



HAL
open science

Contribution aux méthodes hybrides d'optimisation heuristique : Distribution et application à l'interopérabilité des systèmes d'information

Norelislam El Hami

► **To cite this version:**

Norelislam El Hami. Contribution aux méthodes hybrides d'optimisation heuristique : Distribution et application à l'interopérabilité des systèmes d'information. Autre [cs.OH]. INSA de Rouen; École Mohammadia d'ingénieurs (Rabat, Maroc), 2012. Français. NNT : 2012ISAM0009 . tel-00771360

HAL Id: tel-00771360

<https://theses.hal.science/tel-00771360>

Submitted on 8 Jan 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N° d'ordre (Rabat) : xxxx

N° d'ordre (Rouen) : xxxx

Université Mohammed V - Agdal
École Mohammadia d'Ingénieurs
"Laboratoire d'Étude et Recherche en
Mathématiques Appliquées" .Rabat - Maroc
Informatique pour l'Ingénieur

Institut National des Sciences
Appliquées de Rouen
Équipe d'Accueil : "Laboratoire
d'Informatique, du Traitement de
l'Information et des Systèmes"

THÈSE DE DOCTORAT EN COTUTELLE

pour obtenir le grade de

DOCTEUR EN INFORMATIQUE

Informatique et Traitement de l'Information et des Systèmes (LITIS)
INSTITUT NATIONAL DES SCIENCES APPLIQUÉES DE ROUEN

et le grade de

DOCTEUR EN SCIENCES

Mathématiques appliquées et Informatique (LERMA)
ECOLE MOHAMMADIA DES INGENIEURS

présentée par

Norelislam EL HAMI

Sujet de la thèse :

***CONTRIBUTION AUX MÉTHODES HYBRIDES D'OPTIMISATION
HEURISTIQUE. DISTRIBUTION ET APPLICATION À L'INTEROPÉRABILITÉ
DES SYSTÈMES D'INFORMATION***

Soutenue publiquement le 23 *Juin* 2012 devant la commission d'examen composée de :

M. Driss ABOUTAJDINE	Professeur, Université Mohammed V Agdal	Président
Mme. Rajae ABOULAICH	Professeur, EMI de Rabat	Rapporteur
M. Abderahmane HABBAL	Professeur, Université de Nice Sophia Antipolis	Rapporteur
Mme. Souad EL BERNOUSSI	Professeur, Université Mohammed V Agdal	Rapporteur
M. Habib ABDULRAB	Professeur, INSA de Rouen	Examineur
M. Rachid ELLAIA	Professeur, EMI de Rabat	Directeur de thèse
M. Mhamed ITMI	MCF. HDR, INSA de Rouen	Directeur de thèse

REMERCIEMENTS

J'exprime ma profonde gratitude à Monsieur Rachid ELLAIA, Professeur à l'EMI de Rabat pour son encadrement au bon développement de mes travaux. Ses conseils en matière d'enseignement et recherche ont été toujours précieux. Je suis heureux de pouvoir lui exprimer ici mes remerciements.

Toute ma reconnaissance à Monsieur Mhamed ITMI, MCF. HDR à l'INSA de Rouen pour m'avoir accueilli en France dans des conditions favorables, sa disponibilité et sa rigueur lors de la direction exemplaire de ce travail de thèse.

Je suis particulièrement honoré de l'attention que Monsieur Driss ABOUTAJDINE professeur à l'université Mohammed V Agdal a bien voulu être le président du jury de ce mémoire de thèse.

Je tiens à remercier vivement Mme. Rajae ABOULAICH, elle a mis en place les conditions propices au bon développement de mes travaux au sein du laboratoire LERMA et d'avoir accepté d'être rapporteur de cette thèse.

Je remercie les membres du Jury, le Professeur M. Abderahmane HABBAL à l'université de Nice, Professeur Souad EL BERNOUSSI à l'université Mohammed V Agdal et le Professeur Habib ABDULRAB à l'INSA de Rouen, qui m'ont fait l'honneur de juger ce travail. Je suis pleinement reconnaissant pour leur participation à ce jury.

Chacun des membres des deux laboratoires LERMA à Rabat et LITIS à Rouen mérite ma reconnaissance pour leurs disponibilités et le partage des connaissances. L'équipe a ainsi largement contribué au bon déroulement de mes travaux.

Mes remerciements s'adressent aux membres de ma famille, petits et grands. Leur encouragement et leur aide étaient bien bénéfiques pour la réalisation du présent travail.

Finalement, je tiens à remercier tous ceux qui de près ou de loin ont contribué au bon déroulement de mes travaux, qu'ils reçoivent par ces quelques lignes toute ma reconnaissance.

RÉSUMÉ

Résumé. Les travaux présentés dans ce mémoire proposent une nouvelle méthode d'optimisation globale dénommée MPSO-SA. Cette méthode hybride est le résultat d'un couplage d'une variante de l'algorithme par Essaim de Particules nommé MPSO (Particle Swarm Optimisation) avec la méthode du recuit simulé nommé SA (Simulted Annealing). Les méthodes stochastiques ont connu une progression considérable pour la résolution de problèmes d'optimisation. Parmi ces méthode, il y a la méthode Essaim de Particules (PSO) qui est développée par [Eberhart et Kennedy (1995)]. Quant à la méthode recuit simulé (SA), elle provient du processus physique qui consiste à ordonner les atomes d'un cristal afin de former une structure cristalline parfaite.

Pour illustrer les performances de la méthode MPSO-SA proposée, une comparaison avec MPSO et SA est effectuée sur des fonctions tests connues dans la littérature. La méthode MPSO-SA est utilisée pour la résolution des problèmes réels interopérabilité des systèmes d'information, ainsi qu'aux problèmes d'optimisation et de fiabilité des structures mécaniques.

Mots clés. Optimisation globale, algorithme hybride, optimisation par essaim de particules, recuit simulé, interopérabilité, optimisation des structures.

Abstract. The work presented in this Phd thesis contributes to a new method for a modified particle swarm optimization algorithm (MPSO) combined with a simulated annealing algorithm (SA). MPSO is known as an efficient approach with a high performance of solving optimization problems in many research fields. It is a population intelligence algorithm [Eberhart et Kennedy (1995)] inspired by social behavior simulations of bird flocking. Considerable research work on classical method PSO (Particle Swarm Optimization) has been done to improve the performance of this method. Therefore, the proposed hybrid optimization algorithms MPSOSA use the combination of MPSO and simulated annealing SA. This method has the advantage to provide best results comparing with all heuristics methods PSO and SA. In this matter, a benchmark of eighteen well-known functions is given. These functions present different situations of finding the global minimum with gradual difficulties. Numerical results presented, in this paper, show the robustness of the MPSO-SA algorithm. Numerical comparisons with three algorithms namely, Simulated Annealing, Modified Particle swarm optimization and MPSO-SA show that the hybrid algorithm offers better results. This method (MPSO-SA) treats a wide range of optimization problems, in information systems interoperability and in structural optimization field.

Keywords. Global optimization, hybrid algorithm, Particle swarm optimisation, simulated annealing, interoperability, structural optimization.

TABLE DES MATIÈRES

TABLE DES MATIÈRES	4
LISTE DES FIGURES	7
PRÉSENTATION GÉNÉRALE DE LA THÈSE - OBJECTIF ET ORGANISATION	9
1 ÉTAT DE L'ART DES MÉTHODES D'OPTIMISATION	13
INTRODUCTION	14
1.1 TECHNIQUES D'OPTIMISATION	15
1.1.1 Optimisation sans contraintes	15
1.1.2 Optimisation avec contraintes	15
1.1.3 Optimisation multiobjectif	18
1.2 CLASSIFICATION DES MÉTHODES D'OPTIMISATION	18
1.2.1 Algorithmes énumératifs	19
1.2.2 Algorithmes déterministes	19
1.2.3 Algorithmes stochastiques	20
1.2.4 Méthodes hybrides	22
1.2.5 Choix d'un algorithme d'optimisation	22
1.3 MÉTHODES MÉTAHEURISTIQUES	23
1.3.1 Algorithmes génétiques	23
1.3.1.1 Étapes de l'algorithme	23
1.3.1.2 Implémentation de l'algorithme	25
1.3.2 Méthode d'essaim de particules	26
1.3.2.1 Généralités	26
1.3.2.2 Étapes de l'algorithme :	27
1.3.3 Algorithme de colonie de fourmis	28
1.3.3.1 Introduction	28
1.3.3.2 Étapes de l'algorithme	28
1.3.4 Recuit simulé	29
1.3.5 Recherche tabou	30
CONCLUSION	31
2 HYBRIDATION PROPOSÉE DE L'ESSAIM DE PARTICULES MODIFIÉ AVEC LE RE- CUIT SIMULÉ	32
INTRODUCTION	33
2.1 ALGORITHME RECUIT SIMULÉ (SA)	34
2.1.1 Contexte général	34
2.1.2 Étapes de l'algorithme SA	35
2.2 ALGORITHME D'ESSAIM DE PARTICULES (PSO)	36
2.2.1 Principe et généralités sur la méthode PSO	36

2.2.2	Étapes de l'algorithme PSO classique	39
2.3	ESSAIM DE PARTICULES MODIFIÉ MPSO	40
2.3.1	Introduction	40
2.3.2	Étapes de l'algorithme	41
2.4	ALGORITHME PROPOSÉ : ESSAIM DE PARTICULES COUPLÉ AVEC LE RECUIT SIMULÉ	43
2.5	RÉSULTATS ET PERFORMANCE DE LA MÉTHODE MPSO-SA PROPOSÉE	45
2.6	COMPARAISON DES RÉSULTATS	47
2.6.1	Comparaison des taux de succès	47
2.6.2	Comparaison du temps de calcul	48
2.6.3	Comparaison du nombre d'évaluations de la fonction objectif	48
2.7	MODÉLISATION ET IMPLÉMENTATION SOUS JAVA	49
2.8	AMÉLIORATION DU TEMPS D'EXÉCUTION VIA LES THREADS	51
2.9	INTERFACE HOMME-MACHINE	53
	CONCLUSION	54
3	APPLICATION DE LA MÉTHODE MPSO-SA À LA MÉCANIQUE DES STRUC- TURES	56
	INTRODUCTION	57
3.1	INCERTITUDES EN FIABILITÉ DES STRUCTURES	58
3.2	FIABILITÉ DES STRUCTURES	60
3.2.1	Modélisation d'un problème de fiabilité des structures	60
3.2.2	Probabilité de défaillance d'une structure	61
3.2.3	Calcul de la probabilité de défaillance d'une structure	61
	3.2.3.1 Méthode de Monte-Carlo	62
	3.2.3.2 Indice de fiabilité	63
3.3	RÉSULTATS DES APPLICATIONS DE FIABILITÉ DES STRUCTURES	65
	3.3.1 Exemple 1 : Etude dynamique d'une plaque	65
	3.3.2 Exemple 2 : Etude dynamique d'un ressort d'embrayage	69
	CONCLUSION	72
4	OPTIMISATION DE L'INTEROPÉRABILITÉ DES SYSTÈMES D'INFORMATION	73
	INTRODUCTION	74
4.1	CONTEXTE GÉNÉRAL DE L'INTEROPÉRABILITÉ DES SI	75
4.1.1	Barrières et niveaux d'interopérabilité	75
4.1.2	Approches et aspect de l'interopérabilité	76
4.2	MOYENS DE MISE EN ŒUVRE DE L'INTEROPÉRABILITÉ	77
4.2.1	Adoption des standards	77
4.2.2	Adoption d'un unique système intégré	77
4.2.3	Interfaçage pour l'échange de données	78
4.2.4	Interaction orientée service	78
4.2.5	Composition de processus	78
4.3	MODÈLES DE RÉFÉRENCE DE L'INTEROPÉRABILITÉ DES SI	78
4.3.1	Modèle des niveaux d'interopérabilité des SI (LISI)	79
4.3.2	Modèle de maturité organisationnel de l'interopérabilité (OIMM)	80
4.3.3	Modèle de maturité de l'interopérabilité d'entreprise (EIMM)	81
4.3.4	Niveaux de maturité d'interopérabilité dans le gouvernement numérique GIMM	82
4.4	ÉVALUATION DE L'INTEROPÉRABILITÉ	82
4.4.1	Évaluation et potentiel d'interopérabilité	83

4.4.2	Compatibilité à inter opérer et performances opérationnelles	83
4.5	APPROCHE RATIOP	83
4.5.1	Délimitation de l'étendue de l'étude	84
4.5.2	Quantification du potentiel d'interopérabilité	85
4.5.3	Calcul du degré de compatibilité	86
4.5.4	Évaluation des performances opérationnelles	86
4.5.5	Agrégation du degré d'interopérabilité	87
4.6	OPTIMISATION DE L'INTEROPÉRABILITÉ	88
4.6.1	Modélisation du problème d'interopérabilité	88
4.6.2	Résultat de l'optimisation sur l'interopérabilité	90
4.6.2.1	Interopérabilité de quatre SI	90
4.6.2.2	Interopérabilité de dix SI	92
	CONCLUSION	94
5	DISTRIBUTION PAIR A PAIR DU CALCUL D'OPTIMISATION	95
	INTRODUCTION	96
5.1	CONCEPT DE RÉSEAU PAIR A PAIR ET PLATEFORME JXTA	97
5.1.1	Architecture Client-Serveur	97
5.1.2	Architecture pair à pair	97
5.1.3	La technologie JXTA et JXSE	98
5.1.3.1	La technologie JXTA-JXSE	98
5.1.3.2	Protocoles JXTA	99
5.1.3.3	Architecture JXTA	101
5.1.3.4	Services fournis par l'architecture JXTA	103
5.2	DISTRIBUTION DU CALCUL D'OPTIMISATION	103
5.2.1	Modélisation	103
5.2.2	Stratégie de découpage de l'application	104
5.2.3	Interface utilisateur	105
5.2.4	Implémentation	105
5.3	PREMIERES EXPERIMENTATIONS	106
5.3.1	L'approche BBP, pour calculer Π	106
5.3.2	La distribution de l'algorithme MPSO-SA	107
	CONCLUSION	108
	CONCLUSION GÉNÉRALE	110
A	ANNEXES	113
A.1	LISTE DES PUBLICATIONS	114
A.2	FONCTIONS TESTS MATHÉMATIQUES	115
A.2.1	Fonction de Shubert 3 DIM	115
A.2.2	Fonction de Shubert 4 DIM	115
A.2.3	Fonction de Camel	116
A.2.4	Fonction de Dejoint with Noise	116
A.2.5	Fonction de Goldstein and Price	116
A.2.6	Fonction de Rastrigin	117
A.2.7	Fonction de Shekel	117
A.2.7.1	Fonction de Shekel m=5 SHK5	117
A.2.7.2	Fonction de Shekel m=7 SHK7	118
A.2.7.3	Fonction de Shekel m=10 SHK10	118

A.2.8	Fonction de Rosenbrock	119
A.2.9	Fonction de Branins	120
A.2.10	Fonction de Griewank	120
A.2.11	Fonction de Bohachevsky	121
	A.2.11.1 Fonction de Bohachevsky BO1	121
	A.2.11.2 Fonction de Bohachevsky BO2	121
A.2.12	Fonction de Hansen	121
A.2.13	Fonction de Cosine mixture	121
A.2.14	Fonction de Hartman	122
	A.2.14.1 Hartman pour 3 dimension3 HA3	122
	A.2.14.2 Hartman pour 6 dimensions HA6	123
A.3	MODÈLES DE RÉFÉRENCE DE L'INTEROPÉRABILITÉ DES SI	124
A.3.1	Military Communications and Information Systems Interoperability (MCISM)	124
A.3.2	NATO C3 Technical Architecture Reference Model for Interoperability	124
A.3.3	Layered Interoperability Score (i-Score)	124
A.3.4	Evaluation de l'interopérabilité appliqués aux systèmes d'entreprises	124
A.3.5	Interoperability Assessment Methodology (IAM)	125
A.3.6	Levels of Conceptual Interoperability Model (LCIM)	125
A.3.7	Layers of Coalition Interoperability (LCI)	126
A.3.8	System-of-Systems Interoperability Model (SoSI)	126
A.3.9	Organizational Interoperability Agility Model (OIAM)	126

BIBLIOGRAPHIE

129

LISTE DES FIGURES

1.1	Schéma de l'algorithme.	26
2.1	Optimum local et global (recuit simulé).	34
2.2	Oscillation de la bille (recuit simulé).	35
2.3	Oscillation de la bille (recuit simulé).	35
2.4	L'organigramme de la méthode SA.	37
2.5	Déplacement des particules dans PSO classique.	40
2.6	Déplacement des particules dans PSO Modifié.	41
2.7	L'organigramme de la méthode MPSO	42
2.8	L'organigramme de la méthode MPSO SA	44
2.9	Comparaison des Taux de succès.	48
2.10	Comparaison du temps de calcul.	49
2.11	Comparaison du nombre d'évaluation de la fonction objectif.	49
2.12	Gestion de flux entre les paquetages.	50
2.13	Diagramme de classes du paquetage fonctions.	50
2.14	Diagramme de classes du paquetage MPSOSA.	51
2.15	Exemple d'affichage de la position..	54

3.1	Monte Carlo pour un problème à deux variables aléatoires.	63
3.2	Représentation géométrique de β_{HL} pour un problème à deux variables aléatoires.	65
3.3	Plaque soumise à une force.	66
3.4	Algorithme.	70
3.5	Modélisation du ressort d'embrayage.	71
3.6	Modélisation du ressort d'embrayage par élément fini.	71
3.7	le mode à 3 diamètres de nœuds.	71
3.8	le mode à 4 diamètres de nœuds.	72
4.1	Classification de l'interopérabilité.	75
4.2	Moyens de mise en œuvre de l'interopérabilité	77
4.3	Classification de l'interopérabilité.	84
4.4	Résultats de la matrice d'effort à 4 dimensions.	92
4.5	Interaction de dix systèmes d'information.	93
4.6	Vecteur courant et vecteur cible pour un problème de 10 dimensions.	93
4.7	Matrice d'effort optimisée en respectant les contraintes.	93
5.1	Peer Endpoint et Rendez-Vous Protocol.	101
5.2	Couches JXTA Joseph D. Gradecki.	102
5.3	Diagramme des buts du problème.	103
5.4	Diagramme des buts du problème (suite).	104
5.5	Edge Peer.	106
5.6	RDV Peer.	106
5.7	Communication entre les pairs.	107
5.8	L'algorithme MPSO-SA sur l'architecture pair à pair.	108
A.1	Représentation graphique de Shubert de 2 dimensions	115
A.2	Représentation graphique de la fonction de Dejoint with Noise à 2 dimensions	116
A.3	Représentation graphique de la fonction de Goldstein and Price à 2 dimensions	117
A.4	Représentation graphique de Rastrigin de 2 dimensions	118
A.5	Représentation graphique de la fonction de Rosenbrock à 2 dimensions	119
A.6	Représentation graphique de la fonction de Griewank à 2 dimensions	120
A.7	Représentation graphique de la fonction de Cosine mixture à 2 dimensions . .	122

INTRODUCTION GÉNÉRALE

Cette thèse en cotutelle s'inscrit dans le cadre de la collaboration entre le Laboratoire d'Étude et Recherche en Mathématiques Appliquées (LERMA, École Mohammadia d'Ingénieurs, Université Mohammed V Agdal) et le Laboratoire d'Informatique et Traitement de l'Information et des Systèmes (LITIS, Institut National des Sciences Appliquées de Rouen en France).

Durant les trente dernières années, les chercheurs ont consacré beaucoup d'efforts à l'optimisation des systèmes et à la prise en compte des incertitudes. Ces deux domaines de recherche ont suscité un intérêt grandissant en raison de la complexité des problèmes et de leur importance dans les sciences appliquées. Des difficultés sont rencontrées lors de la résolution de ces problèmes dues à la capacité de modéliser des cas réels, le contexte incertain des problèmes réels, la nature de la fonction "objectif", le coût excessif de l'itération et de l'évaluation de la fonction "objectif",...etc.

L'engouement des métaheuristiques et des algorithmes stochastiques itératifs, dans l'optimisation globale est de plus en plus croissant dans différents domaines de la recherche scientifique. Leurs particularités est qu'ils sont capables de résoudre des problèmes pour lesquels on ne connaît pas de méthode classique plus efficace. L'introduction de l'informatique et des processeurs de calcul dans l'optimisation a révolutionné ce domaine. Elle a beaucoup simplifié l'implémentation des métaheuristiques afin de résoudre un nombre important de problèmes délaissés auparavant vu leur complexité.

L'optimisation globale s'est immiscée dans de nombreux domaines, à savoir l'informatique (interopérabilité des systèmes d'information, minimisation de l'énergie consommée par les appareils informatiques, optimisation de la bande passante des réseaux informatique...), la mécanique (optimisation et fiabilité des structures mécaniques pour les constructeurs

d'automobile et dans le domaine de l'aviation ...), le domaine médical, la chimie...etc. L'optimisation globale est devenue essentielle pour l'augmentation des performances dans tous ces domaines. Seulement les méthodes classiques d'optimisation suit difficilement ce besoin accentué en terme de temps de calcul (parfois en temps réel), de qualité et précision des résultats ainsi que de possibilité d'avoir des algorithmes d'optimisation distribués sur plusieurs processeurs de calcul. C'est dans ce contexte que le besoin de nouvelles méthodes d'optimisation devient nécessaire.

Notre travail a pour objectif, la contribution dans le domaine d'optimisation heuristique afin de surmonter les limites des autres méthodes classiques. Pour cela on a proposé une nouvelle méthode d'optimisation globale dénommée MPSO-SA. Cette méthode hybride est le résultat du couplage d'une variante de l'algorithme par Essaim de Particules nommé PSO (Particle Swarm Optimisation), avec la méthode du recuit simulé nommée SA (Simulted Annealing). Cette méthode a les caractéristiques suivantes :

- Donne des résultats meilleurs que les méthodes classiques SA, PSO ;
- Applicable à des problèmes complexes d'optimisation dans les différents domaines, informatique, mécanique...etc ;
- Adaptable au calcul distribuée pour améliorer le couplage systémique.

Organisation du manuscrit : Le rapport est composé d'une introduction, de cinq chapitres, d'une conclusion et des perspectives et de trois annexes.

Ce rapport de thèse est scindé en cinq chapitres. Le premier dresse un état de l'art sur l'optimisation globale. Le deuxième chapitre est consacré à la méthode d'optimisation hybride MPSO-SA proposée ainsi que son implémentation avec différents langages de développement (MATLAB, JAVA) pour améliorer ses performances. Le troisième chapitre présente une application de notre méthode à des problèmes concrets en mécanique des structures. Le quatrième chapitre concerne son application à l'optimisation de l'interopérabilité des systèmes informatiques. Enfin, le cinquième chapitre, fait l'objet d'une distribution du calcul d'optimisation à des fins d'amélioration des performances, et de constitution d'une plateforme pour le calcul scientifique dans une architecture réseau dite pair à pair (Peer to Peer).

Le premier chapitre expose l'état de l'art de l'optimisation globale. Il commence par une classification des méthodes d'optimisation. Ensuite, il passe aux notions de base ainsi qu'aux définitions de quelques métaheuristiques. Ces méthodes ont pour but la recherche de bonnes solutions (optimum global) aux problèmes d'optimisation. Ces problèmes n'ont pas cessé d'être de plus en plus complexes en consommant un temps CPU très important. Ce qui pousse la recherche à exploiter de nouvelles méthodes et techniques d'optimisation afin de suivre la progression technologique actuelle.

Le deuxième chapitre est dédié à la présentation d'une nouvelle méthode d'optimisation globale que nous appelons MPSO-SA. Cette nouvelle méthode hybride que nous proposons est un couplage de l'algorithme d'optimisation par essaim de particules modifiées MPSO (Modified Particle Swarm Optimisation) avec la méthode du recuit simulé (SA). Le processus d'hybridation utilise MPSO comme mécanisme de recherche évolutif pour acquérir une exploration efficace des solutions prometteuses de tout le domaine de recherche. Au cours de ce processus, SA est utilisée pour effectuer une amélioration des solutions trouvées. Plusieurs combinaisons ont été testées pour retenir un modèle approprié.

Le troisième chapitre a comme objectif la résolution des problèmes de fiabilité des structures en mécanique par la méthode hybride d'optimisation proposée. Dans le domaine de fiabilité, il est primordial de prendre en compte les incertitudes pour un dimensionnement optimal et fiable des structures. Ce qui impose l'intégration des approches probabilistes pour une bonne estimation de la fiabilité de la structure, ainsi que l'impact de la variabilité des paramètres sur le comportement de la structure.

Le quatrième chapitre présente une application de la méthode hybride proposée à l'interopérabilité des systèmes d'information. La notion d'interopérabilité caractérise la propriété d'interaction et d'échange entre les systèmes de traitement de l'information. Elle a été considérée (McCall, Boem) comme fonctionnalité essentielle et nécessaire pour tout système informatique moderne et organisé. Pour faire évoluer la collaboration et la coopération organisationnelle entre les systèmes d'information, la méthode MPSO-SA proposée a été utilisée pour trouver la meilleure répartition (optimale) de l'effort nécessaire dans une

situation de travail en réseau.

Le cinquième chapitre traite la notion des réseaux Peer-to-Peer (P2P). Ces réseaux sont apparus pour résoudre des problèmes de congestions et de manque de la bande passante dans les réseaux actuels (coûts faramineux d'entretien). Le calcul scientifiques sous le mode distribué P2P est rarement utilisé alors que l'offre potentielle en capacité de traitement non exploité peut être énorme. Parmi les fonctionnalités utiles des réseaux Peer to Peer, la notion du calcul distribué sera traitée dans ce chapitre en utilisant le Framework de développement JXTA (développé par SUN). Cette architecture de calcul distribué va être testée pour une reconduite projetée sur l'algorithme d'optimisation MPSO-SA.

La conclusion du présent manuscrit portera sur une synthèse des principales contributions, et sur la mise en exergue de nouvelles perspectives sur la base des travaux effectués.

ÉTAT DE L'ART DES MÉTHODES D'OPTIMISATION

1

SOMMAIRE

INTRODUCTION	14
1.1 TECHNIQUES D'OPTIMISATION	15
1.1.1 Optimisation sans contraintes	15
1.1.2 Optimisation avec contraintes	15
1.1.3 Optimisation multiobjectif	18
1.2 CLASSIFICATION DES MÉTHODES D'OPTIMISATION	18
1.2.1 Algorithmes énumératifs	19
1.2.2 Algorithmes déterministes	19
1.2.3 Algorithmes stochastiques	20
1.2.4 Méthodes hybrides	22
1.2.5 Choix d'un algorithme d'optimisation	22
1.3 MÉTHODES MÉTAHEURISTIQUES	23
1.3.1 Algorithmes génétiques	23
1.3.1.1 Étapes de l'algorithme	23
1.3.1.2 Implémentation de l'algorithme	25
1.3.2 Méthode d'essaim de particules	26
1.3.2.1 Généralités	26
1.3.2.2 Étapes de l'algorithme :	27
1.3.3 Algorithme de colonie de fourmis	28
1.3.3.1 Introduction	28
1.3.3.2 Étapes de l'algorithme	28
1.3.4 Recuit simulé	29
1.3.5 Recherche tabou	30
CONCLUSION	31

INTRODUCTION

Ce chapitre présente les notions de base ainsi que les définitions de quelques métaheuristiques. Ces méthodes ont pour but la recherche de bonnes solutions aux problèmes d'optimisation pour lesquels la recherche directe de la meilleure solution peut nécessiter un temps de calcul excessif.

Les méthodes à caractère local, comme la méthode Tabou et le Recuit simulé, sont aussi développées dans ce chapitre. Nous nous contentons ici d'en rappeler simplement le principe.

La recherche Tabou est une méthode d'optimisation qui améliore les méthodes d'optimisation locales usuelles en gardant en mémoire un certain nombre, limité, des dernières meilleures solutions obtenues afin de permettre d'échapper à un éventuel optimum local.

L'algorithme du recuit simulé s'inspire d'une technique utilisée en métallurgie qui alterne des cycles de réchauffages et de refroidissements lents en vue d'améliorer la qualité du matériau concerné en minimisant son énergie.

Deux métaheuristiques sont présentées dans ce chapitre. Elles permettent la recherche de bonnes solutions à des problèmes d'optimisation dans le cas de variables continues ou discrètes. Leurs particularités se résument dans le fait qu'elles sont inspirées de la nature et qu'elles fournissent chacune une famille de solutions.

La méthode d'optimisation par essaim de particules est issue de l'observation du comportement d'animaux sociaux, où la coopération et l'échange d'informations entre individus permet, bien que chacun des individus ait des capacités limitées, de résoudre des problèmes d'optimisation complexes, tels que la recherche du plus court chemin vers une source de nourriture comme c'est le cas chez les abeilles.

Les algorithmes génétiques sont basés sur le principe de la reproduction et de la révolution : croisement, mutation et sélection qui sont les opérateurs usuels de la méthode. L'optimisation étant réalisée par la survie de l'individu le plus adapté. La difficulté dans cette approche est l'expression des solutions du problème sous la forme de chromosomes, chacun représentant totalement une solution envisageable.

1.1 TECHNIQUES D'OPTIMISATION

Un problème d'optimisation est énoncé sous la forme générale d'une fonction (mono-objectif) ou plusieurs fonctions objectifs (multi-objectif). L'optimisation s'effectue par rapport à des paramètres d'entrée X et suivant un ensemble de contraintes. En effet, minimiser $f(X)$ sous la contrainte $X \in S$ revient à trouver X^* tel que $f(X^*) \leq f(X)$, pour tout $X \in S$. f est appelée fonction-objectif (ou fonction coût ou critère ou fonction économique) et S : domaine réalisable (ou admissible).

1.1.1 Optimisation sans contraintes

Un problème d'optimisation est dit continu quand les variables sont réelles. La formulation suivante décrit un problème d'optimisation continue sans contraintes.

$$\begin{cases} \text{Minimise } f(X) \text{ de } \mathfrak{R}^n \rightarrow \mathfrak{R} \\ \text{Trouver } X^* \text{ minimise } f(X) \\ \text{avec } X_{min} \leq X \leq X_{max} \end{cases} \quad (1.1)$$

- $f(X)$ est la fonction objectif (ou critère ou coût).
- X est la variable de décision elle est de dimension n .
- Un problème de minimisation peut toujours se ramener à un problème de maximisation car le minimum de $f(X)$ est le maximum de $-f(X)$
- $[X_{min}, X_{max}]$ délimite le domaine de recherche ou espace de solutions.

