ,ineq,mtot,Util,Ncol,HEN,Iter)

ALLOCATE(chromo(1:NExch,1:HEN),OptiHEN(1:NExch,1:HEN))
ALLOCATE(chreme(1:NExch,1:2*HEN),chromRef(1:NExch,1:HEN,1:Iter))
ALLOCATE(child(1:NExch,1:HEN),Elite(1:NExch,1:2))
ALLOCATE(Goodfit(1:Iter),Qmoy(1:Iter),Q(1:HEN,1:Iter))
ALLOCATE(Refcolopt(1:Util),Option(1:Util),Discrim(1:HEN))
ALLOCATE(Akj(1:Ncol,0:mtot),AkjH(1:Ncol,0:mtot))
ALLOCATE(Tsa(1:equ),Tsb(1:equ),Teb(1:equ),Tea(1:equ)RHS(0:equ))
ALLOCATE(Bkj(1:Ncol,1:equ),Ts(1:equ),Te(1:equ),Cp(1:equ))
ALLOCATE(duties(1:Util,1:HEN,1:Iter),X(1:Util))
ALLOCATE(Aption(1:Util,1:HEN,1:Iter),calapt(1:HEN,1:Iter))
ALLOCATE(BH(1:NExch,1:Nkey),BC(1:NExch,1:Key),Comp(1:HEN))
ALLOCATE(Tih(1:NExch,1:HEN,1:Iter),Toh(1:NExch,1:HEN,1:Iter))
ALLOCATE(Tic(1:NExch,1:HEN,1:Iter),Toc(1:NExch,1:HEN,1:Iter))
OPEN(6,FILE='HENPINCH.TXT')
```

```
CALL PARAMETRES(Nkey,equ,Pc,Pmut,deltaT,Cp,Te,Ts,RHS)
PRINT*,'      UTILISER L'' ELITISME (O/N)?:'
READ '(A)',choix

Performance:&
SELECT CASE (choix)
CASE ('o','O') Performance
Ex=2
CASE ('n','N') Performance
Ex=0
END SELECT Performance

CALL COMPOSITES(deltaT,equ,Nkey,Cp,Te,Ts,Tpinh,Tpinc,&
                    QHneed,QCneed,Qmax)

CALL PINCEMENT(Tpinh,Tpinc,Te,Ts,Cp,deltaT,Nkey,equ,&
                                Tea,Tsa,Teb,Tsb,NuH,NuC)

Exces=0
Nbressai:DO rot=1,2
    IF(rot.EQ.2)THEN
        PRINT*,'2eme ESSAI APPUYER SUR ''ENTER''...'
        PAUSE
      END IF
20  Ess=1
      Quit=0
      chreme=0
      incr=0
      PRINT*,'  CONSTITUTION DES RESEAUX '
      PRINT*,''

48  CALL POPULATION(Te,deltaT,HEN,NExch,Nkey,Key,Level,&
                  Discrim,chromo)
      incr=incr+1
         IF(incr.GE.20)THEN
              CALL BEEPQQ(1500,1000)
              PRINT*,'    ATTENTION!!:Revoir vos donnees'
              GOTO 100
         END IF

 Iteration:DO I=1,Iter
58    somqrec=0.

  Network:DO T=1,HEN
    CALL MATRICE(T,chromo,HEN,RHS,deltaT,Cp,Ncol,mtot,equ,ineq,&
      NExch,Nkey,key,Te,Ts,BH,BC,Bkj,AkjH,cntV,cntH,contVoX)
    CALL PHASE2(NExch,AkjH,mtot,Ncol,cntH,cntV,contVoX,Util,&
                    X,Z,colopt,Option)
      calapt(T,I)=colopt
      Aption(:,T,I)=Option(:)
    IF(Z.EQ.0.)THEN
```

```
      Q(T,I)=0.
      CYCLE Network
  ENDIF
    CALL TEMPERATURE(T,HEN,I,Iter,deltaT,Bkj,BH,BC,Util,equ,Nkey,&
         Key,NExch,Ncol,Cp,Te,Ts,X,Z,Toh,Tih,Toc,Tic,Q,duties)
  somqrec=somqrec+Q(T,I)
    chromRef(:,T,I)=chromo(:,T)
ENDDO Network

 IF(I.EQ.1)THEN
           IF(Quit.EQ.1)GOTO 68
            ct=0
     Verif:DO T=1,HEN
          IF(Q(T,I).GT.1E-3)THEN
               ct=ct+1
                 IF(Ess.EQ.1)THEN
                 ki=ct
                 chreme(:,ki)=chromo(:,T)
                 ELSE
                 ki=ki+1
                 chreme(:,ki)=chromo(:,T)
            ENDIF
            ENDIF
     ENDDO Verif
        IF(ki.LT.HEN)THEN
                 Ess=2
                 GOTO 48
            ELSE
                 DO T=1,HEN
                 chromo(:,T)=chreme(:,T)
                 END DO
                 quit=1
                 Q(:,I)=0.
            GOTO 58
            END IF
ENDIF
68         IF(ABS(somqrec).LE.1E-2)THEN
              CALL PopInit(Te,deltaT,HEN,NExch,Nkey,Key,&
                                         Level,Discrim,chromo)
                  GOTO 58
                 END IF

            Goodfit(I)=MAXVAL(Q(:,I),MASK=Q(:,I).GT.0.)
                  Qmoy(I)=somqrec/HEN
            IF(ABS(Qmax-Goodfit(I)).LE.1E-3)EXIT Iteration
                IF(I.EQ.1)THEN
                PRINT*,' EN COURS D''OPTIMISATION......'
                END IF

    CALL ELITISME(I,Iter,NExch,HEN,chromo,Goodfit,Q,Elite)
    CALL CROISEMENT(Ex,Elite,chromo,I,Iter,NExch,HEN,Nkey,&
```

```
                          somqrec,Q,Pc,Pmut,Child)
            chromo=child
  ENDDO Iteration

      CALL SOLUTION(calapt,Aption,NuH,NuC,Tpinh,Tpinc,QCneed,&
                QHneed,Qmax,Level,Util,Iter,rot,NExch,HEN, &
                  Nkey,Key,duties,Goodfit,Q,chromRef,Tih,Toh,&
                  Tic,Toc,Exces,Ncolopt,Refcolopt,Resopt,OptiHEN,QREC)

        IF(Exces.EQ.1)GOTO 20
        chromo(:,1)=OptiHEN(:,Resopt)
        IF(ABS(Qmax-QREC).LE.1E-3) EXIT Nbressai
END DO Nbressai
      PRINT*,''
      PRINT*,'   VOIR DIVISIONS DES COURANTS DANS ''HENPINCH'' '
      PRINT*,''
      PRINT*,'   POUR QUITTER APPUYER SUR ENTER... '
        PAUSE

100 END Program PRINCIPAL



                ************************************
                *     Sous-programme VARIABLES     *
                ************************************



SUBROUTINE VARIABLES(Nkey,Key,Level,NExch,equ,ineq,mtot,&
                     Util,Ncol,HEN,Iter)
IMPLICIT NONE
INTEGER             ::Nkey,Key,Level,NExch,equ,ineq
INTEGER             ::mtot,Util,Ncol,HEN,Iter

PRINT*,' ************************************************'
PRINT*,' *    SYNTHESE DE RESEAU D''ECHANGEURS DE CHALEUR *'
PRINT*,' *    POUR LA RECUPERATION DE CHALEUR             *'
PRINT*,' ************************************************'
PRINT*,''
PRINT*,''
PRINT*,''
PRINT*,'  FIXER LE NOMBRE D''ITERATIONS  :'
READ*,Iter
10    PRINT*,' FIXER LA TAILLE DE LA POPULATION: '
      READ*,HEN
            IF(HEN.GT.2)THEN
                IF((MOD(HEN,2)).GT.0.0)THEN
                     CALL BEEPQQ(1500,200)
                     PRINT*,'!!!Entrer un nombre Pair!!!'
```

```
                    PRINT*,''
                    GOTO 10
               END IF
               ELSE
                   CALL BEEPQQ(1500,200)
                   PRINT*,'Entrer un nombre superieur a 2'
                   PRINT*,''
                   GOTO 10
              END IF
PRINT*,''
PRINT*,'       CARACTERISTIQUES DE L''INSTALLATION   '
PRINT*,''
PRINT*,'       NOMBRE DE COURANTS CHAUDS:'
READ*,Nkey
PRINT*,'       NOMBRE DE COURANTS FROIDS:'
READ*,key
PRINT*,' NOMBRE DE NIVEAUX:'
READ*,Level
NExch=Key*Level
equ=Nkey+Key
ineq=2*NExch
mtot=equ+ineq
Util=NExch+equ
Ncol=Util+ineq+1

END SUBROUTINE VARIABLES



                  ************************************
                  *     Sous-programme PARAMÈTRES     *
                  ************************************


SUBROUTINE PARAMETRES(NkAy,ega,Pc,Pmut,deltaT,Cp,Te,Ts,RHS)
IMPLICIT NONE
INTEGER                              ::J,Nkay,ega
REAL                                 ::Pc,Pmut,deltaT
REAL,DIMENSION(1:ega)                ::Cp,Te,Ts
REAL,DIMENSION(0:ega)                ::RHS

PRINT*,'  PROBABILITE DE CROISEMENT (%):'
READ*,Pc
Pc=0.01*Pc
PRINT*,'  PROBABILITE DE MUTATION (%)  :'
READ*,Pmut
Pmut=0.01*Pmut
PRINT*,'  TEMPERATURE DE PINCEMENT dTmin:'
READ*,deltaT
PRINT*,'  CAPACITES CALORIFIQUES    :'
```

```
DO J=1,ega
      IF(J.LE.Nkay)THEN
          PRINT 15,J
  15    FORMAT(' courant chaud(',I2,')?')
      ELSE
          PRINT 25,(J-Nkay)
  25    FORMAT(' courant froid(',I2,')?')
      END IF
          READ*,Cp(J)
END DO
          PRINT*,' TEMPERATURES D''ENTREE    : '

DO J=1,ega
      IF(J.LE.Nkay)THEN
          PRINT 26,J
  26    FORMAT(' courant chaud(',I2,')?')
      ELSE
        PRINT 36,(J-Nkay)
  36    FORMAT(' courant froid(',I2,')?')
      END IF
          READ*,Te(J)
  END DO

  PRINT*,' TEMPERATURES DE SORTIE    : '

DO J=1,ega
      IF(J.LE.Nkay)THEN
          PRINT 37,J
  37    FORMAT(' courant chaud(',I2,')?')
      ELSE
        PRINT 47,(J-Nkay)
  47    FORMAT(' courant froid(',I2,')?')
      END IF
          READ*,Ts(J)
END DO
        RHS(0)=0.0
DO J=1,ega
        IF(J.LE.Nkay)THEN
           RHS(J)=Cp(J)*(Te(J)-Ts(J))
        ELSE
             RHS(J)=Cp(J)*(Ts(J)-Te(J))
        END IF
END DO

END SUBROUTINE PARAMETRES
```

```
                ************************************
                *       Sous-programme COMPOSITE     *
                ************************************


SUBROUTINE COMPOSITE(daltaT,ega,Nkay,Cap,Ten,Tso,&
                     Tpinh,Tpinc,QHneed,QCneed,Qmax)
IMPLICIT NONE
INTEGER                          ::ct,J,k,ega,Nkay,cont,incr
INTEGER,DIMENSION(1)             ::N,Zero
REAL                             ::petit,smallneed,QHneed,QCneed,hotQ
REAL                             ::daltaT,Tpinh,Tpinc,Qmax
REAL,DIMENSION(1:2*ega)          ::allTemp,NtervdT,NtervdH,NtervCp,hotcurv
REAL,DIMENSION(0:2*ega)          ::casPow
REAL,DIMENSION(1:2*ega)          ::comp
REAL,DIMENSION(0:2*ega)          ::Ordre
REAL,DIMENSION(1:ega)            ::Ten,Tso,cap,Tin,Tou
REAL,DIMENSION(1:ega,1:2*ega)    ::Cpop

        !!!*********************************
            !!!Temperatures modifiees
        DO J=1,ega
              IF(J.LE.Nkay)THEN
                Tin(J)=Ten(J)-daltaT/2
                Tou(J)=Tso(J)-daltaT/2
                ELSE
            Tin(J)=Ten(J)+daltaT/2
                Tou(J)=Tso(J)+daltaT/2
                END IF
          END DO
            Do k=1,(2*ega)
                IF(k.LE.ega)THEN
                  allTemp(k)=Tin(k)
                  ELSE
                  AllTemp(k)=Tou(k-ega)
                  END IF
        END DO
            !!!!!****************************************
            !!!classement par ordre decroissant des
            !!!intervalles de temperature
            Ordre(0)=MAXVAL(AllTemp)
            petit=MINVAL(allTemp)
            WHERE(AllTemp.EQ.Ordre(0))
            AllTemp=0.
            ELSEWHERE
            END WHERE
        ct=1
 50     DO k=1,(2*ega)
            comp(k)=Ordre(ct-1)-allTemp(k)
        END DO
```

```
            N=MINLOC(comp,MASK=comp.GT.0.)
            Ordre(ct)=AllTemp(N(1))
          IF(Ordre(ct).EQ.petit) GOTO 100
          WHERE(AllTemp.EQ.Ordre(ct))
            allTemp=0.
            ELSEWHERE
            END WHERE
            ct=ct+1
            GOTO 50
  100     cont=ct
          !!!!*********************************
            !!!intervalles des Cp,dT,dH
            DO k=1,cont
            NtervdT(k)=Ordre(k-1)-Ordre(k)
            END DO
       DO k=1,ega
             DO J=1,cont
               IF(k.LE.Nkay)THEN
                 IF(Ordre(J).GE.Tin(k).OR.Ordre(J).LT.Tou(k))THEN
                 Cpop(k,J)=0.
                 ELSE
                 Cpop(k,J)=Cap(k)
                 END IF
               ELSE
              IF(Ordre(J).GE.Tou(k).OR.Ordre(J).LT.Tin(k))THEN
                 Cpop(k,J)=0.
                 ELSE
                 Cpop(k,J)=Cap(k)
                 END IF
             END DO
          END DO
            DO J=1,cont
              NtervCp(J)=-
1*(SUM(Cpop(1:Nkay,J)))+SUM(Cpop(Nkay+1:ega,J))
          NtervdH(J)=NtervCp(J)*NtervdT(J)
            hotcurv(J)=SUM(Cpop(1:Nkay,J))*NtervdT(J)
          END DO
          hotQ=SUM(hotcurv(1:cont))
          !!!!*********************************************************
            !!!cascade des surplus d'énergie intervalle par inter-
valle
          casPow(0)=0.
            incr=0
  200   DO J=1,cont
          casPow(J)=casPow(J-1)-NtervdH(J)
            END DO
  Need:   DO J=1,cont
            IF(casPow(J).LT.(-1E-3))EXIT Need
            IF(J.EQ.cont)GOTO 300
            END DO Need
```

```
          smallneed=MINVAL(casPow)
              casPow(0)=ABS(smallneed)
              incr=incr+1
              IF(incr.LE.100)GOTO 200
              CALL BEEPQQ(1500,1000)
              PRINT*,'  ERROR!!SVP ENTREZ LES DONNEES SUIVANTES:'
              PRINT*,''
              PRINT*,'  Besoins en source chaude QHneed?'
              READ*,QHneed
              PRINT*,'  Besoins en source froide QCneed?'
              READ*,QCneed
              PRINT*,'  Puissance maximale recuperable Qmax?'
              READ*,Qmax
              PRINT*,'  Temperature au point de pincement chaud Tpinh?'
              READ*,Tpinh
              PRINT*,'  Temperature au point de pincement froid Tpinc?'
              READ*,Tpinc
              GOTO 400
300       QHneed=casPow(0)
          QCneed=casPow(cont)
          Qmax=hotQ-QCneed