1.1.2 Optimisation avec contraintes

Un problème d'optimisation avec contraintes est formulé comme suit :

$$\begin{cases} \text{Min } f(X) \\ \text{Sous les contraintes} \\ g_i(x) \leq 0 \quad i = 1, 2, \dots, q \\ h_i(x) = 0 \quad i = 1, 2, \dots, p \\ x_{min} \leq x \leq x_{max} \end{cases} \quad (1.2)$$

- $f(X)$ est la fonction objectif (ou critère ou coût).
- $g = (g_1, \dots, g_q)^T : \mathfrak{R}^n \rightarrow \mathfrak{R}^q$ est la fonction des contraintes d'inégalité.
- $h = (h_1, \dots, h_p)^T : \mathfrak{R}^n \rightarrow \mathfrak{R}^p$ est la fonction des contraintes d'égalité.

- Le problème a une dimension n (variable de décision), p contrainte d'égalité et q contraintes d'inégalité.
- $[X_{min}, X_{max}]$ délimite le domaine de recherche ou espace de solutions.

Méthodes de pénalité :

Une des manières de résolution des problèmes d'optimisation avec contrainte est les méthodes de transformation dite de pénalité. En effet, elles consistent à introduire les contraintes dans la fonction que l'on cherche à optimiser, ainsi le problème d'optimisation avec les contraintes de type inégalité dans 1.3

$$\left\{ \begin{array}{l} \min f(x) \\ g_i(x) \leq 0, \quad i = 1, \dots, n \end{array} \right. \quad (1.3)$$

Peut être transformé en un problème sans contraintes équivalent, formulé par :

$$\min \phi(x, r) = f(x) + r \sum_{i=1}^n P_i(g_i(x)) \quad (1.4)$$

où r est le coefficient de pénalité.

P_i est la fonction de pénalité.

La fonction P_i favorise la sélection des points admissibles par rapport aux points non admissibles. Les méthodes de pénalité sont divisées en deux classes selon la nature de la fonction de pénalité :

Les méthodes de pénalité intérieure : ou méthodes à barrière utilisent une fonction de pénalité qui assure le respect des contraintes tout le long du processus d'optimisation. En effet la fonction forme une sorte de barrière infinie le long de la frontière du domaine réalisable.

La fonction inverse :

$$P_i(g_i(x)) = \frac{-1}{g_i(x)} \quad (1.5)$$

$$P_i(g_i(x)) = -\log(-g_i(x)) \quad (1.6)$$

Les méthodes à barrière sont handicapés par le temps de calcul qui devient important. En effet ces méthodes impose que les paramètres soient situés dans l'espace réalisable et elles utilisent des algorithmes stochastiques par échantillonnage aléatoire.[Rao (1996)].

Les méthodes de pénalités extérieures : à la différence des méthodes de pénalité intérieures sont valables aussi bien à l'intérieur qu'à l'extérieur de l'espace réalisable. La pénalisation des solutions augmente proportionnellement avec l'éloignement de l'espace réalisable. De cette manière elles évitent les inconvénients des méthodes de pénalité intérieures.

Exemple des fonctions de pénalité extérieures [Rao (1996)] :

$$P_i(g_i(x)) = \max[0, g_i(x)] \quad (1.7)$$

$$P_i(g_i(x)) = \max[0, g_i(x)]^2 \quad (1.8)$$

Les contraintes sont satisfaites pour une valeur de r assez grande, mais cela provoque un mauvais conditionnement de $\phi(x, r)$, ainsi que des problèmes numériques durant le processus d'optimisation. Pour éviter ce problème une succession de valeurs de r sont données d'une manière itérative dans le processus de résolution. Ainsi le problème d'optimisation sans contrainte suivant 1.9 est résolu :

$$\min \phi(x, r, k) = f(x) + r^k \sum_{j=1}^n P_j(g_j(x)) \quad (1.9)$$

L'avantage de cette méthode réside dans le faite que le point de départ n'est pas nécessairement admissible contrairement au point final qui doit être dans le domaine admissible.

L'inconvénient de la fonction de pénalité extérieure est qu'elle converge vers à un optimum réalisable seulement quand k tend vers l'infini. Elle approche la solution par une série de valeurs non admissibles. [Luenberger (1973)].

1.1.3 Optimisation multiobjectif

L'optimisation multi-objectif cherche une solution pour un vecteur de fonctions coûts. Cette solution n'est pas unique, comme dans le cas de l'optimisation mono-objectif, mais constituée par un ensemble de solution. d'où la notion de **l'ensemble pareto optimal**. Le choix d'une solution parmi l'ensemble des solutions dépend du problème traité. Le décideur est donc obligé de choisir une solution de bon compromis.

Un problème multi-objectif peut être formulé comme suit :

$$\left\{ \begin{array}{l} \min F(x) = (f_1(x), f_2(x), \dots, f_k(x)) \\ \text{Sous les contraintes :} \\ g_i(x) \leq 0 \quad i = 1, 2, \dots, q \\ h_i(x) = 0 \quad i = 1, 2, \dots, p \\ \text{Avec } X_{min} \leq X \leq X_{max} \end{array} \right. \quad (1.10)$$

Avec

- $F(x) = (f_1(x), f_2(x), \dots, f_k(x))$ est le vecteur des fonctions objectif de dimension k .
- $x = (x_1, x_2, \dots, x_n)$ est vecteur des variables de conception dans l'espace de solutions de dimension n .
- $g(x)$ de \mathcal{R}^n dans \mathcal{R}^q est la fonction des contraintes d'inégalité.
- $h(x)$ de \mathcal{R}^n dans \mathcal{R}^p est la fonction des contraintes d'égalité.

1.2 CLASSIFICATION DES MÉTHODES D'OPTIMISATION

Les méthodes d'optimisation n'ont cessé d'évoluer ces dernières années. La majorité de ces méthodes classiques sont convergentes et donnent de bons résultats, seulement, lorsque la dimension du problème augmente ou si l'on recherche une très bonne précision, l'algorithme risque de converger lentement. Par ailleurs, la plupart de ces méthodes classiques sont très coûteuses en termes de temps de calcul ou de régularité, ce qui rend leur application quasi-impossible vu la structure mal conditionnée des problèmes traités actuellement. La dernière contrainte qu'on peut noter, c'est l'existence de problèmes dits « NP-Complets » c'est-à-dire trop complexes ou difficiles pour être résolus par des méthodes déterministes.

C'est le cas du Problème du Voyageur de Commerce PVC où le temps de calcul croît exponentiellement avec la complexité du problème, ce qui impose l'utilisation des approches heuristiques, probabilistes ou stochastiques.

Pour un domaine continu, on distingue souvent deux types d'optimisation :

- **L'optimisation locale** recherche d'une solution qui est la meilleure localement (dans son voisinage). Cette solution est appelée optimum local.
- **L'optimisation globale** recherche quant à elle la meilleure solution du domaine en entier. Par conséquent, il n'existe aucune solution dans tout le domaine qui lui soit meilleure tout en respectant les contraintes. Cette solution est appelée optimum global.

Il existe plusieurs classes de méthodes développées. Ces méthodes sont réparties selon plusieurs critères :

- Le problème est-il mono objectif ou multi objectif ?
- L'optimisation est-elle discrète ou continues ou mixte ?
- L'optimisation est-elle avec ou sans contraintes ?
- L'approche de résolution est-elle déterministe, énumérative ou stochastique ?
- etc.

1.2.1 Algorithmes énumératifs

Dans un espace de recherche fini, ou infini mais discrétisé, un algorithme énumératif évalue la valeur de la fonction à optimiser en chaque point de l'espace des solutions. L'utilisation d'un tel algorithme est intéressante lorsque le nombre de points n'est pas très important. Mais en pratique beaucoup d'espaces de recherche sont trop vastes pour que l'on puisse explorer toutes les solutions une par une et tirer une information utilisable. Ces méthodes présentent deux inconvénients majeurs [Chekroun (2011)] :

- a) Elles sont inadaptées aux problèmes de grande dimension ;
- b) Elles ne sont pas guidées par un raisonnement ou un processus intelligent, qui conduit la recherche vers des sous-espaces, susceptibles de contenir une bonne solution, sans balayer tout l'espace des solutions [Gherbi et al. (2006)].

1.2.2 Algorithmes déterministes

Il existe de nombreuses méthodes déterministes ou exactes qui permettent de résoudre certains types de problèmes d'optimisation en un temps fini. Cependant, ces méthodes néces-

sitent que la fonction objectif présente un certain nombre de caractéristiques, telles que la convexité, la continuité ou encore la dérivabilité. Parmi les méthodes les plus connues, on peut citer les méthodes de programmation linéaire [Schrijver (1998)], quadratique ou dynamique [Bertsekas (2000)], la méthode de Newton [Nocedal et Wright (1999)], la méthode du simplex [Nelder (1965)] ou encore la méthode du gradient.

Ces algorithmes n'utilisent aucun concept stochastique : c'est-à-dire que pour résoudre un problème, l'algorithme se comportera toujours de la même façon et donnera toujours la même réponse. Les méthodes déterministes se divisent en deux classes principales : les méthodes d'exploration indirecte et les méthodes d'exploration directe. Les méthodes indirectes cherchent à atteindre les optimums locaux en résolvant les systèmes d'équations, souvent non linéaires, obtenus en annulant le vecteur gradient de la fonction étudiée. La recherche d'un optimum hypothétique commence par la restriction de l'espace de recherche aux points de pente nulle dans toutes les directions. Les méthodes d'exploration directes recherchent les optimums locaux en se déplaçant dans une direction qui dépend du gradient de la fonction. Pour trouver un minimum local, on peut emprunter la direction de la plus forte pente [Datta et S.W. (1999)]. Certains problèmes restent cependant trop complexes à résoudre pour les méthodes déterministes. Certaines caractéristiques peuvent être problématiques pour ces méthodes. Parmi celles-ci, on peut citer la discontinuité, la non-dérivabilité ou encore la présence de bruit. Dans ce cas, le problème d'optimisation est dit difficile, car aucune méthode déterministe ne peut résoudre ce genre de problème en un temps raisonnable [COOREN (2008)].

L'optimisation par descente du gradient

L'algorithme d'optimisation déterministe le plus utilisé est de gradient, dont le principe est de partir d'un point aléatoire puis de se déplacer dans la direction de la plus forte pente. En appliquant un certain nombre d'itérations, l'algorithme converge vers une solution qui est un minimum local de la fonction objectif. La vitesse de convergence de l'algorithme est déterminée empiriquement selon l'adéquation de l'inclinaison de la pente. Il existe plusieurs variantes pour la descente de gradient : à pas fixe, à pas variables, à pas optimal, total, conjugué. Il existe aussi une variante stochastique de cette méthode de descente [Chekroun (2011)].

1.2.3 Algorithmes stochastiques

L'échantillonnage aléatoire (ou méthode de Monte-Carlo) est l'une des méthodes stochastiques la plus simple. Elles consistent à évaluer des points engendrés de façon aléatoire et à

conserver le meilleur. Son avantage est d'être simple et très facile à implémenter, mais elle présente un sérieux inconvénient : elle n'est pas efficace, et exige souvent un nombre élevé d'itérations pour donner une solution acceptable. Les méthodes stochastiques sont basées sur une approche en partie ou entièrement guidée par un processus stochastique. Contrairement aux méthodes déterministes, leur convergence n'est pas garantie, ou, dans le meilleur des cas, elle est garantie de manière asymptotique. En termes d'efficacité, les algorithmes d'optimisation stochastiques présentent les avantages suivants [Chekroun (2011)] :

- La nature globale de la solution trouvée ce qui assure sa validité ;
- La rapidité d'obtention d'un code de calcul basé sur ces méthodes ;
- Les procédés mathématiques sur lesquels reposent ces algorithmes ne nécessitent pas de point de départ physiquement cohérent (Ils évitent naturellement les optimums locaux) ;
- Ces méthodes sont directes, c'est-à-dire sans calcul supplémentaire ni hypothèse particulière.

Cependant, les inconvénients qui limitent l'efficacité des algorithmes d'optimisation stochastiques, concernent :

- Leur rapidité d'obtention d'une solution. En effet, ces méthodes convergent lentement et nécessitent un grand nombre d'itérations pour converger asymptotiquement vers un optimum global.
- Il est très difficile de savoir avec quelle précision relative l'optimum global est atteint.

Parmi les différentes méthodes stochastiques d'optimisation globale, nous allons nous intéresser aux méthodes heuristiques (ou méthodes approchées). Contrairement aux méthodes dites exactes, ces heuristiques ne procurent pas forcément une solution optimale, mais seulement une bonne solution en fonction de la complexité du problème et de la rapidité des processeurs de calcul. La plupart d'entre elles sont conçues spécifiquement pour un type de problème donné. Ces méthodes sont caractérisées comme suit :

- Elles sont pour partie stochastiques : cette approche permet de faire face à l'explosion combinatoire des possibilités ; d'origine combinatoire, elles ont l'avantage, décisif dans le cas continu, d'être directes, c'est-à-dire qu'elles ne recourent pas au calcul, souvent problématique, des gradients de la fonction objectif.
- Elles sont inspirées par des analogies : avec la physique (recuit simulé, diffusion simulée), avec la biologie (algorithmes génétiques, recherche tabou) ou avec l'éthologie (colonies de fourmis, essaim de particules)
- Elles partagent aussi les mêmes inconvénients : les difficultés de réglage des paramètres de la méthode, et le temps de calcul élevé.

Algorithmes Evolutionnaires (AE)

Les AE sont des algorithmes stochastiques inspirés de l'évolution Darwinienne des populations. Selon Darwin les individus les plus aptes survivent à la sélection naturelle et se reproduisent. Ce processus se répète d'une génération à l'autre menant selon l'adaptation de la population des individus à l'environnement au cours de l'évolution.

En termes d'optimisation l'évolution se traduit par un processus itératif de recherche de l'optimum dans l'espace de décision. Le critère pour définir les éléments les *plus aptes* d'un sous-ensemble fini de cet espace correspond, naturellement à l'objectif d'optimisation. L'adaptation à l'environnement est réalisée par le fait de trouver à chaque nouvelle itération des solutions potentiellement meilleures, la performance des solutions étant évaluée à la base des valeurs correspondantes de la fonction objectif. L'avantage crucial de ces méthodes devant les algorithmes d'optimisation traditionnels consiste en ce que les AE "se contentent" de connaître les valeurs de la fonction objectif non seulement sans faire appel à la dérivée de cette fonction mais même sans exiger son expression analytique.

1.2.4 Méthodes hybrides

Le concept d'hybridation consiste à combiner deux méthodes d'optimisation afin de combler certaines lacunes de ces méthodes et de combiner leurs avantages. L'hybridation s'applique pour un grand nombre d'algorithmes.

Dans la présente thèse, nous proposons une nouvelle méthode (Voir chapitre 2) résultante d'une hybridation de deux méthodes. Le but est d'apporter une réduction du nombre d'évaluation de la fonction objectif par rapport au nombre normalement nécessaire en n'utilisant que les algorithmes séparément.

1.2.5 Choix d'un algorithme d'optimisation

La classification du type de problème et le choix de la méthode d'optimisation est souvent une étape essentielle avant toute résolution d'un problème d'optimisation globale. Pour bien cerner le problème il faut voir les caractéristiques de la fonction objectif f , le domaine de recherche, types de contraintes, temps de calcul nécessaire pour résoudre le problème, le coût d'évaluation de la fonction objectif, ...etc.

La classification des problèmes et des méthodes donne une bonne orientation pour la résolution du problème, malheureusement, aucune méthode d'optimisation n'est capable de traiter efficacement tous les cas. Le choix d'un algorithme ne doit pas se baser uniquement sur des cas de tests généraux, mais doit aussi étudier des problèmes représentatifs des applications envisagées. Il ne faut pas négliger l'expérience qui guide l'utilisateur dans le choix

de l'algorithme à implanter.

Pour choisir la méthode la plus adaptée à un problème bien précis, les caractéristiques principales prises en compte sont : La capacité à éviter les minima locaux, La capacité à traiter des problèmes avec ou sans contraintes, mono ou multi objectifs et la rapidité de convergence.

1.3 MÉTHODES MÉTAHEURISTIQUES

1.3.1 Algorithmes génétiques

Avant d'aborder le contenu de l'algorithme, il est nécessaire de parler des espaces de recherches c'est-à-dire les espaces où s'effectue la sélection [Fraser. (1957)].

L'espace de recherche est ici $W = 0, 1$ la justification de l'utilisation intensive de cet espace de recherche particulier est fondée sur une analogie avec la biologie (une chaîne de bits étant assimilée à un chromosome).

Il est préférable de définir certains termes qui seront souvent cités comme un vocabulaire général.

Population : un vecteur de variables qui subira les différents opérateurs d'évolution pour passer d'une génération à l'autre.

Individu : c'est tout simplement la variable qu'on traite.

Fonction fitness ou d'évaluation ou d'adaptation : c'est le critère qui sert à déterminer les bons individus des mauvais.

Un vocabulaire spécial aux algorithmes génétiques est résumé comme suit :

Chromosome : individu sous forme de chaîne de bits.

Gène : un bit du chromosome.

1.3.1.1 Étapes de l'algorithme

Afin de permettre la résolution des problèmes d'optimisation par les algorithmes génétique, les étapes décrites ci-dessous sont primordiales. De manière générale, l'algorithme commence par une population de base qui se compose le plus souvent de chaînes de caractères correspondant chacune à un chromosome.

Initialisation : c'est le choix de la population initiale à partir de l'espace de recherche qui se fait généralement de manière aléatoire.

Évaluation : à cette étape, on applique le critère d'évaluation. Dans le cas le plus fréquent d'optimisation de fonctions par algorithmes génétiques, la fitness n'est que la fonction elle-même.

Sélection : elle consiste à choisir les individus les plus performants. Les critères de sélection diffèrent selon le type d'AE. Ci-dessous la citation de deux critères :

L'ordonnement : (ranking) c'est le principe de sélection le plus simple, il consiste à attribuer à chaque individu son classement par ordre d'adaptation. Le meilleur sera dans le premier classement, et ainsi de suite jusqu'à l'obtention d'une population ordonnée.

La roue de la fortune :(wheel selection) le principe de la roue de la fortune consiste à associer à chaque individu q_i une probabilité P_i proportionnelle à sa fitness $f(q_i)$ dans la population. Cette probabilité pourra être calculée comme suit :

$$P_i = \frac{S(f(q_i))}{\sum S(f(q_i))}$$

Où S est une fonction régulière et croissante (généralement on prend S=Identité).

Croisement : le croisement a pour but d'enrichir la diversité de la population en manipulant les composantes des individus. Classiquement, le croisement est envisagé avec deux parents et génère deux enfants. Dans les algorithmes génétiques, cela consiste en un échange d'un certain nombre de gènes entre deux chromosomes parents choisis aléatoirement pour donner deux descendants. En pratique, on définit une probabilité de croisement P_c qui varie en général entre 70% et 95%. On distingue trois types de croisement :

1. Croisement à un point
2. Croisement à deux points
3. Croisement uniforme

Mutation : l'opérateur de mutation apporte l'aléa nécessaire à une exploration efficace de l'espace. Ça consiste à choisir aléatoirement un chromosome avec une probabilité P_m dite (probabilité de mutation) généralement comprise entre 0.5 et 1, puis à choisir un gène aléatoirement dont on va changer la parité.

Évaluation : à cette étape, on calcule la fitness des enfants.

Remplacement : remplacement des parents au moyen du critère de sélection darwinienne appliqué aux enfants, avec participation éventuelle des parents.

Critère d'arrêt : l'évolution s'arrête quand le niveau de performance souhaité est atteint, ou qu'un nombre fixé de générations s'est écoulé sans améliorer l'individu le plus performant.

1.3.1.2 Implémentation de l'algorithme

Etapas de l'algorithme

Les étapes essentielles de l'algorithme sont les suivantes :

1. Population de base générée aléatoirement

- n chaînes de caractères ou de bits ;
- 1 chaîne correspond à 1 chromosome.

2. Evaluation

- A chaque chaîne, une note de correspondance à son adaptation au problème est effectuée.

3. Sélection

- Tirage au sort de $n/2$ couples de chaînes sur une roue biaisée ;
- Chaque chaîne a une probabilité d'être tirée proportionnelle à son adaptation au problème ;
- Optimisation possible : si l'individu le plus adapté n'a pas été sélectionné, il est copié d'office dans la génération intermédiaire à la place d'un individu choisi aléatoirement.

4. Croisement et Mutation :

- Chaque couple donne 2 chaînes filles ;
- **Enjambement** : probabilité : emplacement de l'enjambement choisi aléatoirement ;
- **Mutation des chaînes filles** : probabilité : de 0,1 à 1 ;
- Inversion d'un bit au hasard (remplacement au hasard d'un caractère par un autre) ;
- Probabilité fixe ou évolutive (auto-adaptation) ;
- On peut prendre probabilité = $1/\text{nombre de bits}$.

L'algorithme est schématisé comme suit :

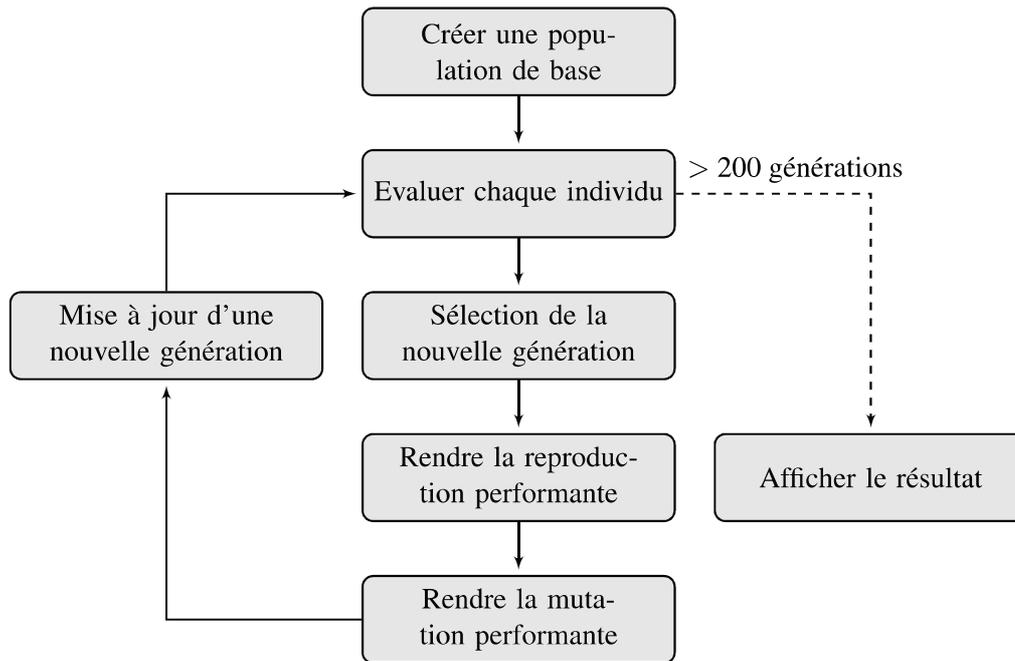


FIGURE 1.1 – Schéma de l'algorithme.

1.3.2 Méthode d'essaim de particules

1.3.2.1 Généralités

L'optimisation par essaim de particules ou Particle Swarm Optimization (PSO) a été développée originalement par James Kennedy et Russell Eberhart (Kennedy et Eberhart, 1995). L'algorithme se base sur un modèle simplifié des interactions sociales entre des «agents», collaboration des individus entre eux. Ces individus peuvent être par exemple des oiseaux ou des abeilles nommés particules, ils utilisent leurs expériences individuelles ainsi que l'expérience de toute la population afin de se déplacer dans un espace donné (domaine de recherche) pour trouver la nourriture. Grâce à la notion de collaboration, une particule qui trouve une solution prometteuse peut attirer tout le reste de la population pour lui faire profiter de sa découverte.

L'algorithme PSO commence par une initialisation de la population. Les particules et les vitesses sont distribués d'une manière aléatoire. A chaque particule correspond un vecteur de vitesse, qui est calculé suivant l'expérience de chaque particule et le mouvement du groupe. Les positions des particules sont mises à jour par la notion de vitesse dans chaque itération. Ce processus est répété jusqu'à convergence.

1.3.2.2 Étapes de l'algorithme :

les différentes étapes de l'algorithme sont présentées ci dessous :

Initialisation

- . $X_i \leftarrow$ Générer des particules initiales de l'essaim
- . $V_i \leftarrow$ Générer la vitesse initiale des particules
- . $X_{besti} \leftarrow X_i$ Déterminer les meilleures positions de la particule
- . $X_{gbest} \leftarrow X_i$ Déterminer les meilleures positions de l'essaim
- . $X_{Nbest} \leftarrow X_i$ Déterminer les meilleures positions du voisin

Répéter

- .*Boucle* $i = 1 : N$ (toutes les particules dans l'essaim)
 - . $Fitness_i(t) \leftarrow EvaluateFitness(X_i)$
 - . *Si* $Fitness_i(t) < Fitness_{(X_{besti})}(t)$
 - . $X_{besti} \leftarrow X_i$ attracteur de particules
 - . *Fin Si*
 - . *Boucle* $i = 1 : M$ (M nombre de voisins dans l'essaim)
 - . $X_{Nbest} \leftarrow$ définir la meilleure position du voisinage.
 - . *Fin Boucle*
 - . *Si* $Fitness_i(t) < Fitness_{(X_{gbest})}(t)$
 - . $X_{gesti} \leftarrow X_i$ attracteur essaim
 - . *Fin Si*
 - . *Update velocity* $V_{id}(t+1)$ dans l'équation de la vitesse
 - . *Update position* $X_{id}(t+1)$
 - . *Si* $X_{id}(t+1) < X_{min}$ ou $X_{id}(t+1) > X_{max}$
 - . $X_{id}(t+1) \leftarrow X_{random}$ $X_{random} \subseteq [X_{min}X_{max}]$
 - . *Fin Si*
 - .*Fin Boucle*

Jusqu'à Critère d'arrêt

La position et la vitesse d'une particule sont définies respectivement par, $X_i = (x_{i,1}, \dots, x_{i,n})$ et $V_i = (v_{i,1}, \dots, v_{i,n})$. L'espace de recherche est de n dimensions. Chaque particule a un meilleure position historique X_{besti} , cette valeur peut changer à chaque itération. La meilleure position de tout l'essaim est représentée par le vecteur X_{gbest} . La notion de voisinage est utilisée dans cette approche, elle consiste à diviser l'essaim en plusieurs sous essaims pour plus de diversité dans la recherche et une meilleurs exploitation des solutions. Chaque sous essaim est présenté par son meilleur élément nommé par X_{Nbest} .

1.3.3 Algorithme de colonie de fourmis

1.3.3.1 Introduction

L'algorithme de colonie de fourmis connu sous l'abréviation ACO (Ant Colony Optimization) a connu ces début en 1992 par [Dorigo (1992)]. Le principe est de simuler le comportement naturel des fourmis. Ces dernières sont capable de trouver collectivement le parcours le plus court entre une source d'aliment et leur nid.

Cet algorithme a été conçu initialement pour résoudre le Problème du Voyageur de Commerce, puis il a connu un développement important en couvrant plusieurs domaines d'optimisation. Le principe de ce problème est de trouver la plus courte tournée qui passent par n villes et retourne à la ville de départ, il faut noter que chaque ville est visitée une seule fois (Problème du voyageur de commerce PVC).

Une fourmi se déplace dans son domaine de recherche autour de sa colonie. Lorsqu'elle trouve de la nourriture dans une position donnée elle rentre directement au nid en laissant une trace nommée par phéromone (substances chimique). Cette trace est un moyen de communication avec les autres fourmis qui travaillent en parfaite collaboration. La notion du plus court chemin se construit par le fait que plus les fourmis passent par un chemin, plus la densité du phéromone laissée sera grande sur ce chemin et bien entendu il sera le plus attractif. En même temps, la quantité de phéromones déposée sur le plus long chemin diminue ce qui engendre sa disparition. C'est de cette façon que toutes les fourmis arrivent à construire le chemin le plus court.

1.3.3.2 Étapes de l'algorithme

Les différentes étapes de l'algorithme sont :

- ✓ Initialisation : Conception d'une population d'agents indépendants.
- ✓ Construction de la solution ;
- ✓ Pour chaque fourmi et d'une manière itérative :
 - Construction basée sur un choix probabiliste des composants de la solution en utilisant les pistes de phéromone ;
 - Mise à jour des piste de phéromone
- ✓ Atteinte du critère d'arrêt

1.3.4 Recuit simulé

Le recuit simulé est un algorithme d'optimisation qui simule un mécanisme naturel en métallurgie, c'est le chauffage et le refroidissement d'un solide suivant certaines conditions. Au début de l'algorithme, la température est grande, ce qui engendre une grande couverture du domaine de recherche. Au fur et à mesure que la température baisse les états à haute énergie deviennent peu probables par rapport à ceux de faible énergie (état plus stable). Cette simulation numérique de l'opération de recuit thermique est appliquée à l'optimisation combinatoire.

Étapes de l'algorithme de recuit simulé .

. $S \leftarrow S_{init}$ Générer la population aléatoire initiale.

. $S_{best} \leftarrow S$ Initialiser la meilleure solution S_{best} .

. $F_{best} \leftarrow F(S)$ Initialiser la valeur F_{best} .

.Initialisation de la température T .

.début Itérations $NbIter \leftarrow 0$

.*Tantque* $T > T_{minimum}$

. $NbIter \leftarrow NbIter + 1$

. Générer S' dans le voisinage de S

. $\Delta \leftarrow F(S') - F(S)$

Calcul de la différence entre la position actuelle et celle générée dans le voisinage.

. *Si* $\Delta < 0$ test d'amélioration

. $S \leftarrow S'$ Affectation

. *Si* $F(S') < F(S_{best})$

. $F_{best} \leftarrow F(S')$ Mémorisation des meilleures valeurs

. $S_{best} \leftarrow S'$ Mémorisation des meilleures positions

. *FinSi*

. *SiNon*

. *Random* p in $[0 1]$

. *Si* $p \leq \exp(-\Delta/T)$ Test de probabilité.

. $S \leftarrow S'$ Échapper des minimums locaux.

. *FinSi*

. *FinSi*

.Fin

La phase d'initialisation est la première étape de l'algorithme, où la température est fixée

à une valeur élevée, les différents paramètres de la méthode sont fixés eux aussi par des valeurs qui peuvent être changées suivant le type de problème. A chaque itération (nouveau pas) la fonction objectif est comparée avec sa valeur précédente. La nouvelle valeur est acceptée s'elle est meilleur, autrement elle peut être acceptée avec une certaine probabilité dite de Boltzmann $P = \exp(-\Delta E/k_B T)$ où $-\Delta E$ est la variation de la l'énergie (fonction objectif par similitude) et T le paramètre de la température.

La température est diminuée par palier avec un coefficient de refroidissement α suivant l'équation $T_k = \alpha T_{k-1}$. Ce processus continue jusqu'à l'atteinte de la température minimale et bien entendu la solution optimale approchée.