          !!!*********************************************
              !!recherche temperature de pincement des courants
              Zero=MINLOC(casPow(1:cont-1))
              IF(casPow(Zero(1)).LE.1E-2)THEN
              Tpinh=Ordre(Zero(1))+daltaT/2
              Tpinc=Ordre(Zero(1))-daltaT/2
              GOTO 400
              ELSE
              CALL BEEPQQ(1500,1000)
              PRINT*,'    TEMPERATURE D''APPROCHE INADMISSIBLE!!!'
              PRINT*,''
        PRINT*,'     SVP ENTREZ LES DONNEES SUIVANTES:'
              PRINT*,''
              PRINT*,'     Besoins en source chaude QHneed?'
              READ*,QHneed
              PRINT*,'     Besoins en source froide QCneed?'
              READ*,QCneed
              PRINT*,'     Puissance maximale recuperable Qmax?'
              READ*,Qmax
              PRINT*,'        Temperature au point de pincement chaud
Tpinh?'
              READ*,Tpinh
              PRINT*,'        Temperature au point de pincement froid
Tpinc?'
              READ*,Tpinc
              END IF

400       END SUBROUTINE COMPOSITE
```

```
                    ************************************
                    *     Sous-programme PINCEMENT       *
                    ************************************


SUBROUTINE PINCEMENT(Tpynh,Tpync,Ten,Tso,Cap,daltaT,Nkay,ega,Tea,&
                     Tsa,Teb,Tsb,NuH,NuC)
IMPLICIT NONE
INTEGER                              ::J,Nkay,ega,NuH,NuC
REAL                                 ::daltaT,Tpynh,Tpync
REAL,DIMENSION(1:ega)                ::Tea,Tsa,Teb,Tsb
REAL,DIMENSION(1:ega)                ::Ten,Tso,Cap


      DO J=1,Nkay
               IF(Ten(J).GT.Tpynh) THEN
                  IF(Tso(J).LE.Tpynh) THEN
                       Tsa(J)=Tpynh
                         Tea(J)=Ten(J)
                            IF(Tso(J).LT.Tpynh)THEN
                              Teb(J)=Tpynh
                                Tsb(J)=Tso(J)
                        ELSE
                                Teb(J)=0.
                                  Tsb(J)=0.
                               END IF
              ELSE
                       Tsa(J)=Tso(J)
                         Tea(J)=Ten(J)
                         Teb(J)=0.
                         Tsb(J)=0.
                        END IF
               ELSE
                  Tsa(J)=0.
                    Tea(J)=0.
                    Teb(J)=Ten(J)
                    Tsb(J)=Tso(J)
          END IF
   END DO

     DO J=Nkay+1,ega
            IF(Ten(J).LT.Tpync) THEN
                 IF(Tso(J).GE.Tpync) THEN
                       Tsb(J)=Tpync
                         Teb(J)=Ten(J)
                            IF(Tso(J).GT.Tpync) THEN
                              Tea(J)=Tpync
                              Tsa(J)=Tso(J)
                            ELSE
                              Tea(J)=0.
```

```
                                Tsa(J)=0.
                        END IF
                    ELSE
                            Tsb(J)=Tso(J)
                              Teb(J)=Ten(J)
                              Tea(J)=0.
                              Tsa(J)=0.
                         END IF
                     ELSE
                       Teb(J)=0.
                         Tsb(J)=0.
                         Tea(J)=Ten(J)
                         Tsa(J)=Tso(J)
                     END IF
   END DO
        NuH=0
          NuC=0
        DO J=1,ega
        IF(Tea(J).GT.1E-3)NuH=NuH+1
          ENDDO
        DO J=1,ega
        IF(Teb(J).GT.1E-3)NuC=NuC+1
          ENDDO
          WRITE(6,*)'COURANTS AU-DESSUS/AU-DESSOUS DU PINCH'
          WRITE(6,*)'--------------------------------------'
          WRITE(6,*)''
          WRITE(6,20)Tpynh
          WRITE(6,30)Tpync
20      FORMAT('Temperature de Pinch courants chauds Tpinh=',F8.3)
30      FORMAT('Temperature de Pinch courants froids Tpinc=',F8.3)
          WRITE(6,*)''
          WRITE(6,*)' ************************************************'
        WRITE(6,*)' au-dessus du Pincement    au-dessous du Pincement'
          WRITE(6,*)' ---------------------    ----------------------'
          WRITE(6,*)'      Te          Ts           Te          Ts'
        WRITE(6,*)''
          DO J=1,ega
          IF(J.LE.Nkay)THEN
          WRITE(6,100)J,Tea(J),Tsa(J),J,Teb(J),Tsb(J)
100   FORMAT(2x,'H',I1,1x,F10.3,1x,F10.3,8x,'H',I1,1x,F10.3,1x,F10.3)
          ELSE
          WRITE(6,200)(J-Nkay),Tea(J),Tsa(J),(J-Nkay),Teb(J),Tsb(J)
200   FORMAT(4x,'C',I1,1x,F10.3,2x,F10.3,14x,'C',I1,1x,F10.3,2x,F10.3)
        ENDIF
        END DO
          WRITE(6,*)' ********************************************  '

END SUBROUTINE PINCEMENT
```

```
                ************************************
                *      Sous-programme POPULATION     *
                ************************************


SUBROUTINE POPULATION(Ten,daltaT,HAN,NAxch,Nkay,&
          Kay,Laval,Discrim,chromo)
IMPLICIT NONE
REAL                              :: daltaT
INTEGER                           :: H,J,k,L,T,Nkay,NAxch,kay,Laval,HAN,ct
REAL ,DIMENSION(1:Kay)            :: R
REAL ,DIMENSION(1:10)             :: Q
REAL ,DIMENSION(1:(Kay+Nkay))     :: Ten
REAL ,DIMENSION(1:NAxch)          :: W
INTEGER,DIMENSION(1:NAxch,1:HAN),INTENT(INOUT):: chromo
INTEGER,DIMENSION(1:HAN)             :: Discrim
INTEGER,DIMENSION(1:Kay,1:Laval)     :: B

          T=1
  DO WHILE(T.LE.HAN)
            CALL RANDOM_SEED()
            CALL RANDOM_NUMBER (Q)

  10      J=1
          DO L=1,Laval
            CALL RANDOM_SEED()
            DO H=1,3E5
              END DO
          CALL RANDOM_NUMBER (R)
            DO k=1,Kay
          w(J)=(Nkay+0.5)*R(k)
              B(k,L)=AINT(W(J))
              chromo(J,T)=B(k,L)
                 ct=0
                   IF(J.GT.2)THEN
      50             IF(chromo(J,T).EQ.chromo(J-2,T))THEN
                       ct=ct+1
                chromo(J,T)=AINT((Nkay+0.5)*Q(ct))
                       IF(ct.EQ.10)GOTO 100
                       GOTO 50
                       ELSE
                       ENDIF
                   ELSE
                   END IF
  100     IF(B(k,L).NE.0)THEN
                IF(Ten(Nkay+k).GT.(Ten(B(k,L))-daltaT))THEN
                B(k,L)=0
                  chromo(J,T)=B(k,L)
                ELSE
                END IF
```

```
            ELSE
            END IF
          J=J+1
            END DO
  END DO
            Discrim(T)=SUM(chromo(:,T))
            IF(Discrim(T).EQ.0)THEN
                T=T
                ELSE
                T=T+1
                END IF
            END DO
END SUBROUTINE POPULATION



                ************************************
                *      Sous-programme MATRICE        *
                ************************************



SUBROUTINE MATRICE(h,chrama,HAN,SHR,dT,Cap,Ncal,mtat,ega,inega,&
                NAxch,Nkay,kay,Ten,Tso,BH,BC,Bkj,AkjH,cntV,cntH,contVoX)
IMPLICIT NONE
INTEGER                 ::Nkay,kay,HAN,NAxch,N,Ncal,Util,mtat,bip,inega
INTEGER                 ::J,k,L,h,slack,str1,str2,ega,cntV,cntH,contVoX
REAL ,DIMENSION(0:ega)                ::SHR
REAL ,DIMENSION(1:ega)                ::Ten,Tso,Cap
REAL ,DIMENSION(1:Ncal,0:mtat)        ::Akj,AkjV,AkjH
REAL ,DIMENSION(1:Ncal,1:ega)         ::Bkj
REAL ,DIMENSION(1:Ncal,1:(inega/2))   ::Okj,Ikj
REAL                                  ::dT
INTEGER,DIMENSION(1:NAxch,1:HAN)      ::chrama,chromBit
INTEGER,DIMENSION(1:NAxch,1:Nkay)     ::BH
INTEGER,DIMENSION(1:NAxch,1:Kay)      ::BC
Util=NAxch+ega

          !************************************************************
          !!Identification des échangeurs existants sur chaque courant
        DO k=1,NAxch
                  IF(chrama(k,h).GT.0) THEN
                      chromBit(k,h)=1
                  ELSE
                      chromBit(k,h)=0
                  END IF
           END DO
          !***************
            !courant chaud
  StreamH:DO J=1,Nkay
```

```
            k=1
         DO WHILE(k.LE.NAxch)
            IF(chrama(k,h).EQ.J) THEN
            BH(k,J)=1
              ELSE
            BH(k,J)=0
              END IF
              k=k+1
           END DO
ENDDO StreamH
         !*****************
           !courant froid
StreamC: DO J=1,Kay
              k=J
            N=1
       DO WHILE(N.LE.NAxch)
          IF(N.EQ.k) THEN
              IF(chrama(k,h).NE.0)THEN
              BC(k,J)=1
              ELSE
              BC(k,J)=0
              END IF
              k=k+Kay
        ELSE
        BC(N,J)=0
            END IF
            N=N+1
       END DO
ENDDO StreamC

  !*********************************************************
  !!Determinations des coefficients des Contraintes égalité
Tabl:DO J=1,ega
              slack=NAxch+J
        IF(J.LE.Nkay)THEN
            IF(SUM(BH(:,J)).EQ.0)THEN
             Bkj(:,J)=0
                   Bkj(slack,J)=1
                   Bkj(Ncal,J)=SHR(J)
                   CYCLE Tabl
                   ELSE
                   END IF
    DO k=1,Ncal
              IF(k.LE.NAxch)THEN
         Bkj(k,J)=BH(k,J)
              ELSE
              Bkj(k,J)=0.0
              END IF
    END DO
            ELSE
                  IF(SUM(BC(:,(J-Nkay))).EQ.0)THEN
```

```
            Bkj(:,J)=0
                  Bkj(slack,J)=1
                  Bkj(Ncal,J)=SHR(J)
                  CYCLE Tabl
                  ELSE
                END IF
    DO k=1,Ncal
              IF(k.LE.NAxch)THEN
              Bkj(k,J)=BC(k,(J-Nkay))
              ELSE
              Bkj(k,J)=0.0
              END IF
      END DO
            END IF
              Bkj(slack,J)=1
            Bkj(Ncal,J)=SHR(J)
END DO Tabl
!*************************************************************
!!Determinations des coefficients des Contraintes inégalité

Inequation1: DO k=1,inega/2
         slack=Util+k

     !***********************************
     !!Recherche connexions courant chaud
       !!et froid de l'echangeur k
  Ligne2:DO J=1,Nkay
         IF(Bkj(k,J).GT.0.0)THEN
           str1=J
             EXIT Ligne2
             ELSE
             IF(J.EQ.Nkay)THEN
                 Okj(:,k)=0.0
                 Ikj(:,k)=0.0
                 GOTO  100
                 ELSE
                 END IF
             END IF
  END DO Ligne2

  Ligne3:DO J=(Nkay+1),ega
         IF(Bkj(k,J).GT.0.0)THEN
             str2=J
             EXIT Ligne3
             ELSE
             END IF
  END DO Ligne3
```

```
        !!*****************************************
        !!Coeffficients de l'equation de l'echangeur k

            !! a) Contraintes de temperature extremite froide

  DO L=1,Ncal
       IF(L.LE.(k-1))THEN
         Okj(L,k)=BH(L,str1)*1/Cap(str1)-BC(L,(str2-Nkay))*1/Cap(str2)
       ELSE
             Okj(L,k)=0.0
       END IF
  END DO
              Okj((NAxch+str1),k)=1/Cap(str1)
            Okj(Ncal,k)=dT-(Tso(str1)-Ten(str2))
              IF(Okj(Ncal,k).GE.0.0)THEN
            Okj(slack,k)=-1.
            ELSE
              Okj(:,k)=-1*Okj(:,k)
            Okj(slack,k)=1.
            END IF

              !! b) Contraintes de temperature extremite chaude

  DO L=1,Ncal
      IF(L.LE.k)THEN
            Ikj(L,k)=BH(L,str1)*1/Cap(str1)-BC(L,(str2-
Nkay))*1/Cap(str2)
      ELSE
            Ikj(L,k)=0.0
      END IF
  END DO
              Ikj((NAxch+str1),k)=1/Cap(str1)
            Ikj(Ncal,k)=dT-(Tso(str1)-Ten(str2))
              IF(Ikj(Ncal,k).GE.0.0)THEN
            Ikj((slack+NAxch),k)=-1
            ELSE
              Ikj(:,k)=-1*Ikj(:,k)
                Ikj((slack+NAxch),k)=1
            END IF
 100 END DO Inequation1

           !!!! Coefficients de la ligne zero (fonction à optimiser)
  DO k=1,Ncal
               IF(k.LE.NAxch)THEN
               Akj(k,0)=-1*chromBit(k,h)
               ELSE
               Akj(k,0)=0.0
               END IF
  END DO
               Akj(Ncal,0)=SHR(0)
```

```
          !!*************************************
          !!!Coefficients des lignes de contraintes

          !! a)contraintes egalité
DO   J=1,ega
          DO k=1,Ncal
              Akj(k,J)=Bkj(k,J)
              END DO
            Akj(Ncal,J)=SHR(J)
END DO
            !! b) contraintes inegalité (extremite froide)
DO   J=(ega+1),(ega+inega/2)
              DO k=1,Ncal
              Akj(k,J)=Okj(k,(J-ega))
          END DO
END DO
          !! c) contraintes inegalité (extremite chaude)
DO   J=(ega+inega/2+1),mtat
              DO k=1,Ncal
              Akj(k,J)=Ikj(k,(J-(ega+inega/2)))
          END DO
END DO
          !!****************************
              !!Reduction taille du tableau
          AkjH=0.
            AkjV=0.
DO J=0,mtat
              DO k=1,Util
              AkjV(k,J)=Akj(k,J)
          ENDDO
END DO
            cntV=Util
ReductionV:DO k=Util+1,Ncal
            bip=0
            DO J=0,mtat
              IF(Akj(K,J).NE.0)THEN
              bip=bip+1
          ELSE
              END IF
          ENDDO
              IF(bip.EQ.0)CYCLE ReductionV
              cntV=cntV+1
              AkjV(cntV,0:mtat)=Akj(k,0:mtat)
ENDDO ReductionV
          AkjH(:,0)=AkjV(:,0)
              cntH=0
ReductionH:DO J=1,mtat
          IF(Akj(Ncal,J).EQ.0.)CYCLE ReductionH
            cntH=cntH+1
          AkjH(1:cntV,cntH)=AkjV(1:cntV,J)
ENDDO ReductionH
```

```
          contVoX=cntV+cntH
END   SUBROUTINE MATRICE

                ************************************
                *      Sous-programme PHASE2         *
                ************************************



SUBROUTINE PHASE2(Nxch,BkjH,mtat,Ncolo,contH,contV,contVoX,&
                    Utal,X,Z,colopt,Option)
IMPLICIT NONE
INTEGER          ::col,ct,mtat,Ncolo,Utal,slack,flag,dep,colopt
INTEGER          ::I,J,k,contV,contH,contVoX,line,Nxch,row,nt
REAL,DIMENSION(1:Ncolo,0:mtat)      ::BkjH
REAL,DIMENSION(1:contV,0:contH)     ::AkjInit,Akj1_2
REAL,DIMENSION(1:contV,1:contH)     ::Akjou
REAL,DIMENSION(1:contV,0:(contV))   ::Obkj
REAL,DIMENSION(1:contV,1)           ::ObRow
REAL,DIMENSION(1:contVoX,0:contH)   ::AkjoXi,SkjoX
INTEGER,DIMENSION(1)                ::N
INTEGER,DIMENSION(1:(contVoX-1))    ::klax
INTEGER,DIMENSION(1:Utal)           ::Option
REAL,DIMENSION(1:contVoX,0:contH)   ::PRow
REAL,DIMENSION(1:contH)             ::Ratio
REAL,DIMENSION(1:(contVox-1))       ::Y
REAL,DIMENSION(1:Utal)              ::X
REAL                                ::Z,coef,ZoX