1.3.5 Recherche tabou

La recherche tabou est originellement créée par Glover [Glover (1986)]. Elle est principalement basée sur une simulation de la mémoire humaine, afin de faire une recherche intelligente dans l'espace des solutions. Cette méthode à la particularité par rapport aux autres métaheuristiques qu'elle peut tirer des leçons du passé et ne pas prendre des chemins inutiles.

Le processus de recherche de la méthode tabou se déroule en utilisant une seule configuration courante. Elle débute par une solution aléatoire, qui progresse au fur et à mesure de l'avancement du programme d'optimisation. La méthode peut se résumer aux points suivants :

- Pour éviter de revenir sur une configuration déjà faite auparavant, une liste contient certains des mouvements précédents qui sont interdits. Cette liste est appelée **la liste tabou** ;
- L'algorithme développe de cette façon une sorte de forme primaire d'une mémoire à court terme ;
- Cet algorithme tel qu'il est définie, n'est pas toujours très performant pour des types de problème complexe d'optimisation. Beaucoup de recherches sont faite pour intensifier et diversifier les processus de recherche [GLOVER (1996)].

La recherche tabou est itérative elle permet d'évoluer d'une solution à une autre dans son voisinage jusqu'à ce qu'un critère d'arrêt choisi soit satisfait. En pratique, elle consiste, partant d'une solution donnée, à rechercher la meilleure solution située dans le voisinage immédiat avec interdiction d'aller sur l'une des N dernières meilleures solutions obtenues, ce qui permet d'éviter le piège du minimum.

CONCLUSION

Ce chapitre porte sur l'état de l'art de l'optimisation. Il a présenté les méthodes tabou, recuit simulé, essaim de particules et les algorithmes génétiques ou à stratégie d'évolution.

Les méthodes stochastiques contribuent à la résolution de problèmes d'optimisation. En effet, le nombre de problèmes qui font appel à l'optimisation sont nombreux, à savoir les problèmes en productique mécanique (voir chapitre 3), les problèmes d'aide à la décision informatique (voir chapitre 4), ainsi que d'autres problèmes réels de la vie courante.

Après la présentation des différentes techniques d'optimisation ainsi que leur classification, le chapitre suivant décrit la méthode hybride proposée qui est le résultat du couplage d'une variante de l'algorithme Essaim de Particules dénommé PSO (Particle Swarm Optimisation) avec la méthode du recuit simulé dénommée SA (Simulted Annealing).

2 HYBRIDATION PROPOSÉE DE L'ESSAIM DE PARTICULES MODIFIÉ AVEC LE RECUIT SIMULÉ

SOMMAIRE

INTRODUCTION	33
2.1 ALGORITHME RECUIT SIMULÉ (SA)	34
2.1.1 Contexte général	34
2.1.2 Étapes de l'algorithme SA	35
2.2 ALGORITHME D'ESSAIM DE PARTICULES (PSO)	36
2.2.1 Principe et généralités sur la méthode PSO	36
2.2.2 Étapes de l'algorithme PSO classique	39
2.3 ESSAIM DE PARTICULES MODIFIÉ MPSO	40
2.3.1 Introduction	40
2.3.2 Étapes de l'algorithme	41
2.4 ALGORITHME PROPOSÉ : ESSAIM DE PARTICULES COUPLÉ AVEC LE RECUIT SIMULÉ	43
2.5 RÉSULTATS ET PERFORMANCE DE LA MÉTHODE MPSO-SA PROPOSÉE	45
2.6 COMPARAISON DES RÉSULTATS	47
2.6.1 Comparaison des taux de succès	47
2.6.2 Comparaison du temps de calcul	48
2.6.3 Comparaison du nombre d'évaluations de la fonction objectif	48
2.7 MODÉLISATION ET IMPLÉMENTATION SOUS JAVA	49
2.8 AMÉLIORATION DU TEMPS D'EXÉCUTION VIA LES THREADS	51
2.9 INTERFACE HOMME-MACHINE	53
CONCLUSION	54

INTRODUCTION

Ce chapitre est dédié à la présentation d'une nouvelle méthode d'optimisation globale dénommée MPSO-SA. La nouvelle méthode hybride proposée est le résultat du couplage d'une variante de l'algorithme par essaim de particules nommé PSO (Particle Swarm Optimisation) avec la méthode du recuit simulé nommé Simulated Annealing (SA). Cette méthode a été proposée après de nombreuses recherches dans le domaine [Savsani et al. (2010)], [Ali et Gabere (2010)], [Sitarz (2009)]. L'origine de la méthode de PSO vient des observations faites lors des simulations informatiques de vols groupés d'oiseaux et de bancs de poissons. Elle est connue comme une approche pouvant être efficace pour résoudre les problèmes d'optimisation dans de nombreux domaines de la recherche scientifique.

PSO est une méthode d'optimisation stochastique, développée par [Eberhart et Kennedy (1995)]. Elle présente quelques particularités intéressantes, avec entre autres la notion stipulant que l'efficacité est due à la collaboration plutôt qu'à la compétition.

Dans PSO, le mouvement des particules est modélisé avec la notion de vitesse. En effet, chaque particule subit un déplacement par la notion de la vitesse donnée durant tout le processus d'optimisation. L'optimisation par essaim de particules a été modifiée en améliorant l'équation de la vitesse par l'ajout d'un nouveau terme [Bochenek et Fory's (2006)]. Ces informations complémentaires fournies influencent le comportement de l'ensemble de l'essaim, et dans de nombreux cas cela améliore les performances de l'essaim.

Le recuit simulé est un processus physique qui consiste à ordonner les atomes d'un cristal afin de former une structure cristalline parfaite. Le recuit simulé exploite cette idée en implémentant dans une méthode d'optimisation l'idée de température [Lamberti (2008)]. Il commence avec une configuration de départ dont le point initial est choisi au hasard et se déplace de manière itérative vers une autre solution dans son voisinage pour une éventuelle amélioration de la solution en cours. Une brève description de cet algorithme est donnée dans cette section afin de montrer la combinaison avec PSO dans les sections suivantes.

L'algorithme modifié d'optimisation par essaim de particules (MPSO) couplé avec l'algorithme de recuit simulé (SA) donne lieu à un nouveau algorithme dénommée (MPSO-SA). Le processus d'hybridation utilise MPSO comme un mécanisme de recherche évolutive pour acquérir une exploration efficace des solutions prometteuses de tout le domaine de recherche, au cours de ce processus SA est utilisé pour effectuer une amélioration des solutions trouvées. Plusieurs combinaisons ont été testées pour choisir un modèle approprié.

2.1 ALGORITHME RECUIT SIMULÉ (SA)

2.1.1 Contexte général

SA est une technique d'optimisation classique qui a été utilisée avec succès pour résoudre un large éventail de problèmes d'optimisation. L'algorithme du recuit simulé (SA) est basé sur la physique du "recuit d'un solide vers une énergie minimum". La solution de départ est générée aléatoirement puis elle est améliorée en plusieurs itérations en faisant des pas successifs dans son voisinage. Dans un palier de température, un test de probabilité est effectué pour accepter d'éventuelles "mauvaises" solutions. A la fin des itérations, l'algorithme converge presque sûrement vers une solution optimale. Échapper des minimum locaux et atteindre le minimum global voir la courbe de la figure 2.1.

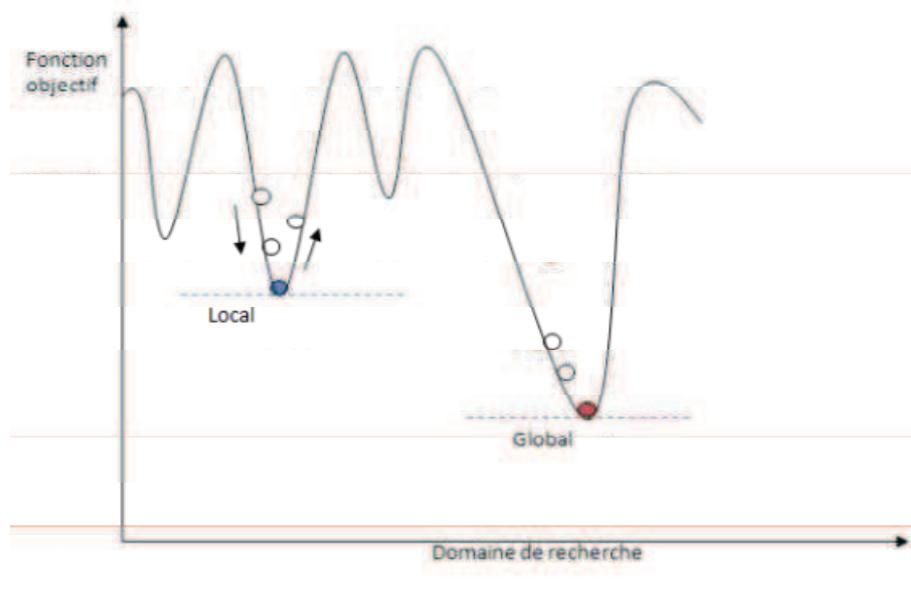


FIGURE 2.1 – *Optimum local et global (recuit simulé).*

Au début de l'algorithme, la température est élevée. Au fur et à mesure du déroulement du processus, on baisse la température par paliers. Quand celle-ci s'approche de 0, la méthode se comporte à la manière d'une descente de gradient. Cette méthode a été mise au point simultanément pour optimiser le placement de composants électroniques et pour optimiser une fonction continue [Kirkpatrick et al. (1983)].

Pour mieux expliquer le recuit simulé, nous prenons l'exemple d'une bille qui oscille sur une pente. Si on laisse tomber la bille elle va osciller avant de se stabiliser au fond du bol,

voir la figure 2.2.



FIGURE 2.2 – Oscillation de la bille (recuit simulé).

Si la bille descend une pente avec une remontée elle peut s’y arrêter et tomber dans un minimum local. Cependant, en lançant la bille de plus haut, elle risque d’éviter le minimum local. Mais fournir trop d’énergie n’est pas une bonne solution car la bille risque de quitter le minimum global pour aller se nicher dans un autre minimum local.

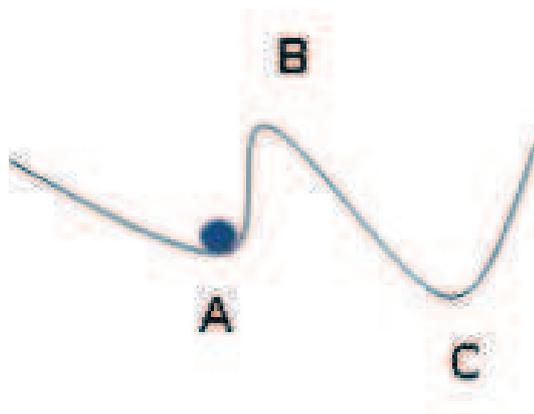


FIGURE 2.3 – Oscillation de la bille (recuit simulé).

La bonne stratégie consiste alors à :

1. Laisser la bille descendre suivant la plus grande pente.
2. La laisser remonter le côté opposé, mais en réduisant progressivement la hauteur quelle peut atteindre. Ainsi, arrivée dans le creux A elle pourra remonter en B et tomber en C (voir 2.3). Cependant, si elle arrive en C elle ne peut plus remonter en B car la hauteur BC est devenue trop importante.

2.1.2 Étapes de l’algorithme SA

Le principe du recuit simulé est d’effectuer une amélioration itérative. Il consiste à explorer partiellement l’ensemble des configurations possibles. Les remontés deviennent de plus en plus faibles durant le processus. En fait, des valeurs moins bonnes de la fonction sont

acceptées avec une probabilité non nulle. En permettant ces remontés occasionnelles dans le processus de recherche, on pourra éviter les pièges des minimums locaux et espérer l'atteinte de l'optimum global.

Dans la première étape, l'algorithme SA commence avec une solution initiale S qui est générée aléatoirement, par la suite, nous cherchons une nouvelle solution S' dans le voisinage de la précédente. Si la nouvelle valeur est meilleure, la transition est acceptée (mise à jour de la solution S par S'). Dans le cas contraire, on pourra accepter la transition avec une certaine probabilité non nulle.

$$\text{la probabilit } P < \exp\left(\frac{\Delta E}{T}\right)$$

où ΔE est la variation de l'énergie, qui est la variation de la fonction objectif de la méthode d'optimisation F .

$\Delta E = F(S') - F(S)$. T est la température, elle est considérée dans cette étude comme un paramètre de contrôle. Ce principe de probabilité est utilisé pour sélectionner le déplacement vers le haut et permettre l'évasion des minimums locaux.

Les avantages du recuit simulé sont :

- Processus assez arbitraire sur les conditions limites et les contraintes imposées sur les fonctions objectif.
- Adaptable à différents types de problèmes.

Les inconvénients du recuit simulé sont :

- Lenteur du processus d'optimisation (système de refroidissement d'un solide).
- Difficulté de choisir les paramètres d'une manière efficace (non déterministe).

La figure 2.4 illustre l'organigramme de la méthode SA.

2.2 ALGORITHME D'ESSAIM DE PARTICULES (PSO)

Dans cette section nous présentons le principe et les étapes de l'algorithme PSO.

2.2.1 Principe et généralités sur la méthode PSO

Dans PSO le mouvement des particules est fait de telle manière qu'il permet de trouver incrémentalement une meilleure solution. Ce résultat est atteint principalement par l'interaction sociale entre les particules. Ces dernières se déplacent continuellement durant le

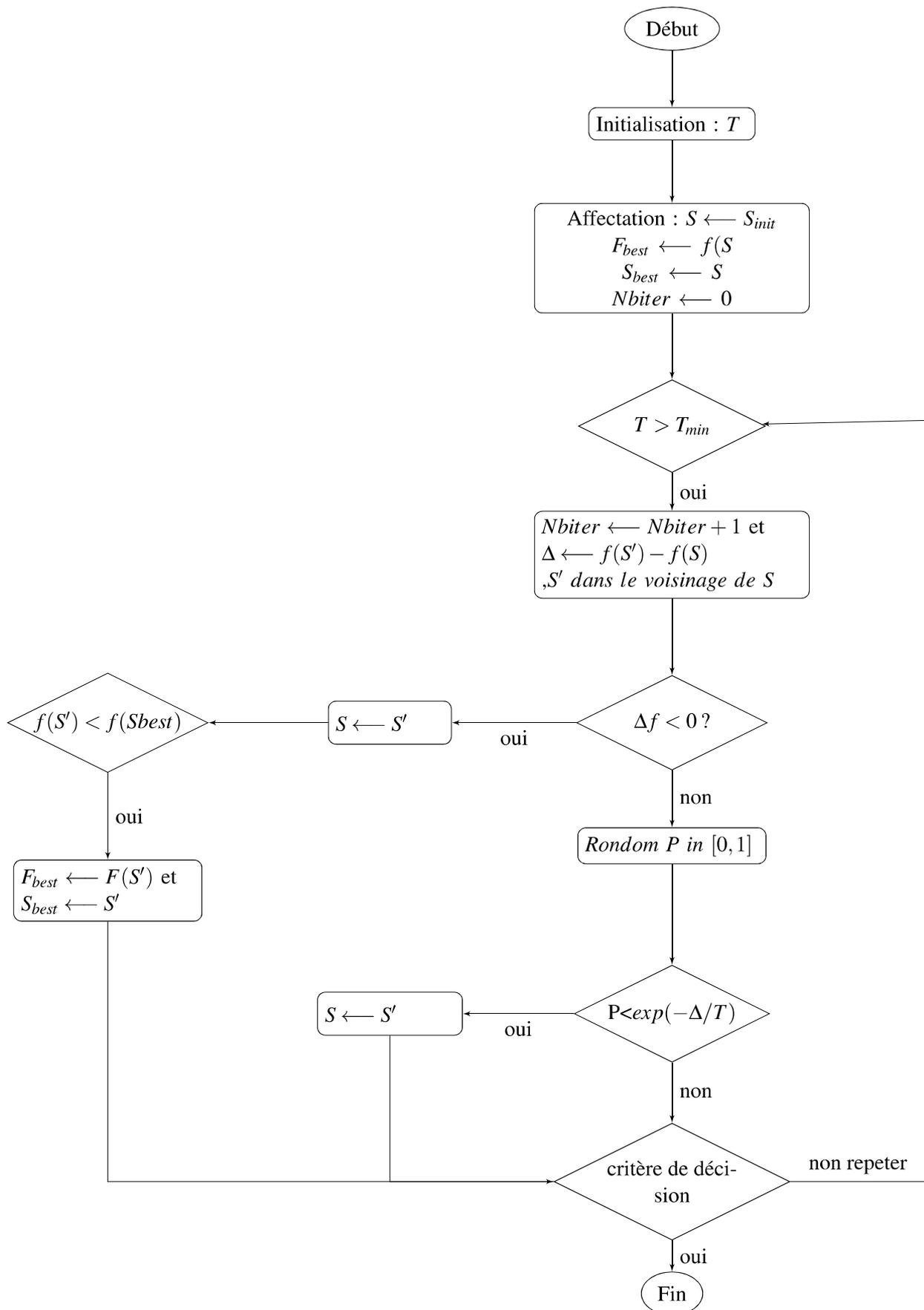


FIGURE 2.4 – L'organigramme de la méthode SA.

processus d'optimisation en utilisant une notion de coopération, afin d'essayer de découvrir des solutions améliorées.

Pour expliquer le principe de PSO, l'exemple du gros poisson au plus profond de l'étang sera une bonne illustration du phénomène : A chaque pas de temps, un pêcheur A indique à son collègue l'endroit le plus profond qu'il a déjà trouvé. Au début il n'y a pas de différence car chacun des pêcheurs suit sa propre route indépendamment de l'autre. Ensuite, quand le pêcheur B trouve un meilleur emplacement, le pêcheur A tend à aller vers lui, ce qui détériore sa position. Dès lors, il a un compromis à faire entre la meilleure position qu'il a trouvée jusque-là et celle trouvée par B. Néanmoins, durant le processus le pêcheur A continue d'aller vers B avec un certain pas, de cette manière il peut échapper à des minima locaux.

Dans cet exemple on trouve des notions essentielles telles que la position, la vitesse (opérateur qui modifie la position), l'échange d'informations, la mémoire (très limitée) et la capacité rudimentaire à combiner les informations pour prendre une décision.

La vitesse est calculée comme combinaison linéaire de trois éléments. Elle est appliquée à la position courante pour donner une nouvelle position. L'efficacité de la méthode tient en particulier au caractère aléatoire des deux derniers coefficients.

La mise à jour de la vitesse en utilisant la notion d'inertie, est donnée par l'équation suivante :

$$\begin{cases} v(t+1) = p * v(t) + \varphi_1 * (p(t) - x(t)) + \varphi_2 * (g(t) - x(t)) \\ x(t+1) = x(t) + v(t+1) \end{cases}$$

Dans notre contexte, l'approche utilisée pour le contrôle de la convergence dans PSO présente l'utilisation d'un coefficient de constriction χ dans l'équation suivante :

$$\begin{cases} v(t+1) = \chi * (w * v(t) + \varphi_1 * (p(t) - x(t)) + \varphi_2 * (g(t) - x(t))) \\ x(t+1) = x(t) + v(t+1) \end{cases}$$

$$\chi = \frac{2}{|2 - \varphi - \sqrt{\varphi^2 - 4\varphi}|}$$

où $\varphi = c_1 + c_2$ et $\varphi > 4$.

2.2.2 Étapes de l'algorithme PSO classique

Dans l'algorithme essaim de particules, chaque particule i est traitée comme un point dans un espace de dimension D avec une position X_i , une vitesse V_i et la meilleure position de la particule X_{besti} . La meilleure position de toutes les particules de l'essaim est représentée par le vecteur X_{gbest} .

$X_i = (x_{i1}, x_{i2}, \dots, x_{id})$ la position de la particule.

$V_i = (v_{i1}, v_{i2}, \dots, v_{id})$ la vitesse de la particule.

$X_{besti} = (p_{i1}, p_{i2}, \dots, p_{id})$ la meilleure position de la particule.

$X_{gbest} = (p_{g1}, p_{g2}, \dots, p_{gd})$ la meilleure position globale de l'essaim.

$i = 1, 2, \dots, n$, n est la dimension du vecteur X_i

d représente le nombre de particules $d \in D$ avec D la dimension de l'espace de recherche.

$$V_{id}(t+1) = \chi(V_{id}(t) + \rho_1[X_{besti}(t) - X_i(t)] + \rho_2[X_{gbest}(t) - X_i(t)])$$

$$X_{id}(t+1) = X_{id}(t) + V_{id}(t+1) \quad (2.1)$$

$\rho_1 = c_1 r_1$ et $\rho_2 = c_2 r_2$

c_1 et c_2 : éléments d'accélération positive appelés paramètre social.

χ : coefficient de constriction.

r_1 et r_2 : nombres aléatoires dans l'intervalle $[0, 1]$

La figure 2.5 illustre le déplacement de la particule dans le cas de PSO classique.

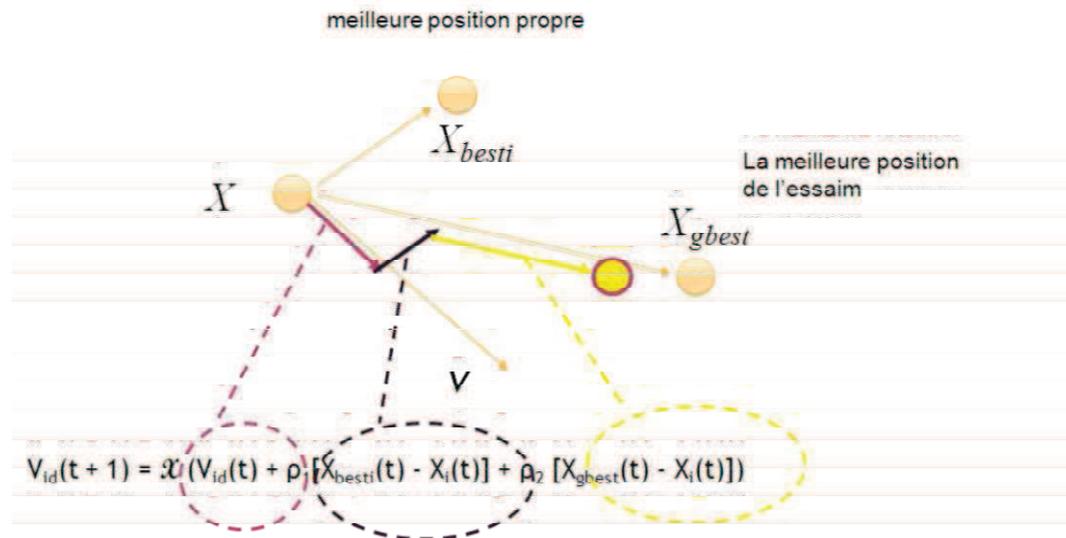


FIGURE 2.5 – Déplacement des particules dans PSO classique.

2.3 ESSAİM DE PARTICULES MODIFIÉ MPSO

2.3.1 Introduction

La modification de la fonction de la vitesse est réalisée en utilisant un nouveau terme dans l'équation 1, ce qui a été introduit par Bochenek et Fory [Bochenek et Fory's (2006)] définie comme :

$X_{Nbest} = (p_{n1}, p_{n2}, \dots, p_{nd})$ La meilleure position du voisinage.

L'équation 2.1 devient :

$$V_{id}(t+1) = \chi(V_{id}(t) + \rho_1[X_{besti}(t) - X_i(t)] + \rho_2[X_{gbest}(t) - X_i(t)] + \rho_3[X_{Nbest}(t) - X_i(t)]) \quad (2.2)$$

$$\rho_3 = c_3 r_3$$

c_3 : accélération positive appelée paramètre sociale .

r_3 : nombres aléatoires dans l'intervalle $[0, 1]$

La figure 2.6 illustre le déplacement de la particule dans le cas de PSO modifié.

L'initialisation de l'essaim et de la vitesse est effectuée de façon aléatoire dans l'espace de recherche. Les meilleures positions sont initialement définies suivant l'essaim initial. Après la première incrémentation, toutes les particules sont déplacées chacune par sa propre vitesse V_i de l'équation (3). Ensuite l'algorithme cherche les optimums en mettant à jour

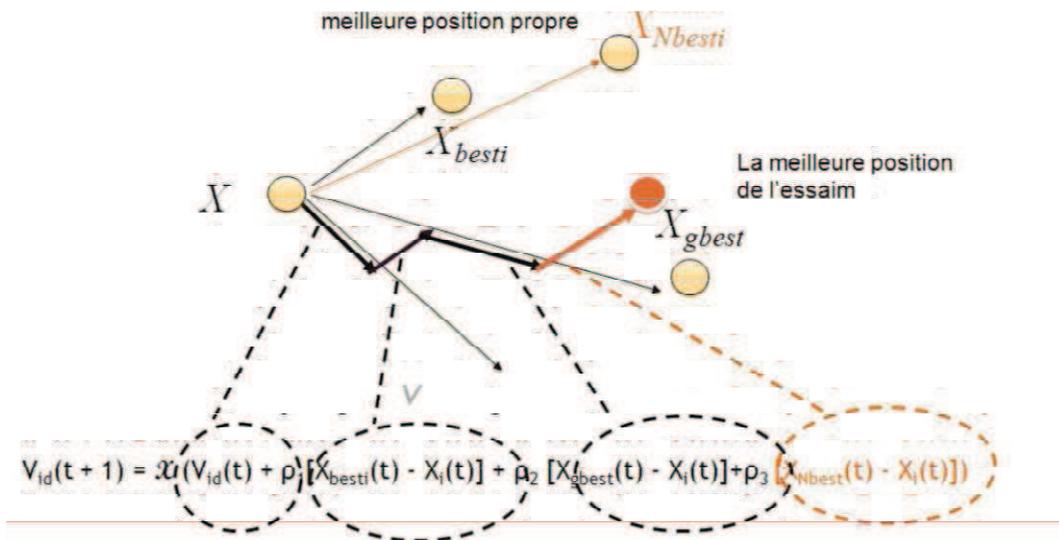


FIGURE 2.6 – Déplacement des particules dans PSO Modifié.

l'état des particules de la nouvelle itération. Les accélération stochastiques c_1 , c_2 et c_3 contribuent respectivement au déplacement de chaque particule vers X_{besti} , X_{gbest} et X_{Nbest} . c_1 représente la confiance de la particule en soi, c_2 représente la confiance que la particule a pour l'essaim, et c_3 représente la confiance que la particule a pour son voisinage.

Dans la plupart des cas, les paramètre d'accélération c_1 , c_2 et c_3 sont affectés à la valeur 1, cependant, si nous voulons éliminer l'influence de la meilleure position de la particule nous prenons $c_1 = 0$; $c_2 = 1$ et $c_3 = 1$ ou d'éliminer l'influence de l'essaim nous prenons $c_1 = 1$; $c_2 = 0$ et $c_3 = 1$ ou éliminer l'influence du voisinage nous prenons $c_1 = 1$; $c_2 = 1$ et $c_3 = 0$. Selon les problèmes à résoudre, nous pouvons faire les choix appropriés pour ces paramètres afin de modifier la vitesse et de promouvoir la convergence.

2.3.2 Étapes de l'algorithme

Les étapes de la méthode PSO peuvent se résumer comme suit :

- Première étape, on initialise l'essaim de particules dans l'espace de recherche d'une manière aléatoire, ce qui n'est pas forcément simple car ça dépend de la forme de l'espace de recherche.
- Deuxième étape : initialisation des vitesses, de manière aléatoire. On suppose que l'essaim est de taille constante.
- Troisième étape : pour chaque particule, on définit son voisinage. Puis on déplace les particules suivant la formule de la vitesse qui change à chaque itération.

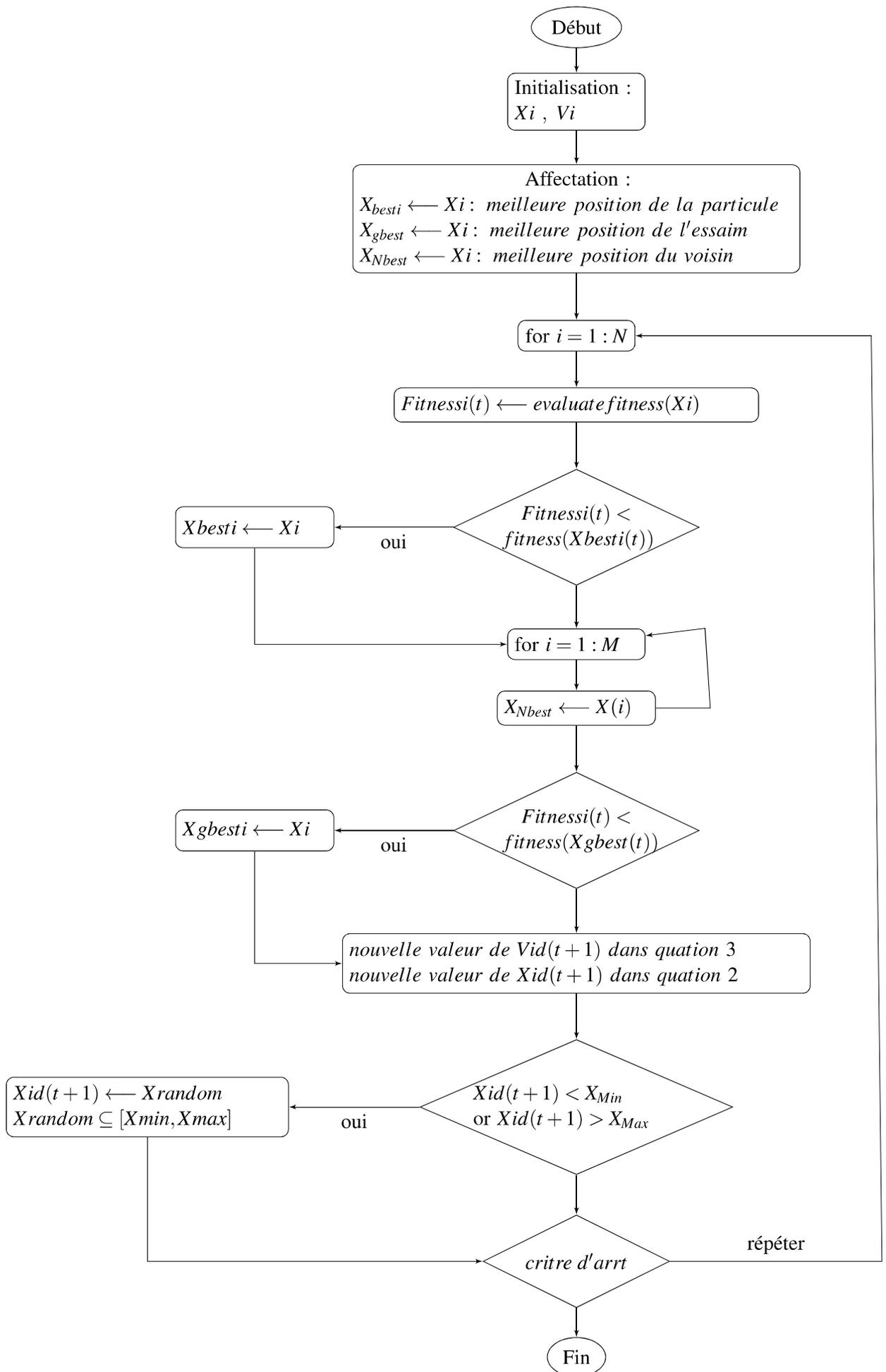


FIGURE 2.7 – L'organigramme de la méthode MPSO

2.4 ALGORITHME PROPOSÉ : ESSAIM DE PARTICULES COUPLÉ AVEC LE RECUIT SIMULÉ

MPSO-SA est une nouvelle méthode d'optimisation évolutionnaire proposée [Elhami et al. (2010)]. Elle est issue d'une hybridation de la méthode essaim de particules modifié (MPSO) et le recuit simulé (SA). Le but de cette méthode est d'accroître la diversité de la population et d'améliorer la convergence.

L'algorithme MPSO commence par une génération (aléatoire) d'une nouvelle population de particules. Cette population peut être améliorée par l'algorithme SA pour donner des solutions qui jouent le rôle de la nouvelle génération. La simulation est répétée jusqu'à ce que le critère d'arrêt soit atteint (aucune amélioration de la solution en cours). Durant le processus, le recuit simulé est adapté pour être utilisé dans l'hybridation suivant les points ci-dessous :

- L'algorithme SA est utilisé comme une méthode améliorant la meilleure particule du voisinage X_{Nbest} et la meilleure position de l'essaim X_{gbest} .
- Le paramètre température T est défini au sein de l'algorithme MPSO. A chaque itération la valeur de la température diminue pour que SA fait ses traitements dans chaque palier de température.
- Le nombre d'itérations dans SA est réduit pour diminuer le temps de calcul.

Les meilleures positions des particules calculées dans MPSO sont données à SA. L'algorithme SA commence sa recherche avec ces positions comme points initiaux. Le mécanisme de recherche commence avec une valeur élevée de la température. Il accepte des solutions données par SA qui ne sont pas nécessairement bonnes, car cela augmente la diversité de l'essaim. Ensuite durant le processus la température diminue. L'amélioration des positions devient de plus en plus probable et la recherche est orientée vers les positions qui ont montré un intérêt important.

L'algorithme MPSO-SA peut être illustré dans le schéma suivant :

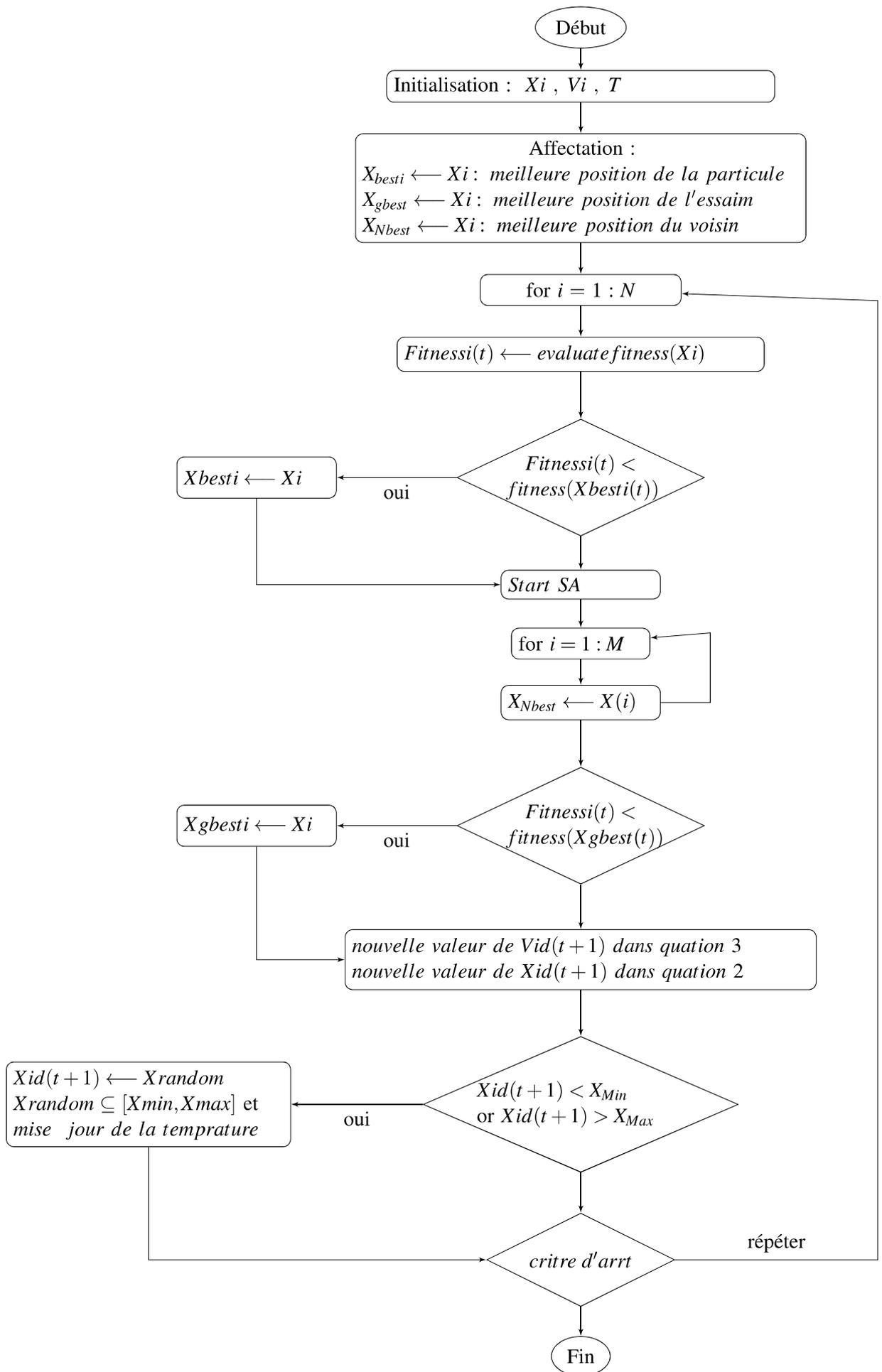


FIGURE 2.8 – L'organigramme de la méthode MPSO SA

2.5 RÉSULTATS ET PERFORMANCE DE LA MÉTHODE MPSO-SA PROPOSÉE

Pour illustrer les performances de la méthode MPSO-SA proposée, une comparaison avec MPSO et SA est effectuée, voir les tableaux 2.1, 2.2, 2.3 . Des fonctions tests connues dans la littérature sont présentées dans l'annexe A2. Ces fonctions ont des tailles différentes. Afin de donner des indications de performance relatives à chaque méthode, 100 essais ont été effectués, les valeurs illustrées dans les tableaux ci-dessous sont des moyennes calculées à la base de l'ensemble des essais.

Dans ce cadre expérimental, les résultats obtenus sont résumés dans les tableaux suivants avec les valeurs moyennes [Elhami et al. (2010)].

Plate forme de simulation Le PC :

- Processeur : Intel(R) Core (TM) 2 Duo P8600 et 2,40GHz
- 3 Go de RAM

Le logiciel : Matlab version : 7.0.1 sous Windows.

Dans les tableaux des résultats, on utilise les notions suivantes :

- SR (Success Rate) est le taux de succès, c'est le pourcentage de convergence de la fonction vers la bonne solution.
- SD (Standard Deviation) est l'écart type il est calculé suivant la formule suivante :

$$SD = \sqrt{1/n \left(\sum_{i=1}^n (x_i - x^*)^2 \right)}$$

Avec x^* la solution optimale et x_i la solution trouvée à chaque essaie.

- EvalF est le nombre d'évaluation de la fonction objectif.

Analyse des résultats

Les résultats présentés dans les trois tableaux 2.1, 2.2 et 2.3 font ressortir les éléments suivants :

- MPSO-SA peut obtenir la solution optimale avec une probabilité plus élevée.
- Le temps de calcul dans MPSO-SA est plus faible que les autres méthodes SA et MPSO utilisées toutes seules.
- SA peut garantir la solution optimale lorsque la taille du problème est petite, mais elle prend plus de temps lorsque la taille du problème devient assez grande.

TABLE 2.1 – Résultats du recuit simulé

FTest	Dim	SA			
		SR	SD	Time(s)	EvalF
SH3	3	80	10.7	40	1342250.6
SH4	4	20	6.76e+003	69	2074982.6
GR	50	10	17.0684	9.6e+002	2814728.3
RO	10	23	44.9	78	2796580.5
HAN	2	100	0.0308	50	1945584.2
SK5	4	60	1.7	2.1e+002	2205679.4
SK7	4	70	0.436	2.8e+002	2195135
SK10	4	100	0.00227	3.6e+002	2104962.2
HA3	3	100	9.02e-005	85	2440969
HA6	6	20	0.00684	5.6e+002	13014268.2
DE	3	100	6.59e-006	47	2463189
CA	2	100	7.45e-007	52	2463189
GP	2	100	9.49e-006	51	2272379.8
CM	4	50	0.000296	55	2438221.8
BR	2	100	4.53e-007	49	2463189
BO1	2	100	6.65e-007	49	2463189
BO2	2	100	4.06e-006	49	2463189
RA	2	30	0.00447	48	2305790.6

TABLE 2.2 – Méthode d'essai de particules modifiée

FTest	Dim	MPSO			
		SR	SD	Time(s)	EvalF
SH3	3	100	3.58e-012	3.6	229284.02
SH4	4	100	1.05e-011	4.6	269959.4
GR	50	40	0.446	20	582084.6
RO	10	62	1.64	7.1	476140
HAN	2	100	1.27e-014	3.1	241828.2
SK5	4	50	3.13	7.7	188575.6
SK7	4	70	3.2	8.9	170397.6
SK10	4	70	3.48	10	139824.58
HA3	3	100	6.49e-015	4.6	248340.8
HA6	6	51	0.0598	5.5	239534.8
DE	3	100	0	5.4	476140
CA	2	100	9.93e-017	2	156030
GP	2	100	5.58e-015	2.6	203344.4
CM	4	100	9.02e-015	3.1	246113
BR	2	100	0	1.2	99409.8
BO1	2	100	0	0.71	61088
BO2	2	100	0	0.72	63182.4
RA	2	100	0	0.83	69802.6

TABLE 2.3 – Résultats de l'hybridation MPSO-SA

FTest	Dim	MPSO-SA			
		SR	SD	Time(s)	EvalF
SH3	3	100	3.69e-012	3.4	137372.08
SH4	4	100	7.4e-011	4.3	455889.6
GR	50	100	1.6e-010	18	570334.2
RO	10	91	1.02	4.9	243578
HAN	2	100	2.6e-013	3.1	364928
SK5	4	58	5.7	6.9	256903
SK7	4	82	3.25	9.3	198789
SK10	4	83	5.76	8.6	176896
HA3	3	100	6.55e-015	2.9	80846.1
HA6	6	61	0.132	3.4	76704.18
DE	3	100	0	3.4	213560
CA	2	100	1.57e-016	1.9	83971.2
GP	2	100	1.06e-014	2	86159.2
CM	4	100	1.16e-014	0.74	32718.4
BR	2	100	0	1.2	57040
BO1	2	100	0	0.73	34725.9
BO2	2	100	0	0.74	35043.9
RA	2	100	0	0.61	28264

- MPSO peut résoudre des problèmes plus importants dans un temps acceptable par rapport au SA.
- Les résultats révèlent que le nombre d'évaluations de la fonction objectif de MPSO-SA est moins important que les autres algorithmes SA et MPSO.
- La comparaison conclue que MPSO-SA est en mesure d'améliorer efficacement les performances de calcul des problèmes d'optimisation complexes.
- Ces résultats indiquent que l'algorithme proposé est une alternative intéressante pour résoudre des problèmes d'optimisation.

2.6 COMPARAISON DES RÉSULTATS

2.6.1 Comparaison des taux de succès

Le taux de succès donne une réponse concrète sur la convergence de l'algorithme (résultat approché de la solution réelle).

Le taux de succès pour chaque fonction est calculé sur 100 essais. Les différents taux sont représentés sur la figure 2.9. La courbe en traits continus traçant les taux de succès de l'algorithme MPSO-SA est la plus proche du 100 %. Dans la quasi totalité des essais, l'al-

gorithme MPSO-SA converge vers la bonne solution.

La courbe qui correspond à l'algorithme MPSO est plus proche de celle de MPSO-SA, les résultats montrent que cette algorithme est moins performant que l'hybridation au niveau de la convergence. La courbe qui concerne l'algorithme SA montre que ce dernier ne converge que rarement vers la bonne solution. Surtout lorsque la dimension des fonctions tests est grande.

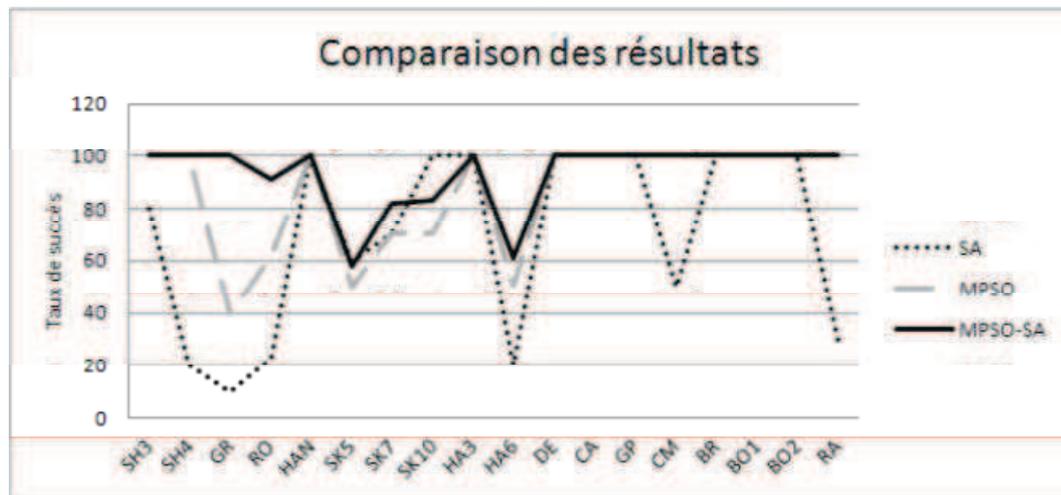


FIGURE 2.9 – Comparaison des Taux de succès.

2.6.2 Comparaison du temps de calcul

L'algorithme du recuit simulé est connu par sa lenteur de convergence. Ce constat est vérifié dans les résultats illustrés de la figure 2.11. Pour la fonction GRIENWANK dont la dimension est 50 l'algorithme SA prend 960 secondes (Voir la plateforme de simulation début de section) en moyenne pour donner la solution.

Le temps de calcul de l'algorithme MPSO est meilleur que SA.

Les premières implémentations de l'hybridation étaient pénalisées par le SA, qui consomme beaucoup de temps CPU. Des efforts ont été fournis pour alléger l'algorithme SA, avec un changement des paramètres, afin d'avoir un algorithme performant.

2.6.3 Comparaison du nombre d'évaluations de la fonction objectif

Pour la résolution des problèmes d'optimisation complexes et de grandes tailles, un nombre réduit d'évaluations de la fonction objectif devient un critère incontournable.

En terme de nombre d'évaluations, l'hybridation MPSO-SA offre des performances nettement meilleures que l'algorithme MPSO pour la majorité des fonctions tests. L'algorithme

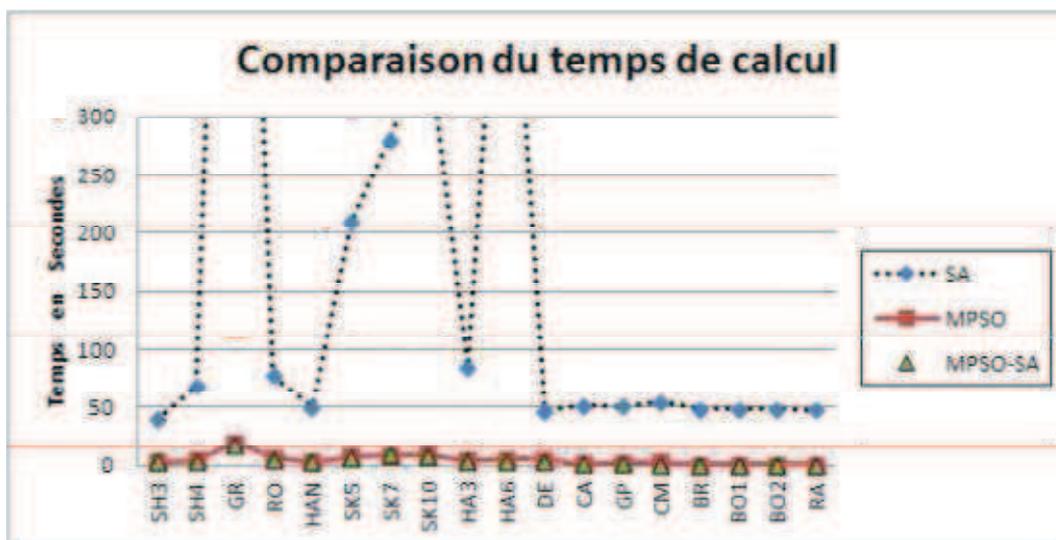


FIGURE 2.10 – Comparaison du temps de calcul.

SA est beaucoup moins performant que les deux autres algorithmes se basant sur l'essaim de particules.

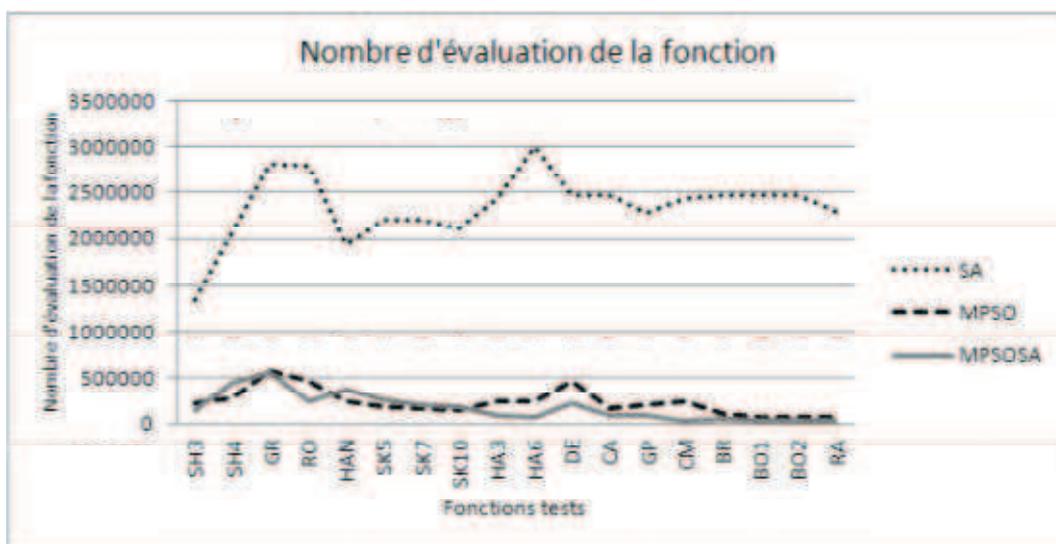


FIGURE 2.11 – Comparaison du nombre d'évaluation de la fonction objectif.

2.7 MODÉLISATION ET IMPLÉMENTATION SOUS JAVA

Dans un souci d'augmentation des performances de la méthode MPSO-SA, on a procédé à son implémentation sous JAVA afin de bénéficier des threads.

La modélisation de l'algorithme MPSO-SA en java est effectuée en créant trois paquets :

- Le premier paquet "Fonction" concerne les fonctions tests utilisées pour le calcul de performance de l'algorithme..

- Le deuxième paquet "MPSOSA" est le coeur du programme, il contient les classes de bases qui font appels a plusieurs fonctions utiles pour tourner l'algorithme d'optimisation ;
- Le troisième paquet "IHM" concerne l'interface utilisateur.

La figure 2.12 représente la gestion de flux entre les trois paquets.

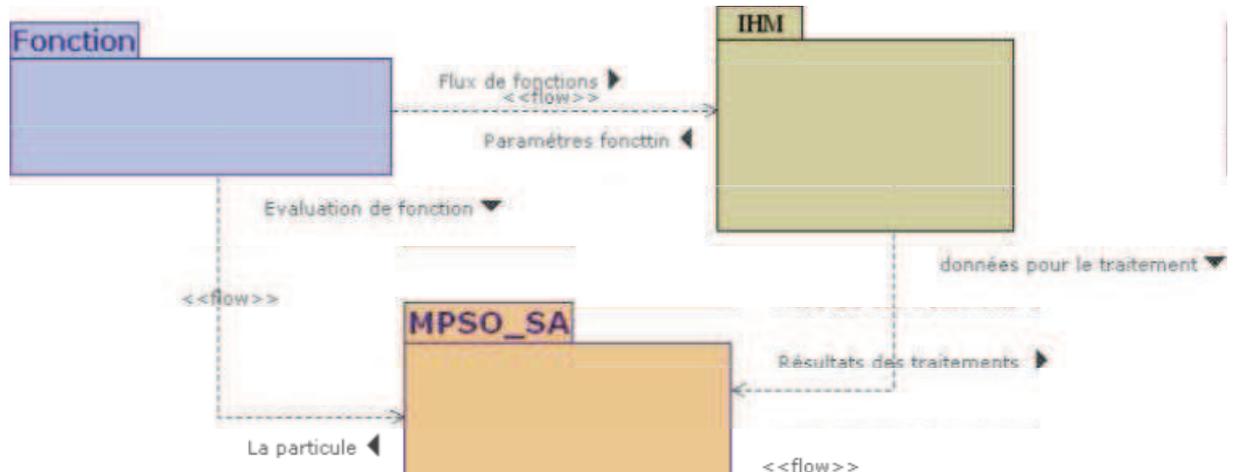


FIGURE 2.12 – Gestion de flux entre les paquetages.

La figure 2.13, représente le diagramme de classes du paquetage fonctions.

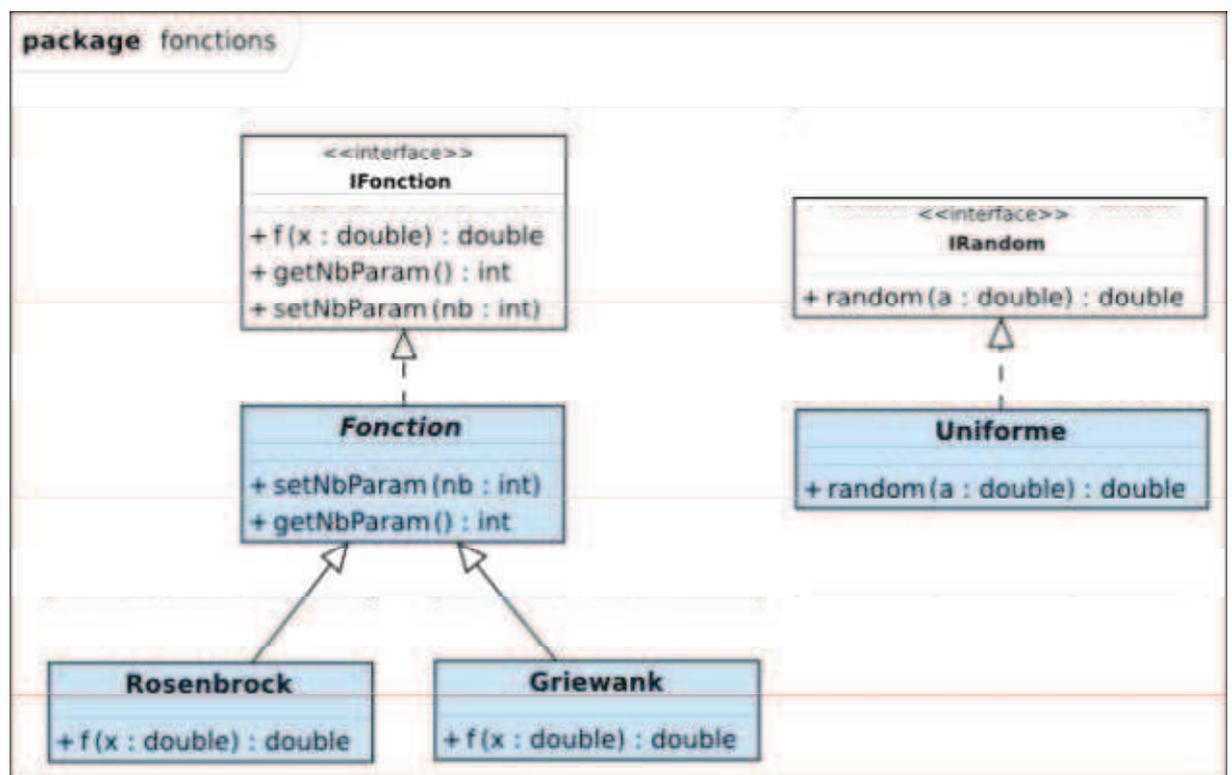


FIGURE 2.13 – Diagramme de classes du paquetage fonctions.

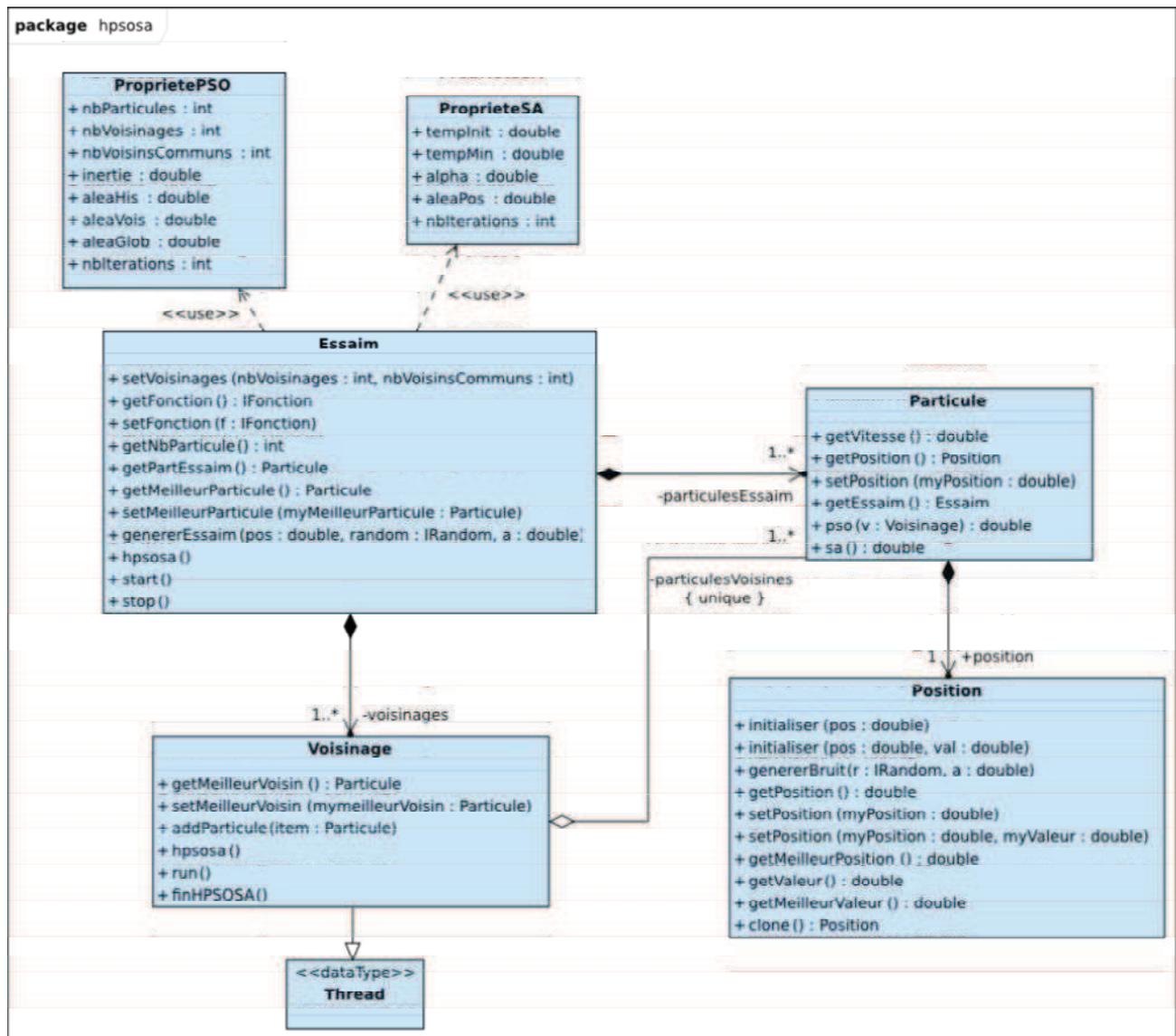


FIGURE 2.14 – Diagramme de classes du paquetage MPSOSA.

2.8 AMÉLIORATION DU TEMPS D'EXÉCUTION VIA LES THREADS

Pour améliorer le temps d'exécution on a utilisé les processus légers de JAVA (threads en anglais). Un thread est en quelque sorte un processus à l'intérieur d'un processus. Les ressources allouées à un processus (temps processeur, mémoire) vont être partagées entre les threads qui le composent. Un processus possède au moins un thread. Contrairement aux processus, les threads partagent la même zone mémoire (espace d'adressage), ce qui rend très facile la communication entre threads.

Chaque thread possède son propre environnement d'exécution ainsi qu'une pile (variables

locales).

Le comportement des threads est différent dans chaque système d'exploitation (Windows NT, UNIX). Certains langages, comme JAVA, définissent leur propre mécanisme de threads, afin de permettre l'utilisation facile et portable des threads sur tous les systèmes.

Les threads font partie intégrante du langage JAVA. Elles sont gérées grâce à la classe Thread.

En Java le multithreading signifie qu'il faut

- créer un thread,
- écrire la tâche exécutée par le thread,
- lier la tâche au thread.

Pour réaliser cela, il faut étudier et comprendre la classe Thread.

Avec plusieurs piles d'appels, on a l'impression que plusieurs choses se passent en même temps. Dans un système multiprocesseurs cela serait réellement possible, mais pas sur un système monoprocesseur. Ce qui est rendu possible par les threads.