  Option=0
  colopt=0

          !!!**************************
          !!!!Tableau de simplex reduit

  DO I=0,contH
             DO k=1,contV
               AkjInit(k,I)=BkjH(k,I)
               ENDDO
  ENDDO
  Toto: DO k=(Utal+1),(contV-1)
             DO I=0,contH
                 IF(AkjInit(k,I).EQ.(-1.))EXIT Toto
               END DO
               IF(k.EQ.(contV-1))THEN
                 Akj1_2=AkjInit
                 GOTO 270
               END IF
  END DO Toto
```

```
          !!!!****************************************
          !!!!Tableau avec des variables artificielles

            DO k=1,contV
              ObRow(k,1)=-1.*(SUM(AkjInit(k,1:contH)))
          END DO
            DO k=1,contVoX
            IF(k.LE.(contV-1))THEN
                   AkjoXi(k,0)=ObRow(k,1)
               ELSE
                  AkjoXi(k,0)=0.
               ENDIF
            END DO
            AkjoXi(contVoX,0)=ObRow(contV,1)

DO J=1,contH
            slacK=(contV-1)+J
            DO k=1,contVoX
                 IF(k.LE.(contV-1))THEN
               AkjoXi(k,J)=AkjInit(k,J)
             ELSE
               AkjoXi(k,J)=0.
                 END IF
              END DO
            AkjoXi(slack,J)=1
            AkjoXi(contVoX,J)=AkjInit(contV,J)
ENDDO

            !!!************************************
            !!!!!RECHERCHE DE LA VARIABLE D'ENTRÉE

              col=0
              flag=1
              dep=1
80 VarEntree: DO k=dep,(contVoX-1)
            IF(flag.EQ.200)GOTO 550
            IF(AkjoXi(k,0).LT.(-1E-3)) EXIT VarEntree
              IF(k.EQ.(contVoX-1))GOTO 620
END DO VarEntree

              col=k
              !!************************************
            !!!Verification que la colonne contient
          !!!au moins une valeur positive

Nosol2:DO I=1,contH
              IF(AkjoXi(col,I).GT.1E-3)EXIT Nosol2
              IF(I.EQ.contH)THEN
                 IF(col.LT.(contVoX-1))THEN
                      dep=col+1
                      GOTO 80
```

```
                ELSE
                   GOTO 550
                END IF
             END IF
END DO Nosol2

         !!!**********************************
           !!!!!LOCALISATION DE LA LIGNE PIVOT

search2: DO I=1,contH
              IF(AkjoXi(col,I).GT.1E-3)THEN
                 Ratio(I)=AkjoXi(contVoX,I)/AkjoXi(col,I)
           ELSE
                 Ratio(I)=1E6
              END IF
END DO search2

         !!*********************************************
           !!!!!! REDUCTION DE LA VARIABLE PIVOT A L'UNITE

         N=MINLOC(Ratio,MASK=Ratio.GE.0.0 )
             DO k=1,contVoX
               PRow(k,N(1))=AkjoXi(k,N(1))/(AkjoXi(col,N(1)))
             END DO
         DO k=1,contVoX
              AkjoXi(k,N(1))=PRow(k,N(1))
         END DO

         !!!*************************
         !!!!!!!PIVOTAGE DES LIGNES

Search3: DO I=0,contH
         IF(I.EQ.N(1))CYCLE Search3
           coef=AkjoXi(col,I)
           DO k=1,contVoX
                PRow(k,N(1))=coef*AkjoXi(k,N(1))
            END DO

              DO k=1,contVoX
                AkjoXi(k,I)=AkjoXi(k,I)-PRow(k,N(1))
              END DO
END DO search3
         IF(AkjoXi(contVoX,0).EQ.0.)GOTO 625
           col=0
           flag=flag+1
           dep=1
         GOTO 80
550     Z=0.0
           X=0.
           GOTO 1700
           !!!*****************************
```

```
          !!!!!AFFICHAGE DES RESULTATS
620 Feasible: DO k=1,(contVox-1)
          IF(AkjoXi(k,0).LT.(-1E-4))GOTO 550
ENDDO Feasible
625       SkjoX=AkjoXi
            ZoX=SkjoX(contVoX,0)
            IF(ABS(ZoX).GT.1E-3)THEN
               Z=0.0
               X(:)=0.
               GOTO 1700
            END IF
              !!!********************
          !!! VARIABLES BASIC
            nt=0
              klax=0
DO k=1,(contVoX-1)
          ct=0
   Search5: DO I=0,contH
            IF(ABS(SkjoX(k,I)).GT.1E-3)THEN
                  IF(SkjoX(k,I).EQ.1.)THEN
                     row=I
                  ENDIF
                     ct=ct+1
            END IF
     END DO search5
           IF(ct.EQ.1)THEN
                  nt=nt+1
                  klax(nt)=row
                  Y(k)=SkjoX(contVoX,klax(nt))
          ELSE
                   Y(k)=0.
          END IF
END DO

          !!!*********************************************
          !!!Tableau final sans les varaibles auxiliaires

DO I=1,contH
            DO k=1,contV
                  IF(k.LE.(contV-1))THEN
                     Akjou(k,I)=SkjoX(k,I)
                  ELSE
                     Akjou(k,I)=SkjoX(contVoX,I)
                  ENDIF
              END DO
ENDDO
              Obkj(:,1)=AkjInit(:,0)
          line=1
              nt=0
Obro:      DO k=1,(contV-1)
            IF(Y(k).LE.0.)CYCLE Obro
```

```
                line=line+1
                nt=nt+1
                Obkj(:,line)=-1*AkjInit(k,0)*Akjou(:,klax(nt))
  END DO Obro
                DO k=1,contV
                 Obkj(k:,0)=SUM(Obkj(k,1:(contH+1)))
                END DO
  DO I=1,contH
                DO k=1,contV
                 Akj1_2(k,I)=Akjou(k,I)
                END DO
  END DO
                 Akj1_2(:,0)=Obkj(:,0)
  270     CALL SIMPLEX(Akj1_2,contH,contV,Utal,Nxch,X,Z,colopt,Option)

1700      END SUBROUTINE PHASE2




                ************************************
                *      Sous-programme SIMPLEX        *
                ************************************


SUBROUTINE SIMPLEX(Akj2F,cntH,cntV,Utol,NAxch,X,Z,colopt,Option)
IMPLICIT NONE
INTEGER ::col,ct,Utol,NAxch,row,colopt,I,k,cntV,cntH,flag,dep,clone
INTEGER,DIMENSION(1)                ::N
INTEGER,DIMENSION(1:Utol)           ::Option
REAL,DIMENSION(1:cntV,0:cntH)       ::Akj2F,Skj2F,PRow
REAL,DIMENSION(1:cntH)              ::Ratio
REAL,DIMENSION(1:Utol)              ::X
REAL                                ::Z,coef,verif
           colopt=0
               Option=0
             !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
             !!!!!RECHERCHE DE LA VARIABLE D'ENTRÉE
               col=0
               flag=1
               dep=1
  80 VarEntree: DO k=dep,(cntV-1)
             IF(flag.EQ.200)GOTO 550
             IF(Akj2F(k,0).LT.(-1E-3)) EXIT VarEntree
               IF(k.EQ.(cntV-1))GOTO 620
  END DO VarEntree
               col=k
               !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
             !!!Verification que la colonne contient
           !!!au moins une valeur positive
```

```
85 Nosol2:DO I=1,cntH
              IF(Akj2F(col,I).GT.1E-3)EXIT Nosol2
              IF(I.EQ.cntH)THEN
                IF(col.LT.(cntV-1))THEN
                  dep=col+1
                  GOTO 80
                  ELSE
                  Skj2F=Akj2F
                  GOTO 550
                  END IF
              END IF
 END DO Nosol2
          !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
            !!!!!LOCALISATION DE LA LIGNE PIVOT
 search2:DO I=1,cntH
              IF(Akj2F(col,I).GT.1E-3)THEN
              Ratio(I)=Akj2F(cntV,I)/Akj2F(col,I)
          ELSE
              Ratio(I)=1E6
              END IF
 END DO search2
          !!!!!!!!!!!!!!1!!!!!!!!!!!!!!!!!!!!!!!!!!!1!!!!
            !!!!!! REDUCTION DE LA VARIABLE PIVOT A L'UNITE