La classe Thread offre un certain nombre de méthodes pour contrôler le comportement des threads.

- La méthode statique *sleep(int d)* de la classe Thread force le thread en cours d'exécution à passer dans l'état bloqué pendant un temps "d" en ms. L'ordonnanceur choisira alors un autre thread dans la liste des threads exécutables.
- *isAlive()* : retourne vrai si le thread auquel on applique la méthode est vivant (c'est à dire a été démarré par start() et que sa méthode run() n'est pas encore terminée. Le thread vivant est donc prêt, bloqué ou en cours d'exécution.
- *getPriority()* et *setPriority(int prio)* permettent de manipuler la priorité du thread.

Lorsque plusieurs threads travaillent sur des mêmes ressources, il est parfois utile ou nécessaire de limiter le parallélisme en assurant qu'un certain objet ne subisse pas en même temps plusieurs séquences d'actions concurrentes.

Nous avons mentionné que les Threads partagent les mêmes ressources (mémoire, variables du processus etc.). Dans certaines situations, il est nécessaire de synchroniser les threads

pour obtenir un résultat cohérent.

Prenons l'exemple d'un avion où il reste une seule place disponible et deux clients se présentant à deux différents guichets. Si la place est proposée au premier client et que ce dernier prenne tout son temps pour réfléchir, nous n'allons pas attribuer cette place au second client qui en a fait la demande. De ce fait, nous synchronisons l'accès à la méthode de réservation de telle manière à ne pas attribuer la même place aux deux voyageurs ou bien au second voyageur avant que le premier ne prenne sa décision de la prendre ou pas.

Pour notre programme on a utilisé les threads au niveau de la classe voisinage.

2.9 INTERFACE HOMME-MACHINE

L'interface graphique d'utilisation permet le paramétrage de l'algorithme. Les fonction tests et les paramètres de l'algorithme peuvent être choisis et modifié pour lancer la recherche d'optimum (voir figure 2.15).

Une fois le bouton "Configuration" est pressé, un essaim est instancié avec les paramètres définis par l'utilisateur. Le bouton "Start" est activé pour permettre de lancer l'optimisation.

Le bouton "Afficher position" est également activé, permettant de montrer la meilleure particule dans l'essaim en cours.

Lors de l'exécution de l'algorithme par cette interface. Les valeurs de la fonction sont affichées en temps réel et les boutons Start et Configurer sont désactivés alors que le bouton "Stop" est activé pour arrêter l'exécution à tout moment.

La figure 2.15 affiche la position de l'essaim une fois arrêté et après avoir cliqué sur le bouton "Afficher position". La position est représentée sous la forme d'un vecteur colonne. On peut ensuite choisir de reconfigurer l'essaim et de lancer une nouvelle simulation.

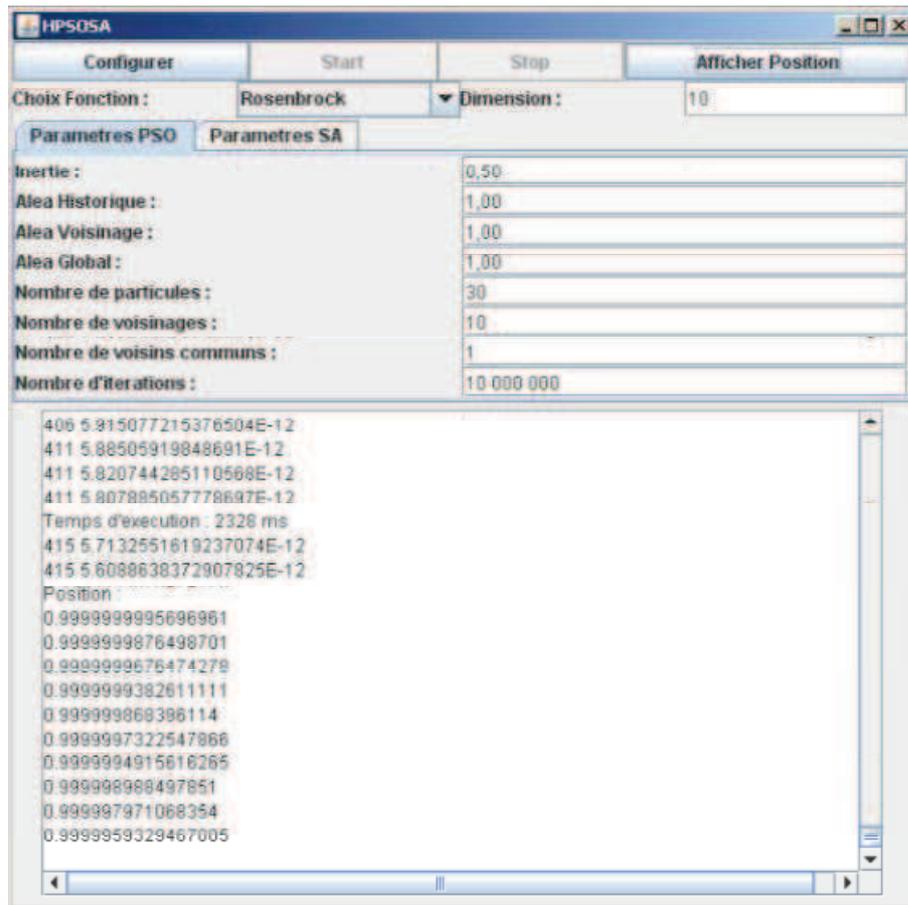


FIGURE 2.15 – Exemple d'affichage de la position..

CONCLUSION

Dans ce chapitre nous avons présenté une nouvelle méthode d'hybridation MPSO-SA basée sur l'idée de la population évolutionnaire de PSO et sur la capacité à chercher une éventuelle amélioration par SA.

Les résultats de la simulation effectuée ont indiqué que l'inconvénient du PSO classique est réduit par la méthode MPSO.

MPSO-SA a amélioré l'optimisation globale en mettant un équilibre entre l'exploration du domaine de recherche et l'exploitation locale des solutions trouvées.

L'hybridation MPSO-SA utilise d'une part, MPSO qui applique le concept de recherche évolutionnaire ce qui engendre une amélioration des individus par une coopération de la population et une exploration du domaine par une notion de concurrence. D'autre part, elle utilise une recherche adaptable SA pour effectuer une bonne exploitation des solutions trouvées.

La méthode hybride proposée MPSO-SA a l'avantage de fournir de meilleurs résultats par

rapport à la méthode d'essaim de particules modifiée et le recuit simulé seul. Pour montrer les performances de notre méthode, des tests sur 18 fonctions complexes connues dans la littérature ont été effectués. Ces fonctions sont sans contraintes mais avec des variables bornées. Une comparaison numérique avec les trois algorithmes est donnée à la fin de ce chapitre.

Les résultats des simulations et comparaisons avec les trois approches ont démontré la supériorité de la proposition de MPSO-SA en termes de qualité, de robustesse et de faible nombre d'évaluation de la fonction objectif.

Notre méthode proposée est une solution prometteuse et nous encourage à l'appliquer à d'autres problèmes d'optimisation globale avec contraintes.

Nous présentons des applications de cette méthode dans les chapitres suivants, chapitre 3 : "Application de la méthode MPSO-SA à la mécanique des structures" et chapitre 4 : "Optimisation de l'interopérabilité des systèmes d'information".

3 APPLICATION DE LA MÉTHODE MPSO-SA À LA MÉCANIQUE DES STRUCTURES

SOMMAIRE

INTRODUCTION	57
3.1 INCERTITUDES EN FIABILITÉ DES STRUCTURES	58
3.2 FIABILITÉ DES STRUCTURES	60
3.2.1 Modélisation d'un problème de fiabilité des structures	60
3.2.2 Probabilité de défaillance d'une structure	61
3.2.3 Calcul de la probabilité de défaillance d'une structure	61
3.2.3.1 Méthode de Monte-Carlo	62
3.2.3.2 Indice de fiabilité	63
3.3 RÉSULTATS DES APPLICATIONS DE FIABILITÉ DES STRUCTURES	65
3.3.1 Exemple 1 : Etude dynamique d'une plaque	65
3.3.2 Exemple 2 : Etude dynamique d'un ressort d'embrayage	69
CONCLUSION	72

INTRODUCTION

La prise en compte des incertitudes dans l'analyse fiabiliste des structures est nécessaire pour un dimensionnement optimal et fiable des structures. C'est pour cet objectif que les approches probabilistes pour la mécanique des structures sont développées depuis plusieurs années.

La complexité des modèles mécaniques (comportement non-linéaire, dynamique, fatigue, mécanique de la rupture, etc.) nécessite l'utilisation des méthodes de couplage mécano-fiabiliste. Ces méthodes nécessitent à leurs tours un appel au calcul déterministe du modèle mécanique pour chaque variable aléatoire. Ce nombre d'appels augmente généralement avec le nombre de variables aléatoires, ainsi le temps de calcul devient très excessif en utilisant les méthodes classiques (simulation de Monté Carlo). Pour cela, plusieurs méthodes fiabilistes sont développées dans le sens de minimiser le temps de calcul et le nombre de simulations [Haldar (2000)].

Ce chapitre définit les différentes méthodes de résolution des problèmes de fiabilité utilisées actuellement. Par la suite, la résolution de problèmes de fiabilité par notre méthode hybride MPSO-SA sera exposée à la fin de ce chapitre.

3.1 INCERTITUDES EN FIABILITÉ DES STRUCTURES

L'étude des structures mécaniques repose sur les deux points suivants :

- L'observation des réponses de la structure considérée par rapport aux différentes actions que nous pouvons lui appliquer ;
- La modélisation de cette structure.

Dans cette étude, il est primordiale de mettre en œuvre les comportements aléatoires, afin de prendre en compte les phénomènes mécaniques mis en jeu. Sur une grandeur mesurable, les incertitudes peuvent être classés en trois types :

- incertitudes naturelles ;
- incertitudes de mesure ;
- incertitudes de modélisation.

Incertitude naturelle

Les aléas naturelles peuvent être classés en deux catégories. La première concerne les quantités dont la valeur est incertaine, ces quantités proviennent en général du processus de fabrication de la structure mécanique (ex : module d'Young, densité de matériau,...). La deuxième représente les quantités qui quantifient les variables représentant les actions agissant sur la structure.

Incertitude de la mesure

L'origine de cette incertitude vient de l'expérimentation. Elle est affectée par les conditions d'identification des grandeurs physiques. L'étude fiabiliste des structures ne prend pas en charge les erreurs de mesure, car elles n'interviennent pas réellement dans le comportement de la structure par rapport à son environnement. Cependant, elles sont bien présentes dans l'étude du système et doivent être intégrées dans le processus d'identification des paramètres. Ce sont des aléas maîtrisés puisque elles sont connus par la précision des moyens de mesure, et elles sont en général inférieurs à la précision désirée sur la mesure.

Les facteurs qui influencent ces incertitudes peuvent être, la précision de la mesure, la dispersion statistique ou sur l'erreur systématique.

- résolution de l'instrument de mesure ;
- la mesure est entachée d'une erreur aléatoire fonction de l'erreur d'échantillonnage, de l'erreur de préparation et de la stabilité de l'appareil ;

- reprenant les erreurs d'échantillonnage et de préparation ainsi que les problèmes d'éta-
lonnage provoquant l'apparition d'offset.

Les dispersions venant des conditions expérimentales peuvent être atténués par la répétition de la mesure.

Incertitude de la modélisation

La modélisation est originaire de l'imagination humaine, donc entre formulation du problème et réalité physique, il existe un aléa de modélisation. Souvent, les concepteurs utilisent des simplifications d'une physique complexe et délimite le domaine de recherche. Les incertitudes peuvent être classé comme suit :

- Incertitude sur le chargement : Les fluctuations sur la sortie d'un modèle mécanique sont considérées étant les principales causes de défaillance d'un système ;
- Incertitude du modèle : La simplification de la géométrie souvent complexe dans les endroits où une solution est plus précise. Le but est de trouver des solutions avec un temps de convergence raisonnable. La robustesse de la réponse du système face aux incertitudes du modèle est faite à l'aide des analyses de convergence ;
- Incertitude des conditions aux limites : le choix d'un type de conditions aux limites se reflète sur la précision de la solution. Par ailleurs, l'approximation sur les conditions aux limites engendre des erreurs sur le comportement global du système. Le choix peut être fait au départ par souci de simplification ;
- Incertitude sur la résolution : lors de la modélisation des systèmes physiques on a recours à une modélisation d'un comportement complexe non-linéaire spatial ou temporel par des comportements linéaires sur des intervalles réduites. Ce qui induit à des erreurs.
Une autre source d'incertitude dans ce cas est au niveau des calculateurs informatiques, qui font des troncatures sur les grandeurs réelles. Seulement cette incertitude est maîtrisable est son effet peut être diminués avec les progressions technologiques dans le domaine des super calculateurs informatiques ;

L'ensemble de ces incertitudes peut avoir un effet non négligeable sur le résultat final. Cependant la réduction de ces incertitudes est possible lorsque la théorie des phénomènes physiques s'enrichit.

La minimisation des aléas contrôlables passe cependant inévitablement par des coûts, que ce soit en temps de calcul pour les méthodes numériques, pour les erreurs de mesures ou pour les processus de fabrication [Deheeger (28 janvier 2008)], [Duprat (2006)].

3.2 FIABILITÉ DES STRUCTURES

Une structure est réalisée pour répondre à un ensemble de besoins clairement identifiés. Dans ce contexte, la fiabilité des structures a pour objectif d'évaluer la probabilité qu'une structure, soumise à des aléas, soit capable de satisfaire l'intégralité de ces besoins et ce pour une durée de vie donnée. Cependant, le calcul de cette probabilité impose une méthodologie d'étude rigoureuse en quatre étapes :

- Etape 1 : définir un modèle mécanique déterministe adapté au problème traité,
- Etape 2 : identifier les paramètres aléatoires de ce modèle et les modéliser par un outil probabiliste adéquat,
- Etape 3 : définir les modes et scénarios de défaillance du problème,
- Etape 4 : évaluer les probabilités d'occurrence de ces modes de défaillance.

La précision et l'exactitude du résultat, la probabilité de défaillance de la structure, est ainsi conditionnée par un ensemble de facteurs experts, expérimentaux, mathématiques, numériques et pratiques intervenant à chaque étape de l'étude fiabiliste.

3.2.1 Modélisation d'un problème de fiabilité des structures

La première étape d'une étude de fiabilité est d'identifier et de positionner le problème à traiter. Un modèle mécanique déterministe adapté est alors défini. Parmi les différents paramètres intervenant dans ce modèle, certains ne sont connus que d'une manière aléatoire. Par une étude statistique on peut identifier une loi de probabilité à chaque variable aléatoire X . Si le problème possède m variables aléatoires, nous définissons le vecteur aléatoire :

$$X = ((X_1, X_2, \dots, X_m)^T) \quad (3.1)$$

Pour assurer l'intégrité de la structure face aux risques envisagés, des règles de bon fonctionnement sont établies. Le non-respect de l'un de ces critères provoque alors un mode élémentaire de défaillance de la structure. L'enchaînement d'événements entraînant une défaillance constitue un scénario de défaillance. Les modes ou scénarios de défaillance sont établis en fonction des risques potentiels : la présence de contraintes ou des déformations excessives dans un élément structural, la rupture d'une pièce, etc. Plusieurs phénomènes physiques peuvent être à l'origine d'une défaillance : la plastification, la fatigue, le fluage, la présence de fissures, de grandes déformations, etc.

Chacun ou la combinaison de ces phénomènes entraîne un mode ou un scénario de dé-

faillance.

La modélisation probabiliste d'un mode de défaillance se fait par la définition d'une fonction $G(X)$ appelée fonction d'état limite ou fonction de performance. Notons que par construction la fonction $G(X)$ est elle-même une variable aléatoire. Ainsi :

- $G(X) > 0$ définit le domaine de sécurité D_s de la structure,
- $G(X) < 0$ définit le domaine de défaillance D_f de la structure,
- $G(X) = 0$ définit la surface d'état limite.

La structure possède donc deux états possibles : un état de bon fonctionnement et un état de défaillance, séparés par une frontière dite d'état limite. Notons toutefois que la réalité est souvent bien plus complexe que cette modélisation binaire.

3.2.2 Probabilité de défaillance d'une structure

La probabilité de défaillance de la structure est alors égale à

$$P_f = P(G(X) \leq 0) \quad (3.2)$$

$$\begin{cases} P_f = \int_{G(X) \leq 0} \\ f_X(x_1, x_2, \dots, x_n) dx_1 \dots dx_n \end{cases} \quad (3.3)$$

$f_X(x_1, x_2, \dots, x_n) dx_1 \dots dx_n$ est la densité de probabilité conjointe des variables aléatoires de base x_i .

La fiabilité de la structure est définie par :

$$P_s = 1 - P_f \quad (3.4)$$

3.2.3 Calcul de la probabilité de défaillance d'une structure

Actuellement, deux grandes catégories de méthodes d'évaluation de la probabilité de défaillance d'une structure existent. La première a pour objectif d'évaluer P_f sur l'intégralité du domaine de défaillance D_f et fait appel à des simulations basées sur la technique de Monte Carlo. La seconde consiste en une idéalisation du domaine D_f permettant le calcul d'une valeur approchée de P_f à partir d'un indice de fiabilité

3.2.3.1 Méthode de Monte-Carlo

Le calcul direct de P_f est généralement très compliqué. Des techniques de simulations numériques permettent toutefois une évaluation précise de la probabilité de défaillance et sont souvent les seuls moyens de prendre en compte la non-linéarité du modèle mécanique ou de la fonction d'état limite. La plus simple d'utilisation est la méthode de Monte Carlo (cf. figure 3.1). Le principe général de la méthode de Monte Carlo est le suivant :

1. Tirage au hasard d'un nombre y^r uniformément distribué entre 0 et 1. Soit le vecteur aléatoire de conception $X = (X_1, X_2, \dots, X_m)^T$. La fonction de répartition F_{X_i} de chacune des variables X_i étant connue, l'échantillon $x^r = (x_1^r, x_2^r, \dots, x_m^r)^T$ représentation du vecteur aléatoire X_i , est obtenu par :

$$x_i^{(r)} = F_{x_i}^{-1}(y_i^{(r)}) \quad (3.5)$$

2. Évaluation de la défaillance ou du bon fonctionnement de la structure, $G(x^{(r)}) \leq 0$ ou $G(x^{(r)}) > 0$, pour chaque échantillon. $G(x^{(r)})$ est une réalisation de la variable aléatoire $G(X)$.
3. Calcul de la probabilité de défaillance après N_t simulations, soit :

$$P_f = \frac{1}{N_t} \sum_{r=1}^{N_t} I[G(x^{(r)}) \leq 0] \quad (3.6)$$

La fonction indicatrice $I[G(x^{(r)}) \leq 0]$ est égal à :

- ✓ 1 si $G(x^{(r)}) \leq 0$
- ✓ 0 si $G(x^{(r)}) > 0$

Ce résultat est exact lorsque N_t tend vers l'infini. Notons que la convergence de la méthode est très lente, en \sqrt{n} . Par ailleurs, il n'existe pas de règles rigoureuses pour déterminer le nombre exact de simulations nécessaires. En pratique, le nombre de tirages minimum pour obtenir une probabilité de 10^{-n} pour un niveau de confiance de 95% et une erreur admise de 20% sur les résultats est compris entre 10^{n+2} et 10^{n+3} . Plus n est élevé, plus les temps de calcul sont prohibitifs. Ainsi, la méthode de Monte Carlo reste un outil efficace lorsque P_f n'est pas trop faible. Cependant, les simulations de Monte Carlo sont souvent le seul moyen d'étalonnage des méthodes approchées.

Des méthodes de simulations plus élaborées ont été développées récemment pour réduire le nombre de calculs nécessaires et augmenter la précision des résultats. Nous citons les méthodes d'échantillonnage stratifié, de tirages d'importance ou de simulations conditionnelles [Devictor (1996)], [O (1994)], [Héliot et al. (1995)], [A (1990)].

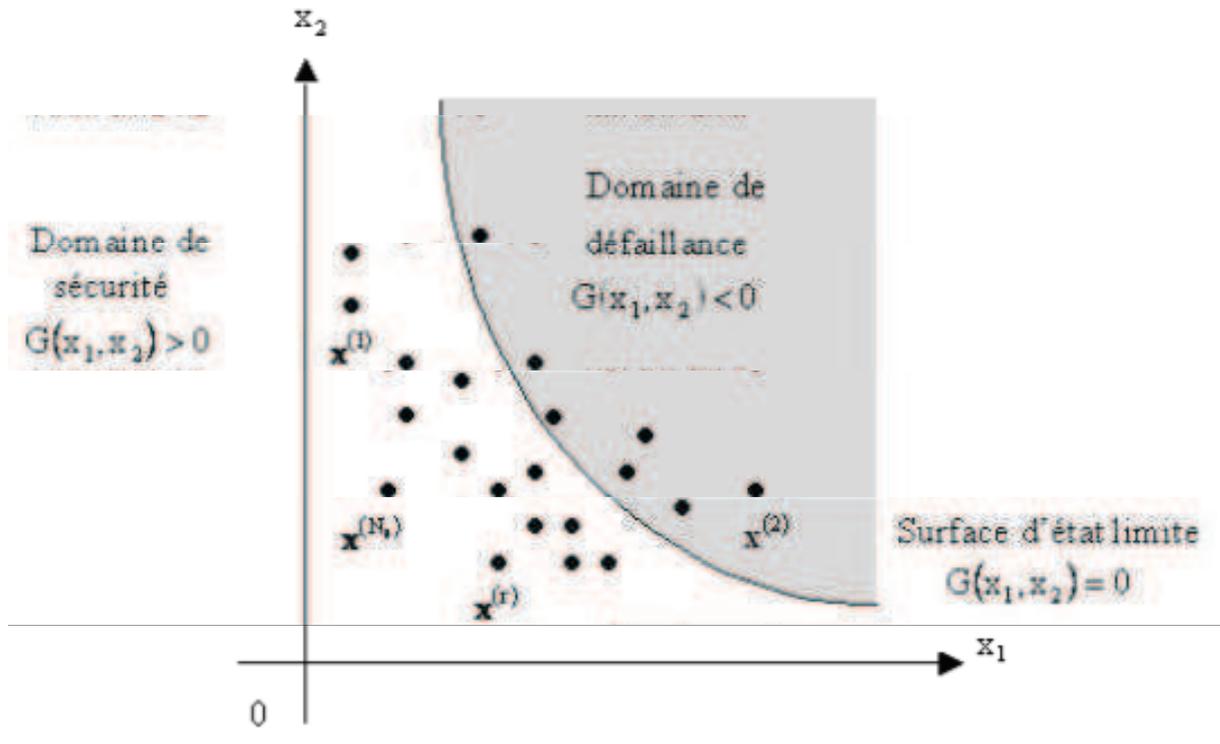


FIGURE 3.1 – Monte Carlo pour un problème à deux variables aléatoires.

3.2.3.2 Indice de fiabilité

Pour calculer la probabilité de défaillance, plusieurs auteurs ont proposé des indices ou index de fiabilité. Les plus connus sont ceux de Rjanitzyne-Cornell et de Hasofer-Lind [HASOFER et al. (1974)]. Si le problème de dépendance vis-à-vis de l'état limite réduit le champ d'application de l'indice de Rjanitzyne-Cornell [Muzeau et al. (1996)], [Proccacia. (1996)], l'indice de Hasofer-Lind est depuis 1974 à la base des méthodes FORM et SORM (First and Second Order Reliability Method).

Indice de Hasofer et Lind

Pour lever le risque de dépendance de l'indice de fiabilité vis-à-vis de l'état limite, Hasofer et Lind ont proposé d'effectuer le calcul de β dans l'espace des variables aléatoires normales centrées réduites et statistiquement indépendantes [HASOFER et al. (1974)]. Pour cela, le vecteur aléatoire X est transformé en vecteur aléatoire U , avec :

$$U_i = T(X_i) \quad (3.7)$$

Les variables aléatoires U_i suivent une loi normale centrée réduite et $\forall i \neq j, U_i, U_j$ sont

mutuellement indépendantes. Cette transformation probabiliste T impose de connaître les distributions statistiques de chacune des variables aléatoires. La fonction d'état limite devient après transformation :

$$H(U) = G[X(U)] \quad (3.8)$$

La probabilité de défaillance est alors égale à :

$$P_f = P(G[H(U)] \leq 0) \quad (3.9)$$

soit :

$$P_f = \int_{H(U) \leq 0} \Phi_m(u) du \quad (3.10)$$

Φ_m la fonction de densité de la loi multinormale centrée réduite. L'indice de fiabilité β_{HL} est défini comme étant la distance euclidienne de l'origine de l'espace normal standard à la surface d'état limite $H(u) = 0$ (cf. figure 3.2). u est une réalisation du vecteur aléatoire U , soit :

$$u = ((u_1, u_2, \dots, u_m)^T) \quad (3.11)$$

$H(u)$ est une réalisation de la variable $H(U)$. Il faut donc résoudre le problème de minimisation sous contrainte suivant :

$$\begin{cases} \beta_{HL} = \text{Min} \sqrt{u^T \cdot u} \\ u \in \mathfrak{R}^m, \text{ avec :} \\ H(u) = 0 \end{cases} \quad (3.12)$$

Ce problème est équivalent à [Liu (1991)], [Devictor (1996)] :

$$\begin{cases} \beta_{HL} = \text{Min} f(u) \\ u \in \mathfrak{R}^m, \text{ avec :} \\ H(u) = 0 \end{cases} \quad (3.13)$$

avec

$$f(u) = \frac{1}{2} u^T \cdot u \quad (3.14)$$

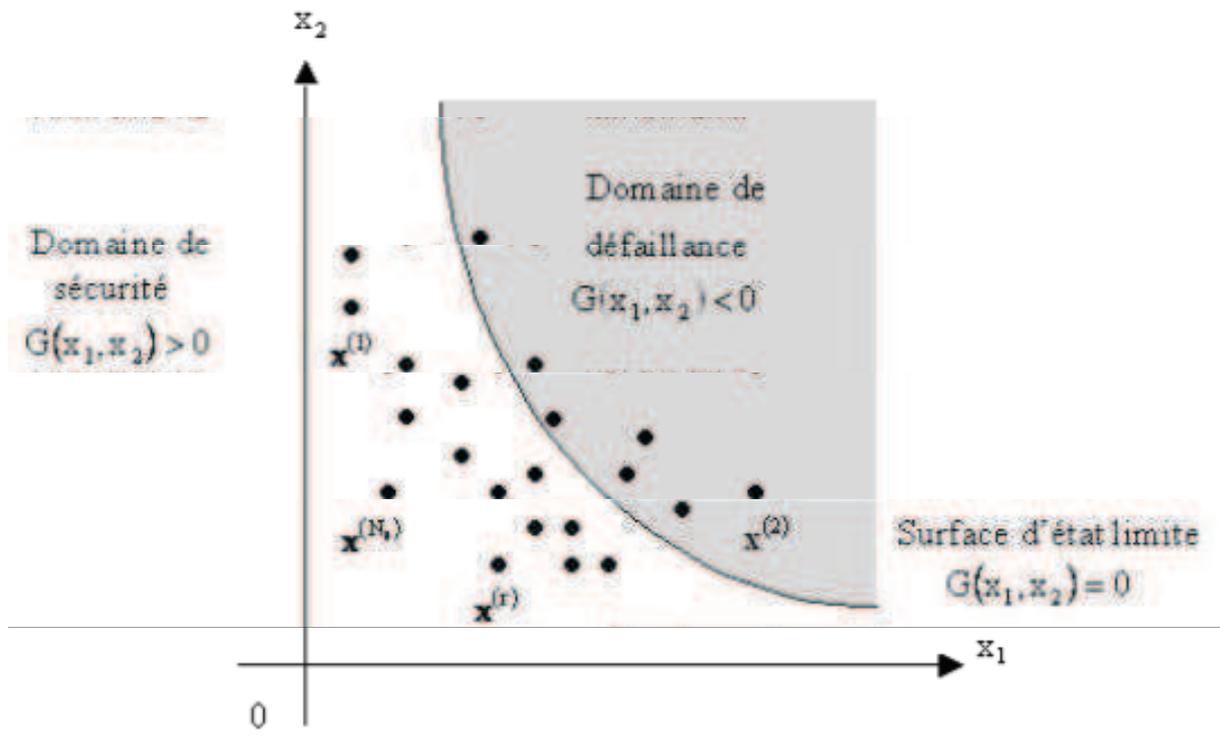


FIGURE 3.2 – Représentation géométrique de β_{HL} pour un problème à deux variables aléatoires.

L'indice de fiabilité est donc le minimum de la fonction f sous la contrainte.

Soient u^* le vecteur solution du problème (voir équation 3.13) et P^* le point de la surface d'état limite tel que $OP^* = u^*$. P^* est appelé point de conception ou point de défaillance le plus probable. Par ailleurs :

$$u^* = -\beta_{HL}\alpha \quad (3.15)$$

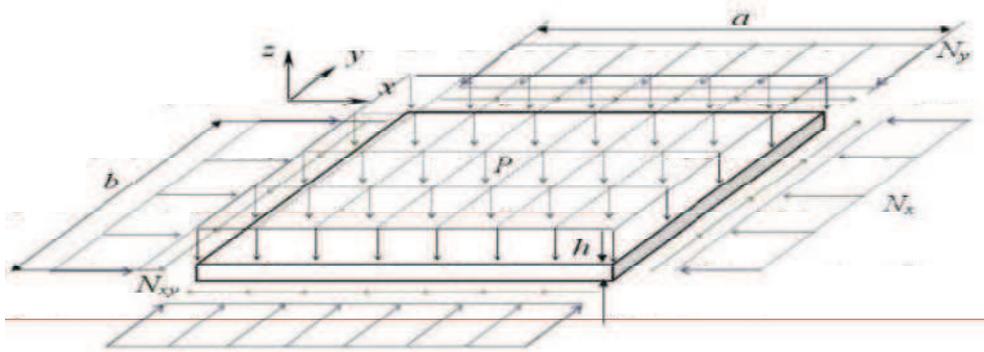
α est le vecteur normal à la surface $H(U) = 0$ au point u^* , soit son gradient normalisé.

L'objectif est alors de relier β_{HL} à la probabilité de défaillance de la structure. Plusieurs méthodes existent et apportent une valeur plus ou moins fine de P_f .

3.3 RÉSULTATS DES APPLICATIONS DE FIABILITÉ DES STRUCTURES

3.3.1 Exemple 1 : Etude dynamique d'une plaque

Nous considérons dans cette partie un exemple simple d'une plaque rectangulaire simplement appuyée sur ses quatre bords [Elhami et al. (2011a)]. Elle est soumise à des charges transversales (cf. figure 3.3).

FIGURE 3.3 – *Plaque soumise à une force.*

Cette application vise à présenter le modélisation numérique par la méthode des éléments finis pour l'évaluation des fonctions d'état limite. Nous utilisons les méthodes d'optimisation heuristiques pour l'estimation de l'indice de fiabilité.

L'analyse de fiabilité présentée ici est basée sur les méthodes FORM, SORM et considère différents scénarios concernant le nombre et la nature de fonctions d'état limite.

Les paramètres de cette modélisation sont illustrés dans le tableau suivant (3.1) :

TABLE 3.1 – *Paramètres statistiques pour les variables de conception.*

	Distribution	Moyenne	Variation(%)
$r(kg/m_3)$	Normal	1200	3
$E(Pa)$	Lognormal	2.410^9	3
$h(m)$	Normal	0.0015	3
$a(m)$	Normal	0.25	3
N_x	Lognormal	202.7	6
$P(Pa)$	Lognormal	100	6

les valeurs initiales des solutions sont choisies comme étant les valeurs moyennes des variables de conception.

On propose d'étudier les différents scénarios de défaillance :

Premier scénario : Le premier scénario est consacré à l'analyse de fiabilité basée sur une seule fonction d'état limite. Elle considère la valeur de la première fréquence propre de la plaque à l'intérieur d'une plage fréquentielle souhaitée. On définit la fonction d'état limite comme suit :

$$G_1(\rho, E, h, a, N_x, P) = 1 - \frac{|f_1 - f_1^m|}{\Delta f} \quad (3.16)$$

où :

– f_1 est la première fréquence propre de la plaque (Hz).

- f_1^m est la valeur souhaitée de la première fréquence propre de la plaque (Hz).
- $\Delta f = 8Hz$ est la plage fréquentielle souhaitée.

Le tableau suivant (3.2) permet de donner les solutions de FORM, SORM.

TABLE 3.2 – Résultats du premier scénario (FORM et SORM)

	FORM	SORM
$r(kg/m_3)$	1160.3	1160.3
E(Pa)	2.50×10^9	2.5×10^9
h(m)	0.0016	0.0016
a(m)	0.24	0.24
N_x	196.40	196.40
P(Pa)	99.82	99.82
N_G	35	64
T(s)	170.52	353.03
b	3.6734	3.6669
Pf	0.012	0.012
Fiabilité	99.99	99.99

Le tableau suivant (3.3) donne MPSO, MPSO-SA et SA successivement.

TABLE 3.3 – Résultats du premier scénario (MPSO-SA, MPSO et SA)

	MPSO-SA	MPSO	SA
$r(kg/m_3)$	1096.6	1193.1	1219.71
E(Pa)	2.32×10^9	2.41×10^9	2.48×10^9
h(m)	0.0015	0.0016	0.0015
a(m)	0.25	0.26	0.26
N_x	187.35	186.25	188.74
P(Pa)	115.13	99.21	98.09
N_G	100	100	100
T(s)	632.3	636.23	237.77
B	4.1352	2.8472	2.4381
Pf	0.002	0.22	0.74
Fiabilité	99.998	99.78	99.26

On remarque que les méthodes d'optimisation heuristiques impliquent des temps de calcul plus élevés que FORM et SORM. Toutefois, pour les méthodes MPSO, MPSO-SA et SA ne nécessite pas le calcul approché du gradient de la fonction d'état limite. Elle peut représenter un avantage déterminant pour certaines applications. En effet selon le chapitre 2 la comparaison entre MPSO, SA et MPSO-SA se confirme dans cet exemple. La méthode MPSO-SA est plus performante en terme de convergence.

Deuxième scénario : La deuxième fonction d'état limite est définie par :

$$G_2(\rho, E, h, a, N_x, P) = 1 - \frac{\delta}{\delta_{lim}} \quad (3.17)$$

Où :

- δ est le déplacement transversal au centre de la plaque ;
- δ_{lim} est le déplacement transversal maximal admis au centre de la plaque.

Différemment de G, cette fonction d'état limite introduit un critère statique que vise à limiter la valeur maximale du déplacement transversal de la plaque.

Troisième scénario : Dans ce scénario, une troisième fonction d'état limite est considérée comme suit :

$$G_3(\rho, E, h, a, N_x, P) = 1 - \frac{|f_2 - f_2^m|}{\Delta f} \quad (3.18)$$

Quatrième scénario : Dans ce scénario, une quatrième fonction d'état limite est incluse, visant à éviter le flambage de la plaque par l'application de la charge de compression dans son plan. Il s'agit, donc, d'un critère lié à la stabilité structurale de la plaque. Cette fonction d'état limite est exprimée sous la forme :

$$G_4(\rho, E, h, a, N_x, P) = 1 - \frac{N_x}{N_x^{crit}} \quad (3.19)$$

N_x est la charge critique de flambage.

Les résultats de l'analyse de fiabilité pour les trois derniers scénarios sont présentés dans le Tableau suivant (3.4) :

TABLE 3.4 – Résultats des 3 scénarios

	Case 2	Case 3	Case 4
$r(kg/m_3)$	1267	1183.1	1195.7
E(Pa)	$2.45 \cdot 10^9$	$2.60 \cdot 10^9$	$2.31 \cdot 10^9$
h(m)	0.0015	0.0015	0.0014
a(m)	0.25	0.25	0.26
N_x	171.69	188.19	192.13
P(Pa)	101.48	104.82	100.33
N_G	100	100	100
T(s)	634.42	633.88	632.83
B	3.4217	3.1377	2.2956
P_f	0.03	0.009	1.08
Fiabilité	99.97	99.91	98.92

Divers applications numériques d'analyse de fiabilité ont été proposées dans cet exemple.

Les différents algorithmes ont été validés (MPSO-SA, SA et MPSO) par rapport aux résultats des méthodes FORM, SORM et Monté carlo.

Une analyse statique et dynamique sous 4 scénarios ont été présentés. Les tests ont montré les avantages associés à notre méthode proposée car elle n'utilise pas d'estimation initiales ni les calculs des gradients. Sa performance est satisfaisante pour la plupart des applications étudiées.

3.3.2 Exemple 2 : Etude dynamique d'un ressort d'embrayage

Le ressort d'embrayage a pour fonction de faire la liaison d'un moteur au mécanisme de transmission. Il est situé entre l'élément moteur et la boîte de vitesse. Il permet de faire un glissement progressif entre ses deux organes ou de réaliser une coupure lors du changement de rapports. Le ressort d'embrayage est modélisé par une coque surbaissée à 12 ailettes [Elhami et al. (2011b)].

L'objectif de cet exemple est de déterminer par un calcul probabiliste les régions critiques de l'embrayage au niveau de ses fréquences de résonance. Dans le domaine de la sûreté de fonctionnement de ce système, nous proposons un indice de fiabilité cible. Nous cherchons deux régions critiques : La première coïncide avec le mode à 3 diamètres de nœuds et la seconde coïncide avec le mode à 4 diamètres de nœuds. Ce calcul va être effectué par la méthode RBDO afin de montrer son efficacité.

Les dimensions initiales R_a , R_i , R_e , H_z et le module de Young E sont supposées aléatoires. Leurs moyennes sont notées respectivement mR_a , mR_i , mR_e et mH_z .

R_a : diamètre intérieur, R_i : diamètre extérieur, R_e : diamètre intérieur des dents et H_z : l'épaisseur.

Les dimensions de cette structure sont les suivantes : $R_a=20.3\text{mm}$, $R_i=26\text{mm}$, $R_e=32.6\text{mm}$ et $H_z=0.9\text{mm}$.

- A_{1a} : le point de conception qui coïncide avec la borne inférieure de la première zone critique.
- B_{1a} : le point qui coïncide avec la borne supérieure de la première zone critique.
- A_{2a} : le point qui coïncide avec la borne inférieure de la seconde zone critique.
- B_{2a} : le point qui coïncide avec la borne supérieure de la seconde zone critique.

Les différents résultats calculés par la méthode hybride fréquentielle sont illustrés dans le tableau 3.5. On peut remarquer aisément que la méthode hybride RBDO-MPSO-SA donne

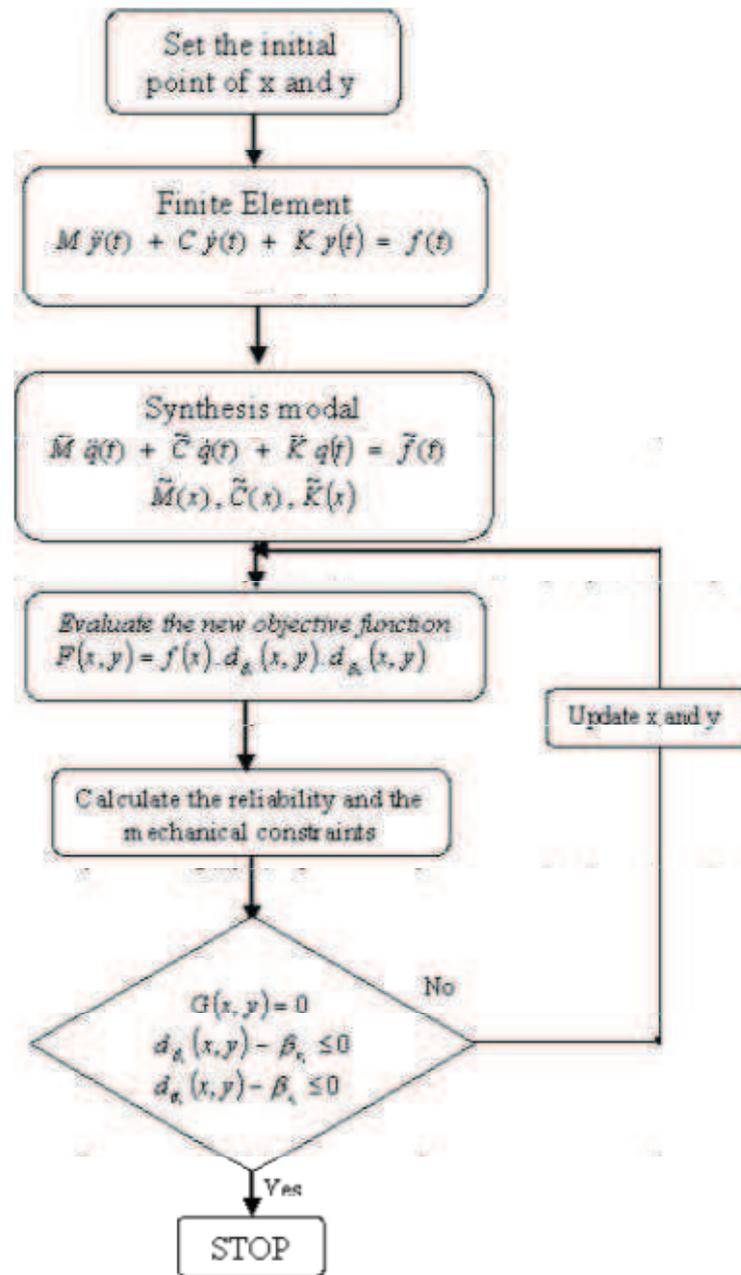


FIGURE 3.4 – Algorithme.

des résultats satisfaisants. Une autre fois, l'approche proposée montre son efficacité et sa robustesse.

Nous avons effectué un calcul probabiliste des régions critiques d'embrayage au niveau de ces fréquences de résonance. Cette région doit garantir un certain niveau de sûreté. Ce dernier est assuré par un indice de fiabilité cible prédéfinie $\beta_c = 3$. Dans cette application, nous avons cherché deux régions critiques : La première coïncide avec le mode à 3 diamètres de nœuds et la seconde coïncide avec le mode à 4 diamètres de nœuds. Ce calcul est effectué par la méthode proposée RBDO-MPSO-SA afin de montrer son efficacité même pour

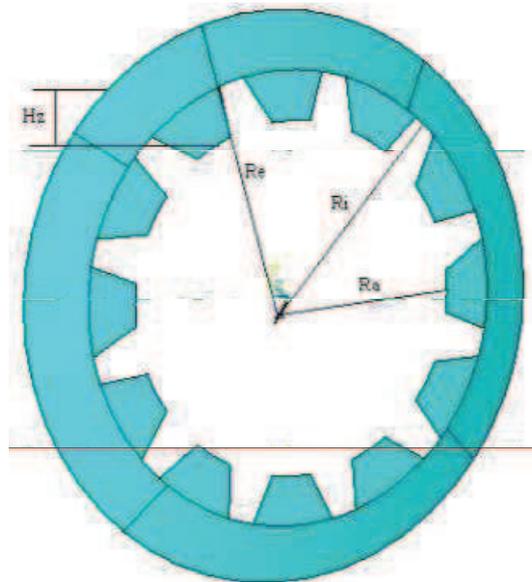


FIGURE 3.5 – Modélisation du ressort d'embrayage.

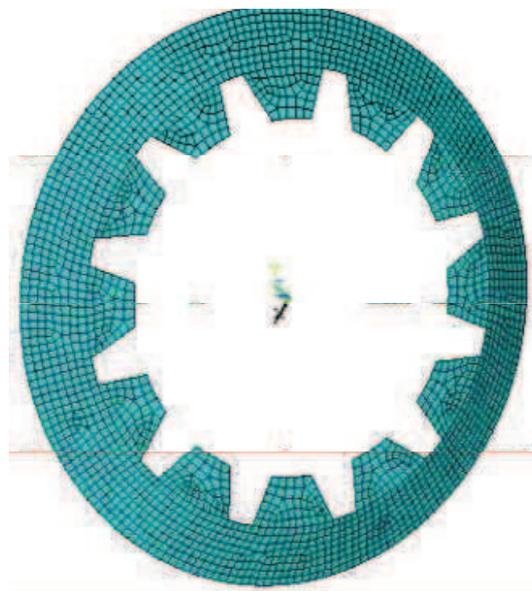


FIGURE 3.6 – Modélisation du ressort d'embrayage par élément fini.



FIGURE 3.7 – le mode à 3 diamètres de nœuds.

les problèmes à plusieurs états limites. On peut remarquer aisément qu'avec la méthode hybride RBDO-MPSO-SA les résultats sont satisfaisants.



FIGURE 3.8 – le mode à 4 diamètres de nœuds.

TABLE 3.5 – Résultats de RBDO-MPSO-SA

Paramètre	Point A1a	Point B1a	Solution optimale	Point A2b	Point B2b
Ra(mm)	20,45	20,523	20,634	20,561	20,642
Ri(mm)	26,306	26,2	26,996	26,397	26,284
Re(mm)	31,81	31,927	31,688	31,805	31,802
Hz(mm)	0,80068	0,79008	0,83286	0,81002	0,78877
b(mm)	3,088	3,03	—	3,1	3,1
Fréquence1(Hz)	4447,8	—	4500,1	4559,2	—
Fréquence2(Hz)	—	5912,7	6000	—	6064

CONCLUSION

Le couplage entre l'optimisation, la fiabilité et notre méthode MPSO-SA a permis une bonne amélioration des résultats par rapport à la méthode RBDO classique. La méthode utilisée consiste à résoudre le problème dans un seul espace hybride regroupant les variables de conception et les variables aléatoires. Ce dernier espace permet un bon contrôle des différents paramètres du problème. Nous avons présenté une nouvelle approche qui consiste à utiliser notre méthode MSPO-SA dans l'approche hybride dynamique RBDO. Avec ces différentes approches hybrides, nous avons proposé un outil d'optimisation efficace pour le contrôle de conception des structures mécaniques. À travers les deux exemples présentés (Plaque vibrante et ressort d'embrayage) dans ce chapitre, nous avons montré l'efficacité de la méthode hybride proposée RBDO-MPSO-SA. Les résultats obtenus sont satisfaisants pour ces deux applications.

Nous présentons dans le chapitre suivant une autre application de la méthode proposée MPSO-SA, l'optimisation de l'interopérabilité des systèmes d'information. Il a comme objectif la résolution du problème de la répartition optimale des efforts pour la mise en œuvre de l'interopérabilité dans une situation de collaboration.

OPTIMISATION DE L'INTEROPÉRABILITÉ DES SYSTÈMES D'INFORMATION

4

SOMMAIRE

INTRODUCTION	74
4.1 CONTEXTE GÉNÉRAL DE L'INTEROPÉRABILITÉ DES SI	75
4.1.1 Barrières et niveaux d'interopérabilité	75
4.1.2 Approches et aspect de l'interopérabilité	76
4.2 MOYENS DE MISE EN ŒUVRE DE L'INTEROPÉRABILITÉ	77
4.2.1 Adoption des standards	77
4.2.2 Adoption d'un unique système intégré	77
4.2.3 Interfaçage pour l'échange de données	78
4.2.4 Interaction orientée service	78
4.2.5 Composition de processus	78
4.3 MODÈLES DE RÉFÉRENCE DE L'INTEROPÉRABILITÉ DES SI	78
4.3.1 Modèle des niveaux d'interopérabilité des SI (LISI)	79
4.3.2 Modèle de maturité organisationnel de l'interopérabilité (OIMM)	80
4.3.3 Modèle de maturité de l'interopérabilité d'entreprise (EIMM)	81
4.3.4 Niveaux de maturité d'interopérabilité dans le gouvernement numérique GIMM	82
4.4 ÉVALUATION DE L'INTEROPÉRABILITÉ	82
4.4.1 Évaluation et potentiel d'interopérabilité	83
4.4.2 Compatibilité à inter opérer et performances opérationnelles	83
4.5 APPROCHE RATIOPI	83
4.5.1 Délimitation de l'étendue de l'étude	84
4.5.2 Quantification du potentiel d'interopérabilité	85
4.5.3 Calcul du degré de compatibilité	86
4.5.4 Évaluation des performances opérationnelles	86
4.5.5 Agrégation du degré d'interopérabilité	87

4.6	OPTIMISATION DE L'INTEROPÉRABILITÉ	88
4.6.1	Modélisation du problème d'interopérabilité	88
4.6.2	Résultat de l'optimisation sur l'interopérabilité	90
4.6.2.1	Interopérabilité de quatre SI	90
4.6.2.2	Interopérabilité de dix SI	92
	CONCLUSION	94

INTRODUCTION

L'innovation dans les entreprises et les organismes est une nécessité pour plus de compétitivité et de modernisation de leur métier. Par conséquent l'ouverture du réseau de l'entreprise vers l'extérieur est une étape importante pour plus de collaboration avec les partenaires (éditeurs, sous-traitants et clients...etc). C'est dans ce contexte que l'interopérabilité des systèmes d'information intervient. Son objectif est de pouvoir interagir et échanger des informations et des fonctionnalités entre systèmes distants (à l'intérieur ou à l'extérieur de l'entreprise).

L'interopérabilité peut être classée selon plusieurs axes. D'abord la notion de barrière, qui empêche l'échange d'information selon certaines conditions. Puis les niveaux, qui interviennent dans la logique d'architecture d'entreprise. Il s'ensuit les notions de l'étendue d'application, le domaine de préoccupation ou le caractère transactionnel de la coopération.

Ce chapitre traite la planification efficace et optimale de la répartition des efforts entre les systèmes d'information, pour les projets complexes de mise en œuvre d'interopérabilité. En effet, la méthode MPSO-SA proposée est utilisée pour optimiser la répartition des efforts entre les différentes activités d'intégration. Cette intégration pouvant faire intervenir plusieurs composants indépendants à travers un réseau de collaboration ayant un but stratégique commun. Les différents modes de simulation développés, ainsi que les résultats obtenus sont donnés à la fin de ce chapitre.

4.1 CONTEXTE GÉNÉRAL DE L'INTEROPÉRABILITÉ DES SI

L'interopérabilité peut être caractérisée selon plusieurs perspectives qui sont illustrées dans la figure 4.1.

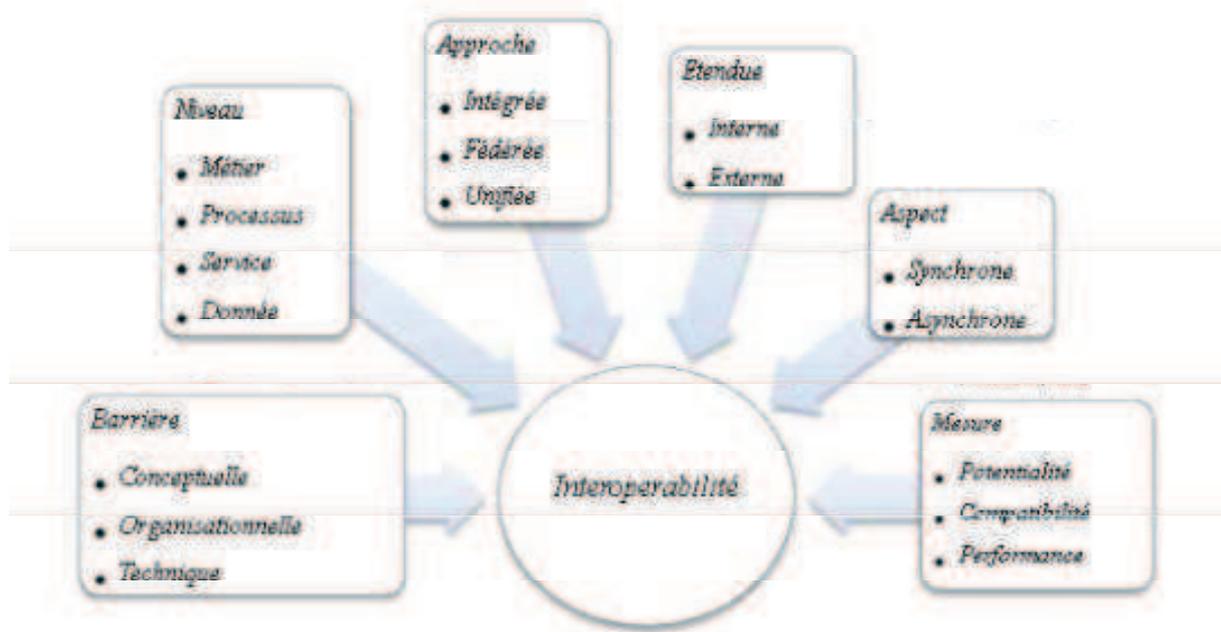


FIGURE 4.1 – Classification de l'interopérabilité.

4.1.1 Barrières et niveaux d'interopérabilité

Les barrières de l'interopérabilité

L'interopérabilité technique s'intéresse à l'homogénéisation des plateformes des systèmes d'information qui échangent des informations suivant des langages et protocoles. La barrière résulte d'une incompatibilité dans ces techniques utilisés empêchant la collaborations entre les systèmes.

L'interopérabilité organisationnelle est attaché au comportement organisationnelle au sein de l'entreprise. La barrière dans cet aspect dépend de l'incompatibilité entre les modes d'organisation des deux structures qui cherche à communiquer.

L'interopérabilité conceptuelle est en relation avec la nature des informations échangées (syntaxiques et sémantiques). L'interopérabilité conceptuelle est possible lorsque les méta-

données sont similaires. Un exemple de métadonnées, le cas qui correspond à introduire des marqueurs dans un langage de programmation spécifique comme XML.

Niveaux d'interopérabilité

Les niveaux d'interopérabilité décrits dans cette section sont, les niveaux métier, processus, services ou données. En effet le système d'information de l'entreprise vise l'automatisation des processus métiers, et essaye de fournir des services de qualité aux différents partenaires. Et cela tout en respectant les différents modèles de données (schémas conceptuels). La collaboration de deux ou plusieurs systèmes pourrait se faire à tous les niveaux selon des approches différentes et des impacts sur les autres couches [Guijarro (2007)].

4.1.2 Approches et aspect de l'interopérabilité

Approches de l'interopérabilité

- L'approche intégrée de l'interopérabilité est le fait d'avoir une coordination entre les différents partenaires en utilisant un modèle dont les composants sont interdépendants. Par conséquent deux systèmes intégrés sont forcément interopérables.
- L'approche unifiée est une conformité à un modèle de donnée qui assure la communication et l'échange d'information entre systèmes.
- L'approche fédérée ne nécessite pas la conformité complète à une spécification ou à un méta-modèle. Dans ce cas les partenaires sont invités à fournir des efforts supplémentaires pour lever les ambiguïtés et faire accélérer l'interopération. Cela consiste en le développement, à la demande, de nouvelles interfaces point à point entre les systèmes appelés à communiquer. Ce qui ne requiert pas un respect d'un format commun entre toutes les parties.

Domaine d'application de l'interopérabilité

Dans le cadre de l'interopérabilité, on peut distinguer deux domaines, un interne et un autre externe. Le besoin de faire collaborer deux processus peut dépasser le cadre interne d'une organisation.

Aspect transactionnel : synchrone ou asynchrone

Il existe deux modes d'interactions entre systèmes. Le mode synchrone est constitué de transactions réparties sur deux ou plusieurs systèmes transactionnels hétérogènes. Quant au mode asynchrone, la partie de l'exécution d'opérations est déportée en différé sur deux ou plusieurs systèmes transactionnels hétérogènes.

4.2 MOYENS DE MISE EN ŒUVRE DE L'INTEROPÉRABILITÉ

Il y a de nombreux moyens pour établir efficacement l'interopérabilité à travers un réseau de collaboration. Cette section présente les plus utilisés d'entre eux. La figure 4.2 résume les principales approches citées de l'interopérabilité en réseau de collaboration.

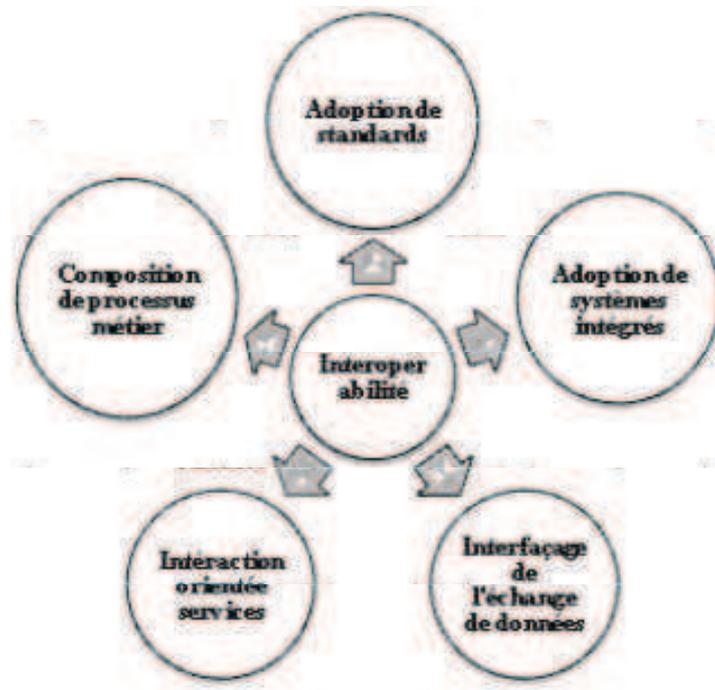


FIGURE 4.2 – Moyens de mise en œuvre de l'interopérabilité .

4.2.1 Adoption des standards

Cette approche unifiée suggère la conformité des systèmes d'information du RdC avec des standards de représentation de données et des protocoles de communication. En conséquence, un nombre considérable de normes ont été élaborées par divers organismes, Couvrant différents domaines. Bien que la normalisation et la standardisation soit importante, elle n'est pourtant pas suffisante pour établir une interopérabilité durable du fait que ces normes changent au fil du temps et il y'a certaines entités qui ne peuvent pas accepter ces normes dans leur traitements internes.

4.2.2 Adoption d'un unique système intégré

Cette approche intégrée se concentre sur l'utilisation des systèmes d'information totalement intégrée. Dans ce genre de configuration, tous les sous-systèmes structurels sont fournis par

un éditeur unique assurant ainsi l'interopérabilité souhaitée entre eux. En termes d'intégration, cette approche demeure la meilleure façon de répondre à un haut degré d'interopérabilité. Cependant, une forte spécialisation et les besoins changeants de chaque domaine requièrent l'utilisation de systèmes différents émanant de plusieurs éditeurs.

4.2.3 Interfaçage pour l'échange de données

Cette approche fédérée consiste en l'élaboration des interfaces point à point personnalisées entre les sous-systèmes d'information. En dépit des aspects négatifs de cette approche telle l'évolution continue des systèmes mis en jeu et les coûts de maintenance des interfaces de collaboration avec de larges réseaux. Cette approche est dans de nombreux cas la voie unique d'échanger des données et d'établir l'interopérabilité.

4.2.4 Interaction orientée service

En plus de l'échange d'information, il y'a un besoin d'intégration des services et la réutilisation des applications. Les systèmes d'information doivent être en mesure non seulement d'accéder et d'utiliser les services fournis par d'autres, mais aussi de réutiliser leurs fonctionnalités. De cette manière, il est théoriquement possible de construire des services d'information complexes à partir de la composition de ceux déjà existants. Cette approche fédérée est utilisée pour établir de nouveaux processus composites orientés services à travers le réseau de collaboration. Il réutilise les services existants au sein des entités en vue de fournir des services métier à forte valeur ajoutée.

4.2.5 Composition de processus

La composition de processus vise à construire de nouveaux processus métier dans un réseau de collaboration. La collaboration au niveau des processus peut être prise en charge par le système reliant les différents sous-systèmes des membre du réseau de collaboration. Cette approche fédérée est utilisé pour établir de nouveaux processus composite orientés services à travers le réseau de collaboration. Un processus métier intégré est automatisé selon des workflows.

4.3 MODÈLES DE RÉFÉRENCE DE L'INTEROPÉRABILITÉ DES SI

Evaluation de l'interopérabilité dans des contextes militaires

Le domaine d'opérations militaires a impulsé la discipline d'évaluation de l'interopérabilité. Ces opérations évoluent dans des contextes de coalition faisant intervenir plusieurs corps d'armées pouvant relever différents armées nécessitent la collaboration sur tous les niveaux, ce qui nécessitent de l'interopérabilité complète entre certains systèmes mis en jeu. Dans ce contexte, il existe une multitude de méthodes visant l'évaluation de cette qualité dans ce contexte opérationnel.

Le modèle du spectre de l'interopérabilité

Le modèle du spectre de l'interopérabilité SoIM (Spectrum of Interoperability Model) propose de combiner quatre niveau techniques (1-4) avec six niveaux de contrôle dits de gestion (1-6) et présente en fin une catégorisation de sept niveaux finaux [LaVean (1980)]. Les sept niveaux proposés par cette classification sont : systèmes séparés, ressources partagées, passerelles, points d'entrées multiples, compatibles à l'interopérabilité, complètement interopérables, et constituants d'un seul système intégré.

Méthode de quantification de l'interopérabilité

La Méthode de quantification de l'interopérabilité QoIM (Quantification of Interoperability Methodology) permet de quantifier sept composants d'interopérabilité (media, langages, standards, exigences de l'environnement, procédures, et les facteurs humains). QoIM demeure l'une des premières approches quantitatives de l'interopérabilité [Mensh et al. (1989)].