          N=MINLOC(Ratio,MASK=Ratio.GE.(-1E-4) )
              DO k=1,cntV
                PRow(:,N(1))=Akj2F(:,N(1))/(Akj2F(col,N(1)))
              END DO
            DO k=1,cntV
                  Akj2F(k,N(1))=PRow(k,N(1))
            END DO
          !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
          !!!!!!!PIVOTAGE DES LIGNES
 Search3: DO I=0,cntH
          IF(I.EQ.N(1))CYCLE Search3
            coef=Akj2F(col,I)
            DO k=1,cntV
                  PRow(k,N(1))=coef*Akj2F(k,N(1))
              END DO
              DO k=1,cntV
                Akj2F(k,I)=Akj2F(k,I)-PRow(k,N(1))
              END DO
 END DO search3
            col=0
            dep=1
            flag=flag+1
          GOTO 80
 550      Z=0.0
            X(:)=0.
            GOTO 1700
 620      Skj2F=Akj2F
```

```
            !!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!!
            !!AFFICHAGE DES RESULTATS
  Feasible: DO k=1,(cntV-1)
            IF(Skj2F(k,0).LT.(-1E-4))GOTO 550
  ENDDO Feasible
            Z=Skj2F(cntV,0)
  DO I=1,cntH
            IF(Skj2F(cntV,I).LT.(-1E-4))THEN
                Z=0.0
                X(:)=0.
                GOTO 1700
            END IF
  END DO
              colopt=0
  Base: DO k=1,Utol
          clone=0
          ct=0
          IF(ABS(Skj2F(k,0)).GT.1E-3)CYCLE Base
     Search5: DO I=1,cntH
              IF(ABS(Skj2F(k,I)).GT.1E-3)THEN
                IF(Skj2F(k,I).EQ.1.)THEN
                row=I
                ENDIF
                ct=ct+1
              ELSE
                clone=clone+1
          END IF
     END DO search5
             IF(ct.EQ.1)THEN
               X(k)=Skj2F(cntV,row)
          ELSE
              X(k)=0.
                   IF(clone.LT.cntH)THEN
                colopt=colopt+1
                Option(colopt)=k
                  ENDIF
          END IF
  END DO Base
          verif=SUM(X(1:NAxcH))
            IF(ABS(Z-verif).GT.1E-2)THEN
            Z=0.
            X(:)=0.
            ENDIF
1700  END SUBROUTINE SIMPLEX
```

```
                ************************************
                *      Sous-programme TEMPERATURE    *
                ************************************


SUBROUTINE TEMPERATURE(Tim,HAN,Iy,cycl,daltaT,Bakj,BaH,BaC,&
                       Utal,ega,Nkay,kay,NAxch,Ncal,Cap,Tie,&
                           Tos,eX,zed,Toh,Tih,Toc,Tic,Q,duties)
IMPLICIT NONE
INTEGER                                 ::Nkay,kay,NAxch,Ncal,Utal,HAN,Tim
INTEGER                                 ::J,k,L,str1,str2,ega,cnt,Iy,cycl
REAL                                    ::zed,daltaT,som
INTEGER,DIMENSION(1:NAxch,1:Nkay)  ::BaH
INTEGER,DIMENSION(1:NAxch,1:kay)   ::BaC
REAL ,DIMENSION(1:ega)                  ::Cap,Tie,Tos
REAL ,DIMENSION(1:NAxch)                ::Tsoh,Tenh,Tsoc,Tenc,epsio,epsie
REAL ,DIMENSION(1:Utal)                 ::eX
REAL ,DIMENSION(1:Ncal,1:ega)           ::Bakj
REAL,DIMENSION(1:NAxch,1:NAxch)         ::dTh,dTc
REAL ,DIMENSION(1:HAN,1:cycl)           ::Q
REAL,DIMENSION(1:Utal,1:HAN,1:cycl)  ::duties
REAL,DIMENSION(1:NAxch,1:HAN,1:cycl)::Tih,Toh,Tic,Toc

DO J=1,ega
      som=0.
      IF(J.LE.Nkay)THEN
          DO k=1,NAxch
              som=som+eX(k)*BaH(k,J)
            END DO
              eX(NAxch+J)=(Tie(J)-Tos(J))*Cap(J)-som
      ELSE
          DO k=1,NAxch
              som=som+eX(k)*BaC(k,(J-Nkay))
            END DO
              eX(NAxch+J)=(Tos(J)-Tie(J))*Cap(J)-som
      ENDIF
ENDDO

           cnt=0
Inequation1: DO k=1,NAxch
           IF(eX(k).LE.1E-3)CYCLE Inequation1

  !!*********************************************************
  !!Recherche des courants chaud  et froid  de l'echangeur k
  Ligne2: DO J=1,Nkay
           IF(Bakj(k,J).GT.0.0)THEN
               str1=J
               EXIT Ligne2
               ELSE
                   IF(J.EQ.Nkay)THEN
```

```
                    GOTO  100
                    ELSE
                    END IF
                END IF
  END DO Ligne2

  Ligne3: DO J=(Nkay+1),ega
           IF(Bakj(k,J).GT.0.0)THEN
               str2=J
               EXIT Ligne3
               ELSE
               END IF
  END DO Ligne3

  !!*****************************************************************
  !!Calcul temperatures extremite chaude et froide de l'echangeur k
  DO L=1,k
      dTh(L,k)=BaH(L,str1)*eX(L)/Cap(str1)
      dTc(L,k)=BaC(L,(str2-Nkay))*eX(L)/Cap(str2)
  END DO

  Tsoh(k)=Tos(str1)+SUM(dTh(1:(K-1),K))+eX(NAxch+str1)/Cap(str1)
  Tenh(k)= Tsoh(k)+dTh(k,k)
  Tenc(k)=Tie(str2)+SUM(dTc(1:(K-1),k))
  Tsoc(k)=Tenc(k)+dTc(k,k)
  epsio(k)=Tsoh(k)-Tenc(k)
  epsie(k)=Tenh(k)-Tsoc(k)

  IF(epsio(k).LT.(daltaT-0.1).OR.epsie(k).LT.(daltaT-0.1)) THEN
         Q(Tim,Iy)=0.
       duties(:,Tim,Iy)=0.
         Toh(:,Tim,Iy)=0.
       Tih(:,Tim,Iy)=0.
       Toc(:,Tim,Iy)=0.
       Tic(:,Tim,Iy)=0.
       GOTO 1000
  ELSE
  END IF
100 END DO Inequation1
         duties(:,Tim,Iy)=eX(:)
         Q(Tim,Iy)=Zed
         Toh(:,Tim,Iy)=Tsoh(:)
       Tih(:,Tim,Iy)=Tenh(:)
       Toc(:,Tim,Iy)=Tsoc(:)
       Tic(:,Tim,Iy)=Tenc(:)

1000  END SUBROUTINE TEMPERATURE
```

```
                  ************************************
                  *      Sous-programme ÉLITISME       *
                  ************************************


SUBROUTINE ELITISME(Iy,cycl,NAxch,HAN,chrama,Gaadfit,Qi,Elite)
IMPLICIT NONE
INTEGER                                  ::T,ct,El1
INTEGER                                  ::NAxch,HAN,cycl,Iy
INTEGER,DIMENSION(1:NAxch,1:2)           ::Elite
INTEGER,DIMENSION(1)                     ::El2
INTEGER,DIMENSION(1:NAxch,1:HAN)         ::chrama
REAL ,DIMENSION(1:cycl)                  ::Gaadfit
REAL ,DIMENSION(1:HAN)                   ::comp
REAL ,DIMENSION(1:HAN,1:cycl)            ::Qi

Elitism:DO T=1,HAN
                IF(Qi(T,Iy).EQ.Gaadfit(Iy))THEN
                    El1=T
                    EXIT Elitism
                END IF
END DO Elitism
                Elite(:,1)=chrama(:,El1)
                DO T=1,HAN
                    IF(T.NE.El1)THEN
                      comp(T)=Qi(El1,Iy)-Qi(T,Iy)
                    ELSE
                      comp(T)=0.
                    END IF
            END DO
                ct=0
                DO T=1,HAN
                  IF(comp(T).LE.1E-3)THEN
                    ct=ct+1
                  ENDIF
                END DO
            IF(ct.EQ.HAN)THEN
               Elite(:,2)=Elite(:,1)
                ELSE
                    El2=MINLOC(comp,MASK=comp.GT.0.)
                    Elite(:,2)=chrama(:,El2(1))
            END IF

END SUBROUTINE ELITISME
```

```
                ************************************
                *      Sous-programme CROISEMENT     *
                ************************************


 SUBROUTINE CROISEMENT(Ox,Elute,chrama,Ia,sycle,NAxch,HAN,Nkay,&
                          somqrac,Qa,Pcro,Pmu,child)
IMPLICIT NONE
INTEGER            ::H,Ox,m,Ia,J,k,T,Ti,range1,range2,HAN,NAxch
INTEGER            ::Pos1,Pos2,ct,Pos,sycle,Nkay
INTEGER,DIMENSION(1:2)  ::N
INTEGER,DIMENSION(1:HAN)       ::Discrim
INTEGER,DIMENSION(1:NAxch,1:HAN)::chrama,child,childm
INTEGER,DIMENSION(1:NAxch,1:2)  ::Elute
REAL         ::somal,som,somqrac,Site,Pcro,Pmu
REAL,DIMENSION(1:HAN)     ::R
REAL,DIMENSION(1:2*HAN)   ::alter
REAL,DIMENSION(1:HAN)     ::w1
REAL,DIMENSION(1:HAN,1:sycle)::Qa

CALL RANDOM_SEED(  )
CALL RANDOM_NUMBER( R )
CALL RANDOM_SEED(  )
CALL RANDOM_NUMBER( alter )
CALL RANDOM_SEED(  )
CALL RANDOM_NUMBER( w1 )


            range2=Ox
                ct=0
            k=1
                N=0
search1: DO Ti=1,(HAN-Ox)/2
            J=1
  100       somal=R(k)*somqrac
            som=0.
     search2:DO T=1,HAN
            som=som+Qa(T,Ia)
            IF(som.GE.somal)EXIT search2
     END DO search2
                k=k+1
            N(J)=T
                IF(N(J).GT.HAN) N(J)=HAN
              IF(J.LT.2)THEN
                 J=J+1
              GOTO 100
              END IF
              !!************************
              !Probabilité de croisement
            IF(R(Ti).GT.Pcro) THEN
```

```
            Pos=0
            GOTO 200
          END IF

        !!!!Choix du site de croisement
        Site=NAxch*R(Ti)
        Pos=AINT(site)

200     CALL RANDOM_SEED(  )
        DO H=1,5E6
          ENDDO
        CALL RANDOM_NUMBER( alter )

        !!!*************************
        !!Croisement des chromosomes
        range1=range2+1
        DO m=NAxch,1,-1
          IF(m.GT.Pos) THEN
            childm(m,range1)=chrama(m,N(1))
              child(m,range1)=chrama(m,N(1))
          ELSE
            childm(m,range1)=chrama(m,N(2))
              child(m,range1)=chrama(m,N(2))
          END IF
        END DO

          range2=range1+1
        DO m=NAxch,1,-1
          IF(m.GT.Pos) THEN
            childm(m,range2)=chrama(m,N(2))
              child(m,range2)=chrama(m,N(2))
          ELSE
            childm(m,range2)=chrama(m,N(1))
              child(m,range2)=chrama(m,N(1))
          END IF
        END DO

        !!**********************
        !Mutation des chromosomes
          ct=ct+1
        IF(w1(range1).LE.Pmu)THEN
           Pos1=AINT(NAxch*alter(ct))+1
             ct=ct+1
           Pos2=AINT(NAxch*alter(ct))+1
           child(Pos1,range1)=childm(Pos2,range1)
           child(Pos2,range1)=childm(Pos1,range1)
        END IF
          ct=ct+1
        IF(w1(range2).LE.Pmu)THEN
           Pos1=AINT(NAxch*alter(ct))+1
             ct=ct+1
```

```
                Pos2=AINT(NAxch*alter(ct))+1
                child(Pos1,range2)=childm(Pos2,range2)
                child(Pos2,range2)=childm(Pos1,range2)
            END IF
END DO search1

  DO T=Ox+1,HAN
             Discrim(T)=SUM(child(:,T))
             IF(Discrim(T).EQ.0)THEN
                DO k=1,NAxch,2
                  child(k,T)=AINT((Nkay+0.5)*alter(k))
                ENDDO
             END IF
  END DO
           IF(Ox.EQ.0)GOTO 400
           child(:,1)=Elute(:,1)
           child(:,2)=Elute(:,2)

400   END   SUBROUTINE CROISEMENT



                ************************************
                *       Sous-programme SOLUTION      *
                ************************************


SUBROUTINE SOLUTION(celept,eptien,NaH,NaC,Tpanh,Tpanc,QCnood,QHnood,&
                    Qmox,Laval,Utal,sycl,rat,NAxch,HAN,Nkay,&
                        Kay,doties,Gaadfit,Qi,chramRef,Tinh,Toth,Tinc,&

      Totc,Exces,Ncolopt,Refcolopt,Resopt,OptiHEN,QREC)
IMPLICIT NONE
INTEGER  ::Laval,Utal,sycl,HAN,NAxch,Nkay,Kay,rat,NaH,NaC
INTEGER  ::I,L,J,k,T,ct,kt,Nbex,Nbc,Nbh,s,coldu
INTEGER  ::Hotu,Nunit,Ncolopt,Exces,resopt,Iteropt
INTEGER,DIMENSION(2)    ::Reup
REAL                    ::QREC,QCnood,QHnood,Qmox,Tpanh,Tpanc
REAL                                  ::QH,QC
REAL,DIMENSION(1:Utal,1:HAN,1:sycl)   ::doties
REAL,DIMENSION(1:NAxch,1:HAN,1:sycl)  ::Tinh,Toth,Tinc,Totc
REAL,DIMENSION(1:NAxch)::OptiTsoh,OptiTenh,OptiTsoc,OptiTenc
REAL,DIMENSION(1:Utal)                ::OptiX
REAL,DIMENSION(1:sycl)                ::Gaadfit
REAL,DIMENSION(1:HAN,1:sycl)          ::Qi
INTEGER,DIMENSION(1:HAN,1:sycl)       ::celept
INTEGER,DIMENSION(1:NAxch,1:HAN)      ::chroumou,OptiHEN
INTEGER,DIMENSION(1:NAxch,1:HAN*sycl)::chromo,chromidem
INTEGER,DIMENSION(1:NAxch,1:HAN,1:sycl)::chramRef
```

```
INTEGER,DIMENSION(1:Utal,1:HAN,1:sycl) ::eptien
INTEGER,DIMENSION(1:Utal)              ::Refcolopt


QREC=MAXVAL(Gaadfit,MASK=Gaadfit.GT.0.)
Reup=MaxLoc(Qi)
Resopt=Reup(1)
Iteropt=Reup(2)
chroumou(:,Resopt)=chramRef(:,Resopt,Iteropt)
OptiX(:)=doties(:,Resopt,Iteropt)
OptiTenh(:)=Tinh(:,Resopt,Iteropt)
OptiTsoh(:)=Toth(:,Resopt,Iteropt)
OptiTenc(:)=Tinc(:,Resopt,Iteropt)
OptiTsoc(:)=Totc(:,Resopt,Iteropt)
Ncolopt=celept(Resopt,Iteropt)
Refcolopt(:)=eptien(:,Resopt,Iteropt)