4.3.1 Modèle des niveaux d'interopérabilité des SI (LISI)

Le modèle des niveaux d'interopérabilité des systèmes d'information LISI (Levels of Information System Interoperability Model) est la première approche ayant été précisément développée pour mesurer l'interopérabilité des systèmes d'information [DepartmentOf-Defense (1998)] Il a été initié à la demande du Département de la Défense des Etats-Unis d'Amérique dans le but d'améliorer l'interopérabilité des armées sur un théâtre d'opérations. Le modèle LISI est défini selon cinq niveaux de maturité qui doivent permettre l'évaluation du degré potentiel d'interopérabilité entre deux systèmes. Chaque niveau consiste en des caractéristiques liées à l'interopérabilité. Il s'agit donc de comparer les caractéristiques des systèmes qui doivent interopérer à celles des niveaux du modèle LISI. Le modèle LISI

s'inspire du SoIM (Voir Annexe A3). Elle classe les cas étudiés en : isolés, connectés, fonctionnels, domaine, et entreprise tout en ajoutant quatre attributs à chaque niveau. Ces attributs communément connus par PAID sont :

- Procédures ;
- Applications ;
- Infrastructure ;
- Données.

4.3.2 Modèle de maturité organisationnel de l'interopérabilité (OIMM)

Le modèle de maturité organisationnel de l'interopérabilité OIMM (Organizational Interoperability Maturity Model) représente une extension du modèle LISI qui incorpore les aspects organisationnels de l'interopérabilité [Clark et Jones (1999)]. Elle a été utilisée pour la première fois pour la caractérisation des opérations militaires. Cette méthode permet d'identifier les problèmes organisationnels tout en évaluant le degré d'interopérabilité. OIMM consiste en cinq niveaux :

- Indépendant ;
- Coopératif ;
- Collaboratif ;
- Combiné ;
- Unifié.

Le modèle OIMM est qualifié par quatre attributs organisationnels :

- Préparation : Cet attribut décrit la préparation de l'organisation à interopérer. Il est composé de la doctrine, l'expérience et la formation.
- compréhension : Cet attribut permet de mesurer le niveau des communications et le volume de partage des connaissances et informations au sein de l'organisation et la façon dont l'information est utilisée.
- commande et coordination : Ceci est l'attribut qui décrit la gestion et le style de commandement de l'organisation - comment les décisions sont prises et comment les rôles et les responsabilités sont attribuées.
- Ethos : L'attribut ethos caractérise la culture, les valeurs, les buts et les aspirations de l'organisation. Le niveau de confiance au sein de l'organisation est également inclus.

L'OIMM, a été développé pour étendre le modèle LISI au-delà des problèmes technologiques. Son développement est parti du constat que le modèle LISI se focalise sur l'aspect

technique de l'interopérabilité et ne tenait pas compte de l'aspect organisationnel. Cinq niveaux de maturité organisationnelle, sont définis dans le cadre du modèle OIM.

Niveau 0 : Indépendant. Les organisations sont indépendantes. Aucun échange n'est prévu ni anticipé. Les interactions entre les organisations se font uniquement entre des personnes.

Niveau 1 : Ad hoc. A ce niveau d'interopérabilité, seul un cadre organisationnel limité est placé pour des actions ponctuelles entre les organisations.

Niveau 2 : Collaboratif. Un cadre pour supporter l'interopérabilité est en place. Les objectifs, les rôles et les responsabilités sont assignés. Les organisations restent cependant distinctes.

Niveau 3 : Combiné. Le niveau combiné est aussi appelé niveau intégré. Les organisations partagent leurs systèmes de valeurs, leurs objectifs et sont préparées à interopérer.

Niveau 4 : Unifié. Le niveau unifié est celui dans lequel les objectifs, les systèmes de valeurs, les structures de décisions et les connaissances de base sont partagés par les organisations.

4.3.3 Modèle de maturité de l'interopérabilité d'entreprise (EIMM)

Le Modèle de maturité de l'interopérabilité d'entreprise EIMM (Enterprise Interoperability Maturity Model). Son objectif est d'améliorer l'aptitude d'une entreprise à pouvoir interopérer avec d'autres en évaluant la maturité de cette entreprise selon d'une part, les différents niveaux de l'entreprise et, d'autre part des caractéristiques et indicateurs de l'interopérabilité. Le modèle de maturité d'entreprise Enterprise Interoperability Maturity Model - EIMM [Kubicek et Cimander (2007)] est basé sur l'exécution de deux tâches principales. La première tâche est l'établissement de la situation actuelle par rapport à plusieurs niveaux de l'entreprise à l'interne et à l'externe. La deuxième tâche est l'identification de l'état de maturité de chacune des situations sorties de la première étape. Il existe donc six niveaux d'entreprise selon le modèle EIMM, à savoir : (i) La stratégie d'affaires et les processus, (ii) L'organisation et les compétences, (iii) Les produits et les services, (iv) Les systèmes et la technologie, (v) L'environnement légal, la sécurité et la confiance, (vi) La modélisation de l'entreprise. Pour la deuxième tâche, cinq états de maturité de l'interopérabilité sont définis :

- Effectué : la collaboration avec d'autres organisations est effectuée d'une manière chaotique et non planifiée.

- Modelé : la collaboration est réalisée d'une manière similaire à chaque fois et la technique pour se faire s'avère applicable.
- Intégré : la collaboration est documentée formellement, communiquée et s'utilise régulièrement.
- Interopérable : l'organisation est capable de bien gérer des processus interopérables et de s'adapter aux changements de ses partenaires.
- Optimisé : l'organisation est capable de réagir et de s'adapter aux changements de l'industrie d'une manière flexible.

4.3.4 Niveaux de maturité d'interopérabilité dans le gouvernement numérique GIMM

Le modèle conceptuel des niveaux de maturité d'interopérabilité dans le domaine du gouvernement numérique GIMM (Governmental Interoperability Maturity Model) a été développé par [Gottschalk (2009)]. Cinq niveaux ont été identifiés :

- Interopérabilité informatique ;
- Interopérabilité des processus ;
- Interopérabilité des connaissances ;
- Interopérabilité des valeurs ;
- Interopérabilité des objectifs.

Le centre d'intérêt de chaque niveau est différent des autres ce qui engendre une satisfaction des différentes exigences. Il faut noter qu'il existe un léger chevauchement entre ces niveaux, mais ça n'empêche pas la couverture complète des questions pertinentes.

4.4 ÉVALUATION DE L'INTEROPÉRABILITÉ

La mesure de l'interopérabilité est un besoin essentiel pour une amélioration des performances et une évaluation de l'état d'intégration d'un système complexe. Pour se faire, il faut d'abord déterminer les différents aspects caractérisant l'interopérabilité, ensuite il faut connaître les outils adéquats pour une bonne évaluation.

Dans cette section le but est d'étudier les approches de l'évaluation et de mesure de l'interopération et des systèmes d'information. Puis une approche sera retenue pour la suite de ce chapitre. Elle prend en considération les trois aspects fondamentaux, à savoir la potentialité, la compatibilité et les performances opérationnelles.

4.4.1 Évaluation et potentiel d'interopérabilité

L'interopérabilité est une caractéristique d'un système d'information. Cette qualité doit être assurée pendant toute la durée de vie de ce système. Pour la mesurer il faut avoir des mécanismes et métriques qui sont capable de donner une représentation fidèle de l'interopérabilité d'un système.[Ford (2008)]

Plusieurs travaux caractérisant la potentialité à inter opérer essaye de classer le système étudié selon des niveaux prédéfinis selon des modèles de maturité [Elmir et Bounabat (2010)].

TABLE 4.1 – Comparatif de terminologie de modèles de maturité d'interopérabilité

Modèle	Niveau 1	Niveau 2	Niveau 3	Niveau 4	Niveau 5
LISI	Isolé	Connecté	Fonctionnel	Domaine	Entreprise
OIMM	Indépendant	Adaptatif	Collaboratif	Combiné	Unifié
EIMM	Réalisé	Modélisé	Intégré	Interopérable	Optimisé
GIMM	Indépendant	Ad hoc	Collaboratif	Intégré	Unifié

4.4.2 Compatibilité à inter opérer et performances opérationnelles

Deux ou plusieurs systèmes sont compatibles à inter opérer si toutes les barrières, technique, organisationnelle et conceptuelle sont éliminés.

Pour les performances opérationnelles en informatique de deux systèmes qui coopèrent entre eux. Il s'agit dans ce cas de processus de : contrôle interne, monitoring de performance, optimisation, excellence opérationnelle.

4.5 APPROCHE RATIOP

A la lumière des trois aspects fondamentaux, à savoir la potentialité, la compatibilité et les performances opérationnelles exposé dans la section précédente, une approche constituée de 5 étapes pour l'évaluation de l'interopérabilité [?] a été proposée. Ces étapes sont illustrées dans les points suivants :

1. Délimitation de l'étendue de l'étude.

2. Quantification du potentiel d'interopérabilité.
3. Calcul du degré de compatibilité.
4. Évaluation des performances opérationnelles.
5. Agrégation du degré d'interopérabilité.

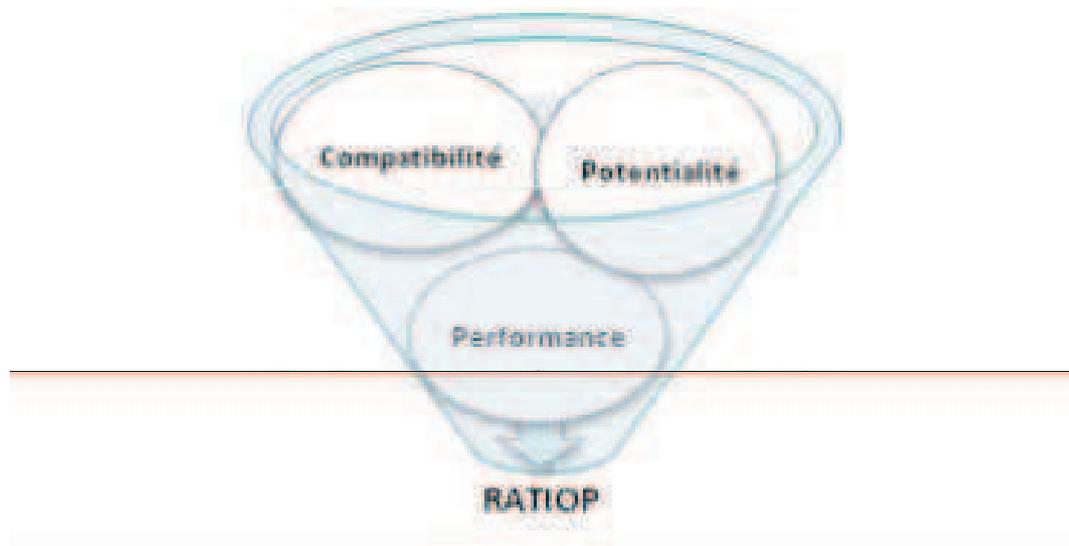


FIGURE 4.3 – Classification de l'interopérabilité.

4.5.1 Délimitation de l'étendue de l'étude

Pour évaluer l'interopérabilité d'un système intégré de services électroniques on passe en premier lieu par la reconnaissance de son écosystème. Sur le plan pratiques, le service a étudié en ligne est basé sur un processus métier. Ce processus macro se compose d'un ensemble de sous processus automatisés entre des entreprises indépendantes. Ces sous-processus sont reliés entre eux par plusieurs interfaces identifiées à l'avance. Dans ce cas, la phase préliminaire vise à identifier le contexte de la prestation de services en ligne a étudié puis énumère les processus sous-jacents automatisées nécessaires à l'interaction. Ainsi, la première étape comprend l'identification :

- Les entreprises impliquées dans la coopération.
- Impliquez-sous-processus au sein de chaque département afin d'étudier la compatibilité.
- Systèmes d'information utilisées pour soutenir l'activité des processus automatisés au sein de chaque entreprise.

4.5.2 Quantification du potentiel d'interopérabilité

Beaucoup de modèles de maturité d'interopérabilité (IMM) a été introduit pour décrire la potentialité d'interopérabilité. Ils sont principalement inspirés par le modèle CMM / CMMI, La liste de quelques modèle connus dans la littérature est donnée comme suit :

- LISI (niveau d'interopérabilité des systèmes d'information) ;
- OIMM (organisation Maturity Model interopérabilité) ;
- EIMM (modèle d'entreprise à l'échéance d'interopérabilité) ;
- GIMM (matrice de maturité gouvernement d'interopérabilité) ;

Chaque modèle adopte un vocabulaire spécifique pour exprimer les niveaux de maturité. Toutefois, les modèles ont habituellement cinq échelles allant de faible à élevé. Une organisation avec un faible niveau d'interopérabilité est caractérisée comme travaillant de manière indépendante ou isolée. Une organisation avec un haut niveau d'interopérabilité est caractérisée comme étant capable de travailler avec d'autres organisations comme une seule entreprise unifiée. Cela permet de maximiser les avantages de la collaboration entre les organisations. Dans notre cas, le calcul du potentiel de l'interopérabilité au sein de l'entreprise exige l'adoption d'un de ces modèles de maturité ci-dessus. L'organisation est classée, puis sur un de ces cinq niveaux noté IMML (pour le niveau de l'interopérabilité du modèle de maturité).

Le mapping du tableau ci-dessous permet l'identification du degré de potentiel de l'interopérabilité

TABLE 4.2 – *Quantification des niveaux de maturité de l'interopérabilité*

Maturity level (IMML)	Potentiality qualification
1	0.2
2	0.4
3	0.6
4	0.8
5	1

Au sein d'une entité, le potentiel est calculé selon l'équation suivante :

$$PI_k = 0.2 * IMML_k \quad (4.1)$$

La potentialité interopérabilité finale est donnée par (voir l'équation suivante)

$$PI = \min(PI_k) \quad (4.2)$$

4.5.3 Calcul du degré de compatibilité

Afin de dématérialiser un processus d'affaires et de l'interconnexion avec son écosystème, il y a une nécessité d'étudier les interfaces externes de ses systèmes de soutien. Dans cette phase, le degré de compatibilité DC est calculé sur la base d'un mapping entre les composantes sous-jacentes et les processus adjacents. Plusieurs études ont porté sur la caractérisation de la compatibilité d'interopérabilité. [Kasunic et Anderson (2004)] Par exemple, identifie plusieurs indicateurs pour décrire cette compatibilité. Pour évaluer le degré de compatibilité, ce chapitre utilise la matrice de compatibilité de [Chen et al. (2008)]. La matrice de compatibilité se compose d'une combinaison des niveaux et des barrières de l'interopérabilité. En notant le degré de compatibilité élémentaires interopération dc_{ij} (i prend des valeurs de 1 .. 4, et j prend les valeurs de 1 .. 6). Par conséquent, si les critères dans une zone marquée satisfaction de la valeur 0 est assignée à dc_{ij} ; autrement, si beaucoup d'incompatibilités sont remplies, la valeur 1 est assignée à dc_{ij} .

Le degré de compatibilité DC est donnée comme suit :

$$DC = 1 - \sum \left(\frac{dc_{ij}}{24} \right) \quad (4.3)$$

4.5.4 Évaluation des performances opérationnelles

La performance opérationnelle (PO) est faite sur la base des indicateurs tels que le score de disponibilité des serveurs d'applications, la qualité de service de communication et le degré de satisfaction des utilisateurs finaux. En notant :

- DS : taux de disponibilité global des serveurs d'application impliqués sur la livraison de l'e-service ;
- QoS : qualité de service des différents réseaux de communication utilisés pour interagir des composants. La QoS est représentée principalement par la disponibilité globale des réseaux et de la bande passante effective au cours d'interopérabilité ;
- TS : niveau de satisfaction des utilisateurs finaux au sujet interopérabilité (basé sur les enquêtes).

Compte tenu de la nature cumulative de ces trois taux, l'évaluation de la performance opérationnelle est donnée par la moyenne géométrique de l'équation suivante 4.4 :

$$PO = \sqrt[3]{(DS * QoS * TS)} \quad (4.4)$$

4.5.5 Agrégation du degré d'interopérabilité

Le taux calculé dans cette phase sert à caractériser l'interopérabilité. Il est le résultat d'une agrégation des trois indicateurs précédents.

En utilisant une fonction f définie comme suit :

$$\begin{cases} f \text{ de } \mathfrak{R}^3 \text{ dans } \mathfrak{R} \\ X \rightarrow f(X) \\ \text{Avec } X = (x_1, x_2, x_3) \end{cases} \quad (4.5)$$

$$x_1 \in [0, 1] \rightarrow PI \quad x_2 \in [0, 1] \rightarrow DC \quad x_3 \in [0, 1] \rightarrow PO$$

$$RatIop = f(PI, DC, PO) \quad (4.6)$$

Étant donné la nature indépendante de ces trois indicateurs, la moyenne arithmétique est la solution la plus triviale, elle est donnée comme suit :

$$RatIop = (PI + DC + PO)/3 \quad (4.7)$$

Si le réseau métier dispose d'éléments clés pour pondérer les trois composants en introduisant des notions de poids (w_1, w_2, w_3). Le calcul du RatIop sera donné en utilisant la moyenne arithmétique pondérée.

$$RatIop = (w_1 * PI + w_2 * DC + w_3 * PO)/(w_1 + w_2 + w_3) \quad (4.8)$$

4.6 OPTIMISATION DE L'INTEROPÉRABILITÉ

4.6.1 Modélisation du problème d'interopérabilité

L'optimisation de la mise en œuvre de l'interopérabilité est une condition clé pour développer et faire évoluer efficacement la collaboration intra et inter organisationnelle. Cette section présente l'approche de modélisation pour la représentation de l'évolution d'interopérabilité des systèmes d'information en utilisant la notion de RatIop.

Pour augmenter l'efficacité de l'interopérabilité, l'algorithme d'optimisation MPSO-SA est utilisé comme méthode d'optimisation heuristique afin de trouver la meilleure répartition de l'effort nécessaire dans une situation de travail en réseau. Le problème de la répartition optimale des efforts est un problème d'optimisation pour la mise en œuvre de l'interopérabilité dans une situation de collaboration.

En plus de la fonction objectif qui limite l'effort maximal à fournir dans le multi projet de mise en œuvre, on prend en considération les équations de contrôle optimal de l'interopérabilité de chacun des sous systèmes étudiés, on est donc devant un problème d'optimisation avec contraintes d'égalités. Si on prend de plus les limites minima et maxima. des efforts mis en jeu, on est alors devant un problème d'optimisation avec contraintes d'égalités et d'inégalités.

La mise en œuvre de l'interopérabilité au sein d'un réseau de collaboration et à travers des organismes indépendants peut être considéré comme un multi-projet qui vise un objectif de collaboration unifiée en faisant impliquer des équipes différentes dans le but d'interconnecter les systèmes d'information indépendants. Dans un tel environnement, le défi considéré est l'affectation des ressources et la répartition d'efforts afin d'établir efficacement l'interopérabilité à un niveau projeté.

Dans cette section, l'objectif est d'obtenir une répartition optimale des efforts en vue d'établir une situation de collaboration spécifique.

RatIop est une approche d'évaluation centrée sur chacun des sous systèmes qui vise à

quantifier le degré d'interopérabilité d'un SI au sein de son écosystème.

RatIop stipule que l'amélioration de l'interopérabilité d'un SI est obtenue par la mise en œuvre des changements dans ces systèmes de support et également apporter des améliorations sur les interfaces externes des systèmes avec lesquels il est interconnecté.

Dans ce qui suit, on propose de coupler la méthode RatIop avec de la modélisation linéaire. Ce couplage tente de caractériser l'évolution du degré d'interopérabilité globale d'un ensemble de SI interconnectés. Il s'appuie sur la surveillance des efforts nécessaires pour améliorer le degré d'interopérabilité du réseau de collaboration.

Pour illustrer cela, prenons un ensemble de n systèmes (S_1, S_2, \dots, S_n) . Nous supposons que :

- Chaque système gère le SI d'une seule organisation.
- Chaque SI est automatisé et soutenu par exactement une infrastructure informatique.
- Chaque système peut interagir avec tous les autres systèmes.
- Les systèmes informatiques ont la capacité d'interopérer de façon homogène avec l'environnement.

Nous associons à chaque système S_i un ratio $a_i = RatIop(S_i)$ représentant le degré de l'interopérabilité au sein de son écosystème ; Nous avons pour objectif de suivre l'évolution de cet indicateur de façon macroscopique. Le vecteur (a_1, \dots, a_n) évolue en conformité avec les efforts déployés pour adapter le système interconnectés de l'état actuel vers l'état cible en termes de vision de l'architecture d'entreprise. En notant :

$I = (a_i)$ représentant le vecteur courant de l'interopérabilité.

$I' = (a'_i)$ représentant le vecteur cible de l'interopérabilité.

Pour chaque système :

$$a'_i = \sum_j E_{ij} * a_j \quad (4.9)$$

E_{ij} représente l'effort à fournir au niveau du S_i en vue d'améliorer le niveau d'interopérabilité du système S_j .

$E = (E_{ij})$ la matrice d'effort à fournir en vue d'atteindre le niveau cible d'interopérabilité.

$$I' = EI$$

$$\begin{pmatrix} E_{11} & E_{12} & \dots & E_{1n} \\ E_{21} & E_{22} & \dots & E_{2n} \\ \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots \\ E_{m1} & E_{m2} & \dots & E_{mn} \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ \dots \\ \dots \\ a_n \end{pmatrix} = \begin{pmatrix} a'_1 \\ a'_2 \\ \dots \\ \dots \\ a'_n \end{pmatrix}$$

En supposant que tous les SI étudiés sont compatibles les uns avec les autres et qu'il n'existe pas de barrières empêchant l'interaction explicite, l'effort E_{ij} est équivalent dans ce cas au ratio de la charge de travail N_{ij} (charge de travail allouée à l'amélioration des interfaces externes de S_i pour faciliter le degré RatIop du S_j) sur l'ensemble du charge de travail allouée à l'amélioration de l'interopérabilité.

Dans ce cas, l'objectif tactique recherché est de trouver l'effort optimal en vue d'atteindre le vecteur cible d'interopérabilité.

Les fonctions objectifs à minimiser respectivement pour chaque système S_i sont présentées dans les équations suivantes :

$$a'_i - \sum_i \sum_j E_{ij} * a_i \leq 0 \quad (4.10)$$

$$\sum_j E_{ij} \leq 100\% \quad (4.11)$$

E_{ij} est à multiplier avec la valeur $N_i/N_{overall}$ sachant que

$$N_i = \sum_j N_{ij} \quad (4.12)$$

4.6.2 Résultat de l'optimisation sur l'interopérabilité

4.6.2.1 Interopérabilité de quatre SI

L'approche proposée MPSO-SA dans le chapitre 2 est utilisée pour optimiser le multi projet de mise en œuvre de l'interopérabilité. Elle proposera un effort optimal théorique nécessaire pour atteindre la fonction objectif. Cette distribution d'effort est à comparer avec les propositions des architectes d'intégration concernant l'estimation de l'effort nécessaire

à l'interfaçage et l'interconnexion des systèmes impliqués.

Pour bien illustrer l'application de la méthode d'optimisation dont les couples RatIop et MPSO-SA interviennent, nous prenons le cas de quatre systèmes d'information qui interagissent au sein d'une organisation (voir ??). Ces systèmes sont les suivants :

- Ressources humaines (RH)
- La gestion comptable du système (AC)
- Supply Chain Management System (CS)
- Vérification de la gestion et de contrôle (CM).

Après une évaluation initiale du degré d'interopérabilité de chaque système cité, nous remarquons qu'ils ont atteint respectivement des valeurs de : 0,7, 0,3, 0,8 et 0,2.

$$\begin{pmatrix} E_{11} & E_{12} & E_{13} & E_{14} \\ E_{21} & E_{22} & E_{23} & E_{24} \\ E_{31} & E_{32} & E_{33} & E_{34} \\ E_{41} & E_{42} & E_{43} & E_{44} \end{pmatrix} \begin{pmatrix} 70\% \\ 30\% \\ 80\% \\ 20\% \end{pmatrix} = \begin{pmatrix} 80\% \\ 45\% \\ 90\% \\ 35\% \end{pmatrix}$$

Les Acteurs des systèmes d'information (en conformité avec les équipes métiers) ciblent une amélioration du degré d'interopérabilité de cette situation de collaboration pour atteindre le vecteur 0,8, 0,45, 0,9 et 0,35 voir figure.

Dans ce cas, nous appliquons l'algorithme MPSO-SA pour trouver la matrice optimale de l'effort qui minimisent la fonction objective et respecter les contraintes dans les équations 4.10 et 4.11.

Après application de l'algorithme PSO pour optimiser le système, la matrice optimum d'effort minimisant la fonction objectif et qui respecte les contraintes est représentée dans de l'interface JAVA sur la figure 4.4, elle est donnée comme suit.

$$\begin{pmatrix} 27,5\% & 2,4\% & 69,7\% & 0\% \\ 30,6\% & 0\% & 68,5\% & 0\% \\ 32,6\% & 16,2\% & 65,1\% & 1,1\% \\ 28,7\% & 4,5\% & 66,7\% & 0\% \end{pmatrix}$$

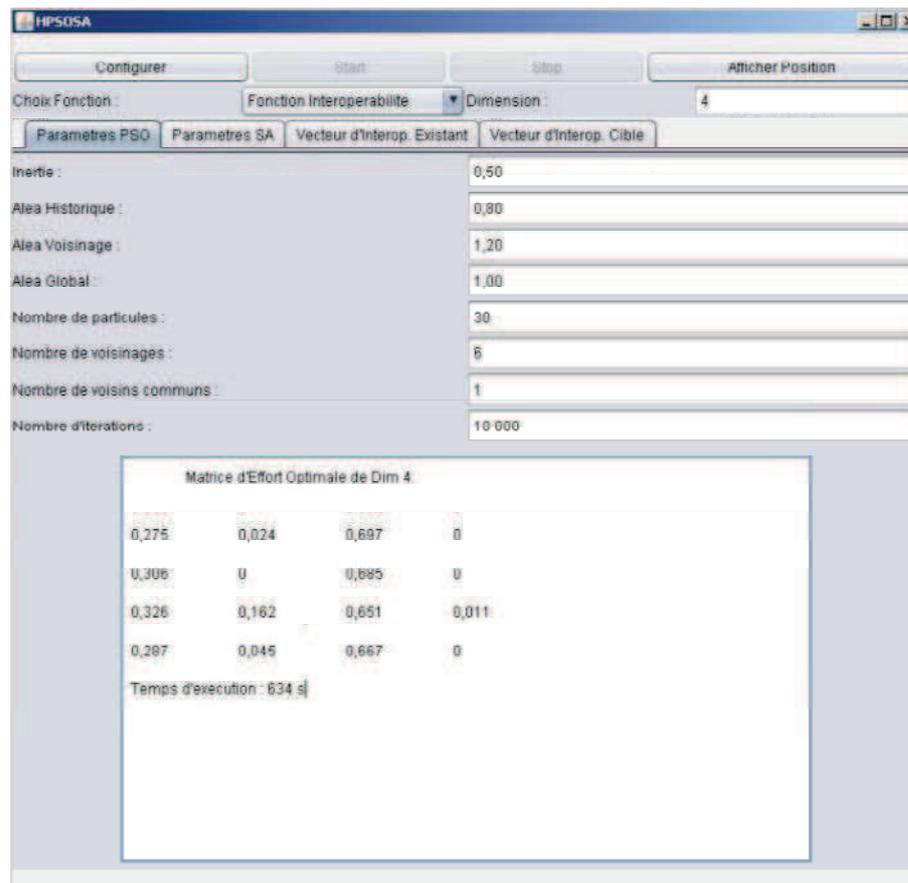


FIGURE 4.4 – Résultats de la matrice d'effort à 4 dimensions.

4.6.2.2 Interopérabilité de dix SI

Dans cette section, la taille et la complexité du problème est augmenté en utilisant dix systèmes d'information S_1, S_2, \dots, S_{10} qui interagissent entre eux voir la figure 4.5. Ce qui rend le calcul de la matrice d'effort optimal quasiment impossible d'une manière manuel. Ils existe 100 termes dans la matrice qui varient en parallèle durant le processus d'optimisation. Le challenge est de trouver la matrice optimale en respectant l'espace de recherche ainsi que les contraintes du problème.

La taille du vecteur courant et du vecteur cible d'interopérabilité est de 10. La taille de la matrice d'effort recherchée est de 10x10. Une première évaluation par RatIop pour les dix processus donne les degrés d'interopérabilité présentés dans la figure 4.6.

Après application de l'algorithme MPSO-SA pour optimiser le système, la matrice optimum d'effort minimisant la fonction objectif et qui respecte les contraintes est représentée dans la figure 4.7

Le problème d'optimisation de la matrice d'effort suppose que tous les systèmes d'information interagissent les uns avec les autres. Il n'y a pas d'hypothèse de simplification comme

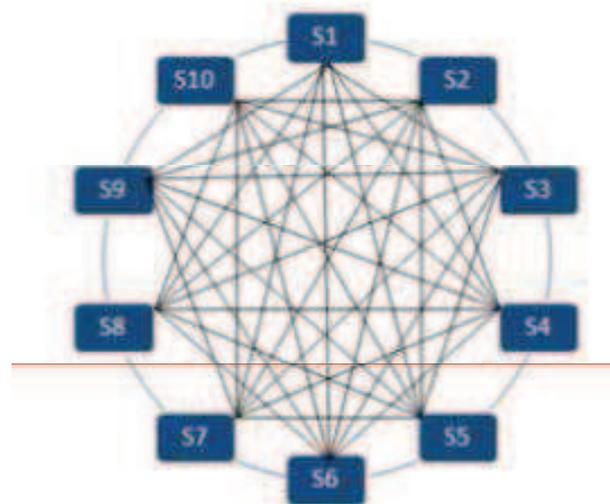


FIGURE 4.5 – Interaction de dix systèmes d'information.



FIGURE 4.6 – Vecteur courant et vecteur cible pour un problème de 10 dimensions.