  IF((QREC-Qmox).GT.0.25)THEN
  Exces=1
  GOTO 2000
  ELSE
  Exces=0
  ENDIF
  WRITE(*,'')
  DO k=1,NAxch
      IF(OptiX(k).GT.1E-3)THEN
              OptiHEN(k,Resopt)=chroumou(k,Resopt)
        ELSE
              OptiHEN(k,Resopt)=0
        END IF
  END DO
        IF(QHnood.GT.1E-3)THEN
              Hotu=1
        ELSE
              Hotu=0
        ENDIF
      IF(QCnood.GT.1E-3)THEN
              Coldu=1
        ELSE
              Coldu=0
        ENDIF
              Nunit=(NaH+Hotu-1)+(NaC+Coldu-1)

  IF(rat.EQ.1)THEN
        WRITE(*,*)'L''ANALYSE DES DONNEES DU PROBLEME INDIQUE : '
        WRITE(*,*)'----------------------------------------'
        WRITE(*,250)QHnood
      WRITE(*,300)QCnood
        WRITE(*,320)Qmox
        WRITE(*,325)Nunit
  250 FORMAT('Besoins en Rechauffeur: QHmin=',F8.3)
```

```
  300 FORMAT(' Besoins Refroidisseur: QCmin=',F8.3)
  320 FORMAT(' Puissance maximale à recuperee: Qmax =',F8.3)
  325 FORMAT('Nombre minimum d''unites : Nunit=',I2)
        WRITE(*,*)''
        WRITE(*,350)Tpanh
        WRITE(*,400)Tpanc
  350 FORMAT('Temperature de pincement courants chauds:Tph =',F8.3)
  400 FORMAT('Temperature de pincement courants froid : Tpc=',F8.3)
        WRITE(*,*)''
END IF
        WRITE(*,*)''
        WRITE(*,420)rat
  420 FORMAT(' RESULTAT TROUVE AU ',I2,'e ESSAI :')
        WRITE(*,*)'*********************************'
        WRITE(*,*)''
        WRITE(*,*)'Meilleure Topologie de Reseau dans cet essai:'
        WRITE(*,*)'----------------------------------------'
        WRITE(*,500)OptiHEN(:,Resopt)
  500 FORMAT(2x,20(2x,I2))
        WRITE(*,*)''
      WRITE(*,*)'Montage d''echangeurs entre:'
        WRITE(*,*)''
      ct=0
        kt=0
  DO L=1,Laval
      DO J=1,Kay
              ct=ct+1
              IF(OptiHEN(ct,Resopt).NE.0)THEN
                kt=kt+1
              WRITE(*,600)kt,J,OptiHEN(ct,Resopt)
  600        FORMAT(I2,') courant froid',I1,' / courant chaud',I1)
              END IF
    END DO
  ENDDO
      WRITE(*,*)''
      WRITE(*,*)'Puissances echangees au niveau des echangeurs '
    WRITE(*,*)'-----------------------------------------------'
      ct=0
  DO k=1,NAxch
          IF(OptiX(k).GT.1E-3)THEN
                ct=ct+1
                WRITE(*,1000)ct,OptiX(k)
              END IF
  1000 FORMAT(' Q(',I2,')  =',F9.3)
  END DO
    Nbex=ct
    WRITE(*,*)''
      WRITE(*,*)'Refroidisseur (Cooling Water) '
      WRITE(*,*)'------------------------------'
      ct=0
      QC=0.
```

```
      Nbc=0
  DO k=(NAxch+1),(NAxch+Nkay)
     IF(OptiX(k).GT.1E-3)Nbc=Nbc+1
      QC=QC+OptiX(k)
      ct=ct+1
      WRITE(*,1100)ct,OptiX(k)
  1100 FORMAT(' QC(',I1,')  =',F9.3)
  END DO
     WRITE(*,*)''
      WRITE(*,*)'Rechauffeur(Steam)'
      WRITE(*,*)'--------------------'
      ct=0
      QH=0.
      Nbh=0
  DO k=(NAxch+Nkay+1),(NAxch+Nkay+Kay)
     IF(OptiX(k).GT.1E-3)Nbh=Nbh+1
      QH=QH+OptiX(k)
      ct=ct+1
      WRITE(*,1200)ct,OptiX(k)
  1200 FORMAT(' QH(',I1,')  =',F9.3)
  END DO
      WRITE(*,*)''
      WRITE(*,*)'                    RECAPITULATIF '
      WRITE(*,*)' ================================================='
      WRITE(*,*)'  No      Q       Thi      ho      Tci       Tco   '
      WRITE(*,*)' ================================================='
      ct=0
  DO k=1,NAxch
      IF(OptiX(k).GT.1E-3)THEN
           ct=ct+1
         WRITE(*,1500)ct,OptiX(k),OptiTenh(k),OptiTsoh(k),&
                OptiTenc(k),OptiTsoc(k)
  1500      FORMAT(2x,I2,4x,F9.3,4x,F9.3,4x,F9.3,4x,F9.3,4x,F9.3)
      END IF
  END DO
     WRITE(*,*)' ================================================='
      WRITE(*,1510)QH,Nbh
      WRITE(*,1520)QC,Nbc
  1510  FORMAT(3x,'QH  =',F9.3,18x,'(',I2,1x,'rechauffeurs)')
  1520  FORMAT(3x,' QC =',F9.3,18x,'(',I2,1x,'refroidisseurs)')
      WRITE(*,1550)QREC,Nbex
  1550  FORMAT(3x,'QREC =',F9.3,18x,'(',I2,1x,'Exchangers)')
      WRITE(*,1560)(QREC/Qmox)*100
  1560      FORMAT(3x,'Taux de Recuperation =',2x,F7.3,1x,'%' )
     WRITE(*,*)''
     IF(ABS(Qmox-QREC).LE.1E-3)THEN
           WRITE(*,1565)(Nbex+Nbh+Nbc),Nunit
  1565   FORMAT(3x,'Nbre d''unites utilisees =',I2,' /',I2)
      END IF
      WRITE(*,*)' ================================================='
!!!*************************************************************
```

```
!!!Recherche de chromosomes differents donnant le meme QREC
IF(ABS(Qmox-QREC).LE.1E-3)THEN
      ct=0
  DO I=1,sycl
       DO T=1,HAN
              IF(Qi(T,I).EQ.QREC)THEN
              ct=ct+1
              chromo(:,ct)=chramRef(:,T,I)
                DO k=1,NAxch
                     IF(doties(k,T,I).GT.1E-3)THEN
                       chromo(k,ct)=chramRef(k,T,I)
                     ELSE
                       chromo(k,ct)=0
                     END IF
                  ENDDO
          ENDIF
         END DO
  END DO
TITI1:DO I=1,ct-1
              IF(SUM(chromo(:,I)).EQ.0) CYCLE TITI1
  TITI2:DO J=I+1,ct
              IF(SUM(chromo(:,J)).EQ.0) CYCLE TITI2
            DO k=1,NAxch
            IF(chromo(k,J).NE.chromo(k,I)) CYCLE TITI2
              IF(k.EQ.NAxch) chromo(:,J)=0
              END DO
  END DO TITI2
END DO TITI1

          s=0
      DO J=2,ct
              IF(SUM(chromo(:,J)).NE.0)THEN
              s=s+1
                chromidem(:,s)=chromo(:,J)
              ENDIF
        END DO
              IF(s.GE.1)THEN
                WRITE(*,*)' Autres combinaisons optionnelles'
                WRITE(*,*)'  --------------------------------'
                  DO k=1,s
                  WRITE(*,1700)chromidem(:,k)
  1700              FORMAT(20(2x,I2))
              ENDDO
              END IF
           WRITE(*,*)''
ENDIF

 IF(rat.EQ.2)THEN
   WRITE(*,*)'Thi:Temperature courant chaud entree de l''echangeur'
   WRITE(*,*)'Tho:Temperature courant chaud sortie de l''echangeur'
   WRITE(*,*)'Tci:Temperature courant froid entree de l''echangeur '
```

```
   WRITE(*,*)'Tco:Temperature courant froid sortie de l''echangeur '
   WRITE(*,*)'Q  :Puissance echangee par l''echangeur  '
   WRITE(*,*)'QH :Puissance totale des Rechauffeurs (Heaters)  '
   WRITE(*,*)'QC :Puissance totale des Refroidisseurs (Coolers) '
   WRITE(*,*)'QREC:Puissance recuperee'
END IF

2000 END SUBROUTINE SOLUTION
```