17,33%	7,03%	29,28%	15,32%	11,73%	1,37%	2,46%	2,54%	9,98%	2,95%
5,41%	5,06%	26,64%	8,08%	9,75%	3,57%	12,54%	12,95%	15,34%	0,66%
1,90%	3,26%	9,65%	10,88%	17,62%	6,25%	12,74%	27,85%	7,44%	2,21%
6,35%	7,77%	22,22%	7,01%	13,48%	8,10%	9,93%	8,72%	16,25%	0,17%
4,45%	2,75%	21,93%	6,44%	39,96%	6,41%	0,33%	3,69%	9,07%	4,96%
2,97%	12,25%	23,12%	4,79%	20,09%	8,17%	3,06%	9,27%	15,74%	0,56%
5,03%	2,55%	20,93%	21,39%	7,38%	15,47%	6,98%	4,98%	13,00%	2,29%
7,59%	2,88%	18,61%	12,42%	18,73%	4,00%	7,70%	14,27%	10,57%	3,23%
8,33%	4,80%	20,22%	9,59%	17,37%	2,75%	11,49%	7,47%	14,81%	3,18%
4,75%	4,29%	32,51%	8,38%	12,95%	11,72%	3,12%	15,74%	3,55%	2,99%

FIGURE 4.7 – Matrice d'effort optimisée en respectant les contraintes.

par exemple, supposée que le système S_2 ne communique pas avec le système S_5 . Ceci diminue le nombre de variables dans la matrice d'effort.

Analyse des résultats

- Le résultat de l'optimisation de la matrice d'effort permet aux architectes des systèmes d'information d'avoir une visibilité sur la solution optimale possible dans le domaine des solutions.
- Pour avoir une bonne gouvernance des systèmes d'information, il faut avoir une solution futur qui convergence vers une solution théorique.
- Pour améliorer les résultats, en modifiant le vecteur cible une nouvelle solution (matrice d'effort) peut être générer automatiquement. Le système d'optimisation est parfaitement paramétrable pour des éventuelles changements.
- Lorsque la dimension des vecteurs courant et cible d'interopérabilité devienne grande ≥ 4 la prévision de la matrice d'effort devient laborieuse voir impossible d'une manière manuelle. Un système d'optimisation devient primordiale.

CONCLUSION

Ce chapitre a traité l'interopérabilité comme une caractéristique de qualité de l'interaction entre les systèmes d'information intra et inter organisations. La méthode MPSO-SA proposée dans le chapitre 2 optimise la répartition des efforts (Matrice des efforts) dans une situation de collaboration des SI. Nous avons utilisé un ratio métrique pour mesurer cette qualité en prenant en compte les trois principaux aspects opérationnels : la potentialité, la compatibilité et le suivi des performances.

La méthode MPSO-SA proposée est utilisée pour l'optimisation du multi projet de mise en œuvre de l'interopérabilité (Interopérabilité de quatre et dix SI). Elle propose un effort optimal théorique nécessaire pour minimiser la fonction objectif. Pour une bonne exploitation de ces résultats, la comparaison de la distribution d'effort avec les propositions des architectes d'intégration est essentielle. Cela permet une bonne estimation de l'effort nécessaire à l'interfaçage et l'interconnexion des systèmes d'information impliqués.

Le chapitre suivant présente les différents concepts de la technologie Peer to Peer ainsi que la mise en place d'un framework. Pour augmenter les performances, une application java basée sur une architecture Peer to Peer permettra la distribution du calcul de l'algorithme MPSO-SA.

DISTRIBUTION PAIR A PAIR DU CALCUL D'OPTIMISATION

5

SOMMAIRE

INTRODUCTION	96
5.1 CONCEPT DE RÉSEAU PAIR A PAIR ET PLATEFORME JXTA	97
5.1.1 Architecture Client-Serveur	97
5.1.2 Architecture pair à pair	97
5.1.3 La technologie JXTA et JXSE	98
5.1.3.1 La technologie JXTA-JXSE	98
5.1.3.2 Protocoles JXTA	99
5.1.3.3 Architecture JXTA	101
5.1.3.4 Services fournis par l'architecture JXTA	103
5.2 DISTRIBUTION DU CALCUL D'OPTIMISATION	103
5.2.1 Modélisation	103
5.2.2 Stratégie de découpage de l'application	104
5.2.3 Interface utilisateur	105
5.2.4 Implémentation	105
5.3 PREMIERES EXPERIMENTATIONS	106
5.3.1 L'approche BBP, pour calculer Π	106
5.3.2 La distribution de l'algorithme MPSO-SA	107
CONCLUSION	108

INTRODUCTION

Le nombre de réseaux informatiques interconnectés sur Internet est en constante augmentation, ce qui provoque des problèmes de congestion des réseaux (manque de bande passante) et des coûts faramineux d'entretien. L'apparition des réseaux pair à pair (en anglais Peer-to-Peer) a permis d'atténuer ces phénomènes courants et inhérents aux réseaux informatiques.

Le réseau pair à pair introduit le concept de la relation d'égal à égal entre deux ordinateurs. Il offre de nombreuses possibilités, entre autres, le partage de fichiers et le calcul distribué. Dans ce chapitre le Framework de développement JXTA (développé par SUN) sera utilisé pour bâtir une architecture de calcul distribué. Nous présentons, d'une part, les différents concepts de la technologie pair à pair, son mode de fonctionnement, puis sa comparaison avec les réseaux informatiques traditionnels. Puis nous montrons la mise en place d'un framework avec une application java basée sur une architecture pair à pair permettant le calcul distribué. L'exemple utilisé pour tester la plateforme est celui du calcul de π en utilisant la formule connue sous le nom de BBP (Bailey-Borwein-Plouffe). La particularité de cet exemple est que le calcul des chiffres après la virgule de π peut se faire d'une manière indépendante. Ainsi le calcul peut être partagé sur plusieurs pairs autonomes.

Les traitements et les calculs distribués représentent un domaine prometteur dans la résolution des problèmes d'optimisation. Ce chapitre est une perspective de ces travaux avec une évolution de la méthode MPSO-SA promettant une amélioration du temps de calcul par la distribution sur un réseau pair à pair.

Nous proposons de construire une plateforme dédiée à l'optimisation par le calcul distribué. Nous nous inspirons pour ce travail de recherche des travaux menés dans [CABANI (2007)]. Après la présentation du concept de réseau pair à pair, nos premières expérimentations iront au calcul distribué de Pi et à la recherche de la distribution de l'algorithme MPSO-SA comme présenté ci-dessus.

5.1 CONCEPT DE RÉSEAU PAIR A PAIR ET PLATEFORME JXTA

Un réseau informatique implique l'interconnexion de deux ou plusieurs ordinateurs. Cette section définit les deux architectures client-serveur et pair à pair.

5.1.1 Architecture Client-Serveur

L'architecture Client-Serveur est une architecture réseau classique très répandue dans de nombreux organismes. Le traitement de la demande est divisé entre les postes clients et serveurs. Il est essentiel que tous les ordinateurs clients soient connectés au serveur de façon indépendante, de manière à fournir un service pour chaque client. Généralement, le poste client s'occupe de la présentation des informations, tandis que le serveur s'occupe de l'appliquatif et de l'accès aux bases de données.

L'architecture client-serveur est appelé aussi architecture deux tiers. Bien que cette architecture présente de nombreux avantages, y compris la flexibilité et la maintenabilité, il implique de nombreux inconvénients, notamment, la perte en bande passante et l'indisponibilité du réseau tout entier si le serveur tombe en panne.

5.1.2 Architecture pair à pair

Actuellement, l'architecture réseau pair à pair est très répandue. La différence entre les architectures client-serveur et pair à pair est que dans un réseau pair à pair, les composants principaux, appelés Pairs, peuvent agir simultanément en tant que serveur et client.

La caractéristique principale du pair à pair est que chaque ordinateur peut avoir à la fois un module serveur et un module client. Cela permet non seulement aux ordinateurs d'avoir accès à des modules logiciels présents sur d'autres ordinateurs, mais aussi de fournir des services pour les autres ordinateurs du réseau. Le but de cette architecture est de permettre à tous les clients de fournir des ressources comme la bande passante, l'espace de stockage, et la puissance de calcul. Ce dernier aspect est essentiel pour l'établissement d'une plateforme de calcul distribué basée sur une architecture pair à pair.

Dans un réseau de type pair à pair le nombre de nœuds peut augmenter de façon aléatoire, ce qui engendre une augmentation de la capacité totale du système. Par contre, dans l'architecture client-serveur, l'ajout de plus de clients se traduirait par une baisse du débit de transfert de données sur le réseau.

Parmi les avantages des réseaux pair à pair, il y a la robustesse aux défaillances d'un ou

plusieurs nœuds. Cela est assurée par la réplication de données sur plusieurs pairs. Lorsqu'un pair hébergeant une ressource tombe en panne ou se déconnecte, cela ne pose pas de problème car la donnée en question est récupérée sur une autre machine du serveur.

Avantage des réseaux pair à pair

- ✓ Coût de déploiement ;
- ✓ Robustesse du réseau ;
- ✓ Décentralisation des ressources.

Inconvénient des réseaux pair à pair

- ✓ Complexité de conception
- ✓ Latence
- ✓ Problèmes de sécurité
- ✓ Décentralisation des ressources

5.1.3 La technologie JXTA et JXSE

La plate forme JXTA (Juxtapose) est très connue pour le développement des applications pair à pair. JXTA représente un langage de programmation libre (open source) et un protocole pair à pair. Indépendamment de la plateforme de développement, elle a été développée par Sun Microsystems en 2001.

Dans JXTA, les périphériques connectés au réseau peuvent échanger, communiquer et travailler en collaboration d'une manière indépendante de la topologie du réseau sous-jacent. Le langage essentiel de la plate forme est XML, mais d'autres langages de programmation peuvent être utilisés à savoir, JAVA, le C, C++, le C#,...etc.

Le protocole JXTA permet aux pairs de créer un réseau virtuel sur lequel chaque pair peut interagir et utiliser les ressources des autres pairs. Cette interaction peut se faire directement ou même lorsque le pair se trouve sur un autre réseau physique ou derrière un FireWall.

5.1.3.1 La technologie JXTA-JXSE

JXTA est un framework Open Source pair à pair développé sur le modèle open source Apache par Sun Microsystems. " La technologie JXTA est un ensemble de normes ouvertes,

généralisant les protocoles pair à pair et permettant à tout appareil connectable (téléphone mobile, smartphone, PC, serveur) à un réseau de communiquer et de collaborer " [Sun Microsystems, 2001]

Les objectifs de base de JXTA s'expriment en 6 points :

- ✓ les pairs se découvrent mutuellement sans serveur central ;
- ✓ les pairs peuvent rechercher, créer et joindre un groupe spécifique ;
- ✓ les pairs peuvent se signaler sur le réseau et découvrir les ressources qui y sont disponibles ;
- ✓ les pairs sont en mesure de se surveiller les uns les autres ;
- ✓ les pairs sont en mesure de communiquer les uns avec les autres ;
- ✓ les pairs n'ont pas à se soucier de la topologie du réseau pour effectuer une quelconque transaction.

Comme mentionné, JXTA ne nécessite pas l'utilisation d'un langage spécifique ou même d'un système d'exploitation en particulier. Il est complètement libre de contrainte de ce type.

En outre les protocoles constitutifs de JXTA sont conçus de façon à ne nécessiter aucune méthode d'authentification, de méthode de sécurité ou de modèle de chiffrement particulier. JXTA est conçu pour accueillir tout type de service. Ses protocoles peuvent également être mis en collaboration avec d'autres protocoles tels que : TCP-IP, http et Bluetooth. Le développement et la mise en place d'une application avec JXTA peut se faire avec plusieurs langages de programmation, pour rappel, le C/C++, le JAVA, le PYTHON et PERL. En outre JXTA s'utilise sur quasiment tous les périphériques technologiques et cela quelle que soit leur couche logicielle.

5.1.3.2 Protocoles JXTA

La version utilisée de JXTA dans ce rapport de thèse est (v2.6). Elle est constituée de six protocoles essentiels, chacun d'entre eux joue un rôle particulier dans le remplissage des objectifs de la charte de JXTA.

Ces protocoles sont conçus pour les services de manutention JXTA. Ils s'occupent chacun de la gestion de services spécifiques. Ils constituent la couche de base de JXTA et sont :

- Les protocoles Peer Resolver (PRP)
- Les protocoles Peer Discovery (PDP)
- Les protocoles Peer Information Protocol (PIP)
- Les protocoles Peer Pipe Binding (PBP)
- Les protocoles Peer End point Protocol (PEP)
- Les protocoles Rendez-Vous Protocol (RVP)

Le Peer Resolver Protocol (PRP)

Ce protocole est utilisé pour envoyer une requête à un ou plusieurs pairs et de recevoir leur réponse. Les requêtes peuvent être envoyées à tous les pairs ou à un pair spécifique au sein du groupe.

Le Peer Discovery Protocol (PDP)

Grâce à ce protocole, tout composant JXTA peut signaler sa présence sur le réseau. Les pairs peuvent publier leurs ressources et signaler leur présence sur le réseau, tout en ayant la possibilité de découvrir les autres pairs connectés ainsi que les ressources hébergées et publiées par ceux-ci. Hormis les pairs, tout composant d'un réseau JXTA peut également signaler sa présence et son existence (Pipes, Pairs, groupe de pair, etc). Les signalements des pairs sont représentés sous forme de documents XML qui ont une structure de métadonnées décrivant les ressources réseau du pair.

Le Peer Information Protocol (PIP)

Ce protocole permet d'obtenir des informations d'état sur les pairs telles que la disponibilité, la charge de trafic, les capacités et d'autres informations.

Le Pipe Binding Protocol (PBP)

Grâce à ce protocole, les pairs peuvent établir une communication ou un pipe avec un ou plusieurs pairs.

Le Peer Endpoint Protocol (PEP)

Le Endpoint protocol est responsable de la découverte de route par les pairs. Grâce à lui les pairs peuvent découvrir une route ou des séquences entre eux, qu'ils utilisent pour s'envoyer des messages. Par exemple, si les pairs veulent s'envoyer des messages et qu'il n'y a pas de route directe entre eux, alors les pairs ont besoin de trouver des pairs intermédiaire(s) qui permettront de faire le lien en jouant le rôle de passerelle pour leurs messages. Le Endpoint Protocol est également connu comme Endpoint Routing Protocol (ERP) et est utilisé pour détecter les informations de route entre pairs. Par conséquent, si la topologie du réseau venait à changer ou encore qu'une portion en devenait indisponible, alors ERP peut être

utilisé pour déterminer un itinéraire alternatif. On peut voir que lorsqu'il n'y a pas de route directe entre pairs "A" et "B", un autre itinéraire passant par "C" sera utilisé pour envoyer des messages via les pairs.

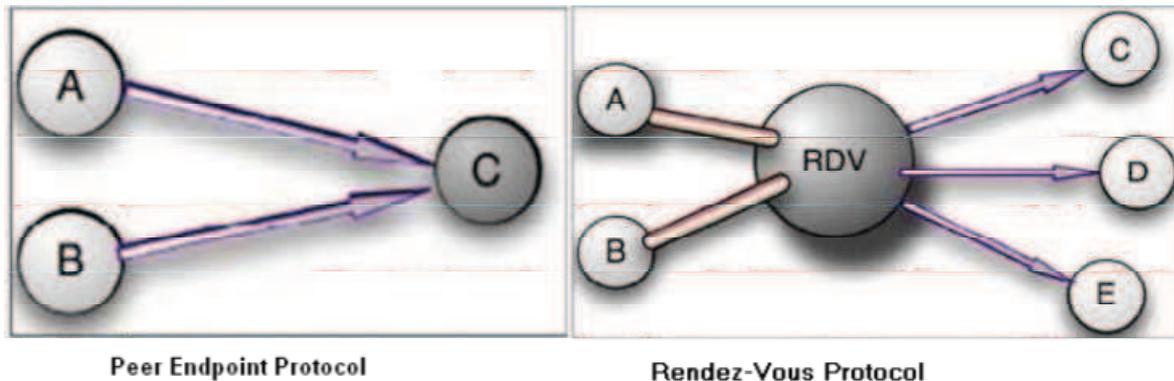


FIGURE 5.1 – *Peer Endpoint et Rendez-Vous Protocol.*

Rendez-Vous Protocol (RVP) Le Rendez-Vous protocol est utilisé pour propager des services par le biais de groupe de pairs. Dans le groupe de pairs, les pairs peuvent être soit des pairs Rendez-vous (Rendez-vous Peer) ou des pairs qui sont à l'écoute de rendez-vous. Ce protocole permet aux pairs d'envoyer les messages à toutes les instances en écoute des services. Les pairs Rendez-vous peuvent être utilisés par les Peer Resolver et Pipe Binding protocoles pour propager des messages à travers le groupe de pairs. Toutefois, il n'existe aucune garantie que les messages propagés par les pairs Rendez-vous atteigne toujours tous les pairs dans le groupe. La figure 5.1 est une illustration de Rendez-vous par les pairs et la propagation message à travers le groupe de pairs.

5.1.3.3 Architecture JXTA

Le protocole JXTA [Gradecki (2002)] est subdivisé en trois couches intrinsèquement liées les unes aux autres. Ces couches sont :

La couche de base ou Core Layer

La couche de services ou Service Layer

La couche applicative ou Application Layer

La Core Layer

La couche de base est faite de tous les composants essentiels et communs à tous les réseaux pair à pair, ces composants sont :

– pairs



FIGURE 5.2 – Couches JXTA Joseph D. Gradecki.

- pairs de découverte
- surveillance par les pairs
- communication par les pairs
- des primitives de sécurité associée

Cette couche est partagée entre tous les dispositifs pair à pair pour permettre l'interopérabilité.

Le Service Layer

Le service layer est au-dessus de la couche core layer, elle est commune pour les réseaux pair à pair. Elle ne constitue pas une nécessité absolue. Le genre de services fournis sont le partage de fichiers, la messagerie temps réel, le stockage et le systèmes de fichiers...etc.

L'Application Layer Les composants logiciels au niveau de cette couche peuvent être utilisés comme services par d'autres composants services. C'est dans cette couche que les développements d'applications JXTA sont effectués.

5.1.3.4 Services fournis par l'architecture JXTA

La technologie JXTA permet aux développeurs de déployer une application pair à pair, dont tous les services utilisés respecte la norme pair à pair, et cela indépendamment du langage de programmation informatique ou du système d'exploitation. JXTA peut être utilisé pour la mise en œuvre d'applications sur de nombreux appareils avec des piles logicielles totalement différentes. JXTA donne également la capacité de travailler sur toute topologie réseau à l'insu de l'architecte du réseau. Après cette brève introduction aux concepts JXTA, la phase suivante sera consacrée à l'analyse et la conception du calcul distribué.

5.2 DISTRIBUTION DU CALCUL D'OPTIMISATION

5.2.1 Modélisation

Cette section explique le principe de fonctionnement de l'application. Au départ, il y a une exposition des exigences de l'application dont le principe est la mise en place d'une plateforme de calcul distribué, basée sur une architecture pair à pair. Pour atteindre cet objectif, plusieurs objectifs intermédiaires doivent être remplis. Ces différentes conditions ainsi que les actions nécessaires, sont détaillées dans le diagramme de la figure 5.3 et 5.4.

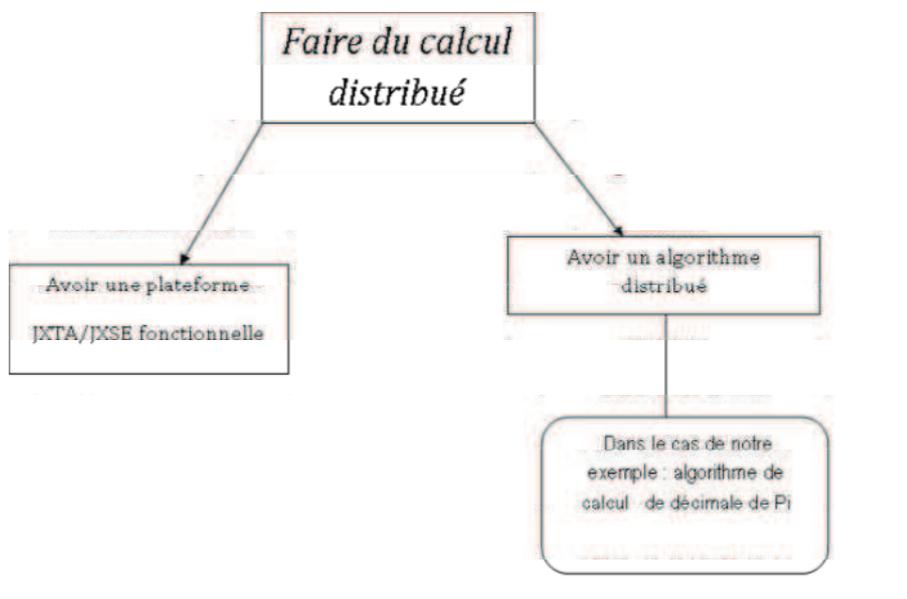


FIGURE 5.3 – Diagramme des buts du problème.

L'objectif fondamental est de manipuler le mécanisme de déploiement d'un service sur le réseau, ce service sera par la suite l'algorithme de calcul distribué.

Fondamentalement, obtenir une plateforme fonctionnelle passe par le fait d'arriver à faire

en sorte que les pairs sur le réseau puissent se trouver, se différencier, communiquer et cela indépendamment de la topologie réseau.

Pour mettre en place une architecture de calcul, un moniteur de connectivité est indispensable pour permettre à chaque instant de connaître l'état du réseau.

Le but est de pouvoir disposer d'un système fonctionnel, remplissant les objectifs exposés dans le diagramme 5.4 :

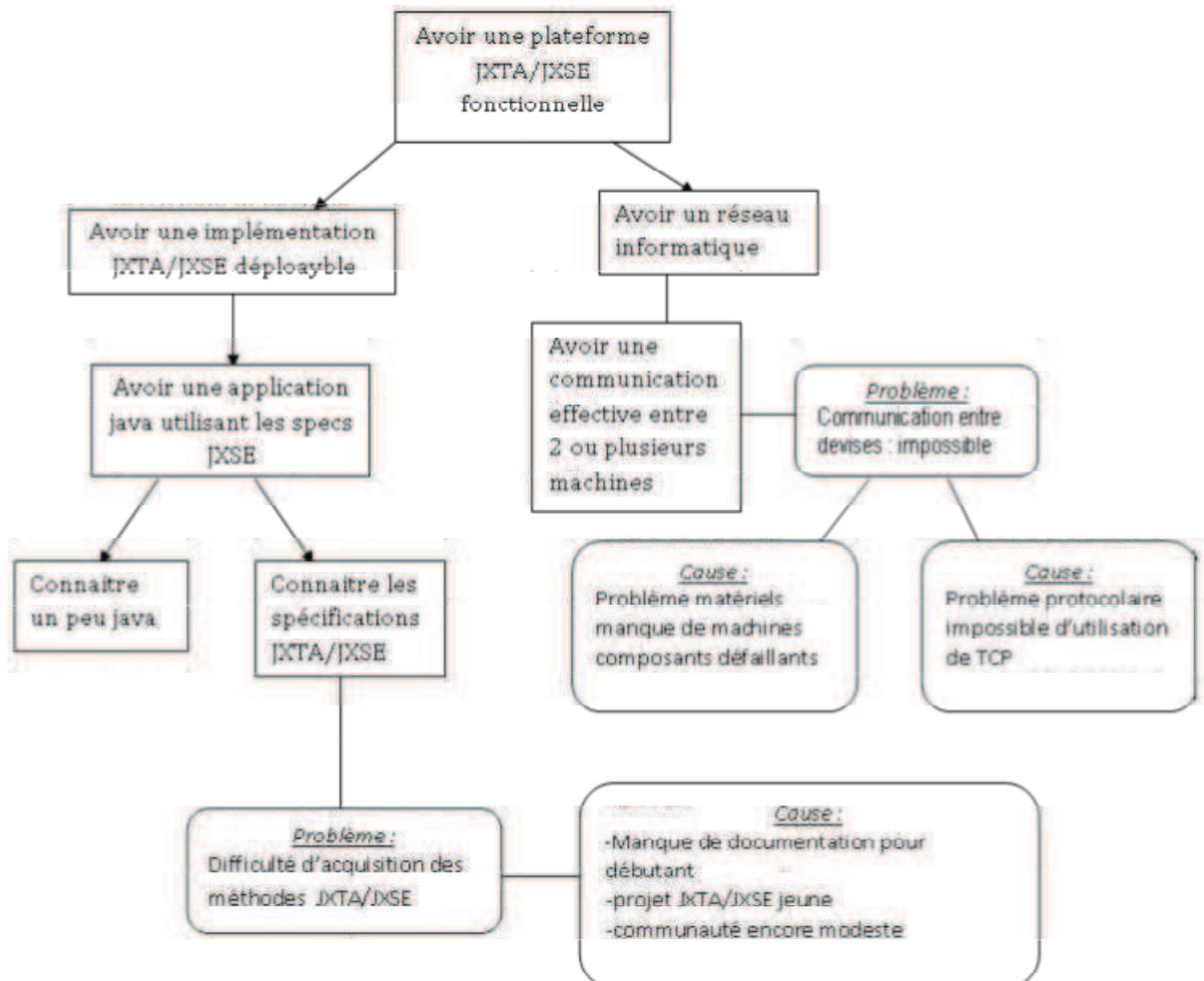


FIGURE 5.4 – Diagramme des buts du problème (suite).

5.2.2 Stratégie de découpage de l'application

Les spécificités attendues de cette application sont de faire cohabiter les trois entités, à savoir, le edge Peer, le RDV Peer et le service. Dans le but d'obtenir un squelette fonctionnel de l'application.

Dans cette conception, l'application développée est une sorte de suite d'interactions entre un RDV pair, à la fois moniteur d'un groupe de pairs et muni également d'un moniteur de

connectivité (un moniteur lui permettant de constater en temps réel, quel pair est disponible sur le réseau). Ensuite l'ensemble des Edge Peers qui héberge le service de calcul et qui met à la disposition du RDV demandeur leurs ressources de calcul.

5.2.3 Interface utilisateur

L'interface graphique d'une application est essentiellement faite pour faciliter son utilisation. Cela permettra aux utilisateurs d'accroître la performance en utilisant des composants de l'application.

Il y a deux fenêtres principales JFrame qui encapsulent les fonctionnalités de tous les composants utilisés dans une application.

5.2.4 Implémentation

Cette section va enquêter sur la mise en œuvre de l'application et les composants utilisés pour créer l'application finale. En outre, les classes utilisées dans cette application et leurs objectifs seront également expliqués ainsi que l'utilisation de JXTA.

Les entités JXTA

Le Edge Peer

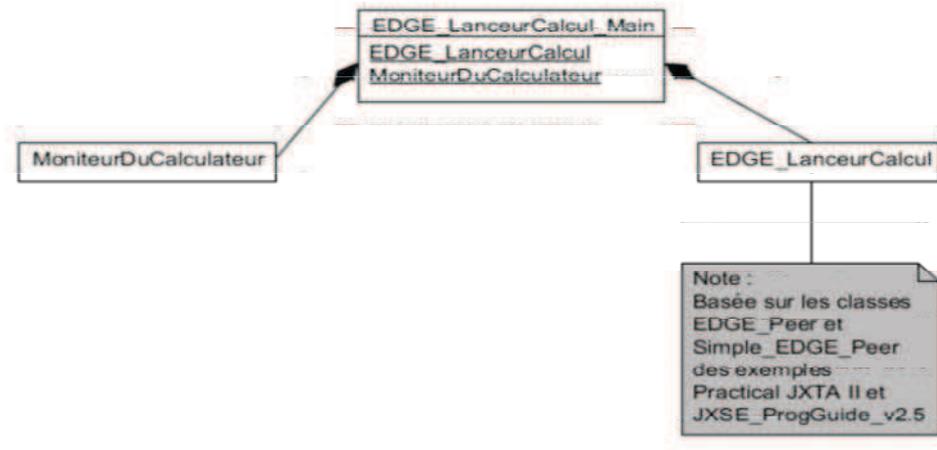
Le EDGE Peer voir figure 5.5 héberge la capacité de calcul. Une fois connecté, il mettra à disposition des RDV demandeur de ressources pour effectuer leurs opérations, ses ressources propres.

Le EDGE Peer est l'amoncellement de trois classes :

- Une métier : "EDGE LanceurCalcul"
- Une interface graphique : "MoniteurDuCalculateur"
- Une classe main faisant la jonction entre les deux classes sus-citées

Le RDV Peer

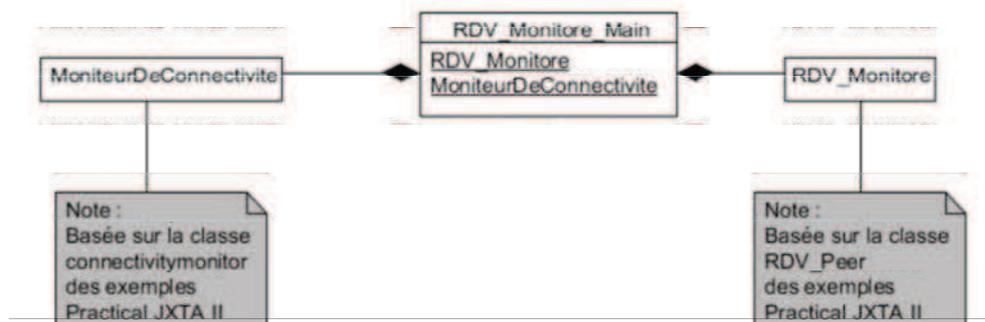
Tout comme l'EDGE ci-dessus, cette entité occupe un rôle important voire central dans l'implémentation. En effet elle représente le gestionnaire de tout calcul. Elle est également subdivisée en trois classes. Elle représente à la fois le demandeur du calcul ainsi que le réceptionniste des résultats produits par les unités de calcul que sont les EDGEs. Dans l'implémentation, cette entité est relativement basique. Il serait particulièrement avisé de jeter un œil à l'implémentation au sein de cette entité, de couches telles qu'une stratégie

FIGURE 5.5 – *Edge Peer*.

de gestion d'envoi des données aux pairs en capacité de calcul et de réception des résultats produits.

Les trois classes constitutives se présentent comme suit :

- Une classe métier : "RDV Monitore ClientDeService"
- Une interface graphique : "EDGE LanceurCalcul"
- Une classe main faisant le lien entre les deux classes sus-citées

FIGURE 5.6 – *RDV Peer*.

5.3 PREMIERES EXPERIMENTATIONS

5.3.1 L'approche BBP, pour calculer Π

Les chercheurs Simon Plouffe, David Bailey et Peter Borwein ont découverts la formule dite à la base d'un programme informatique en 1997, elle est donnée comme suit :

$$\pi = \sum_{k=0}^{\infty} \frac{1}{16^k} \left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} \right) \quad (5.1)$$

La formule BBP (Bailey-Borwein-Plouffe) ci-dessus permet de calculer assez rapidement la valeur de π . Elle permet de déterminer le nième chiffre après la virgule en base hexadécimale ou binaire sans avoir à en calculer les précédents. Cela se fait en utilisant très peu de mémoire et de temps CPU et commence à une position de départ arbitraire.

La figure 5.7 illustre la communication des pairs. Le "RDV Peer" identifie la connection des autres pairs "EDGÉs". Il demande et réceptionne les résultats produits par les unités de calcul.

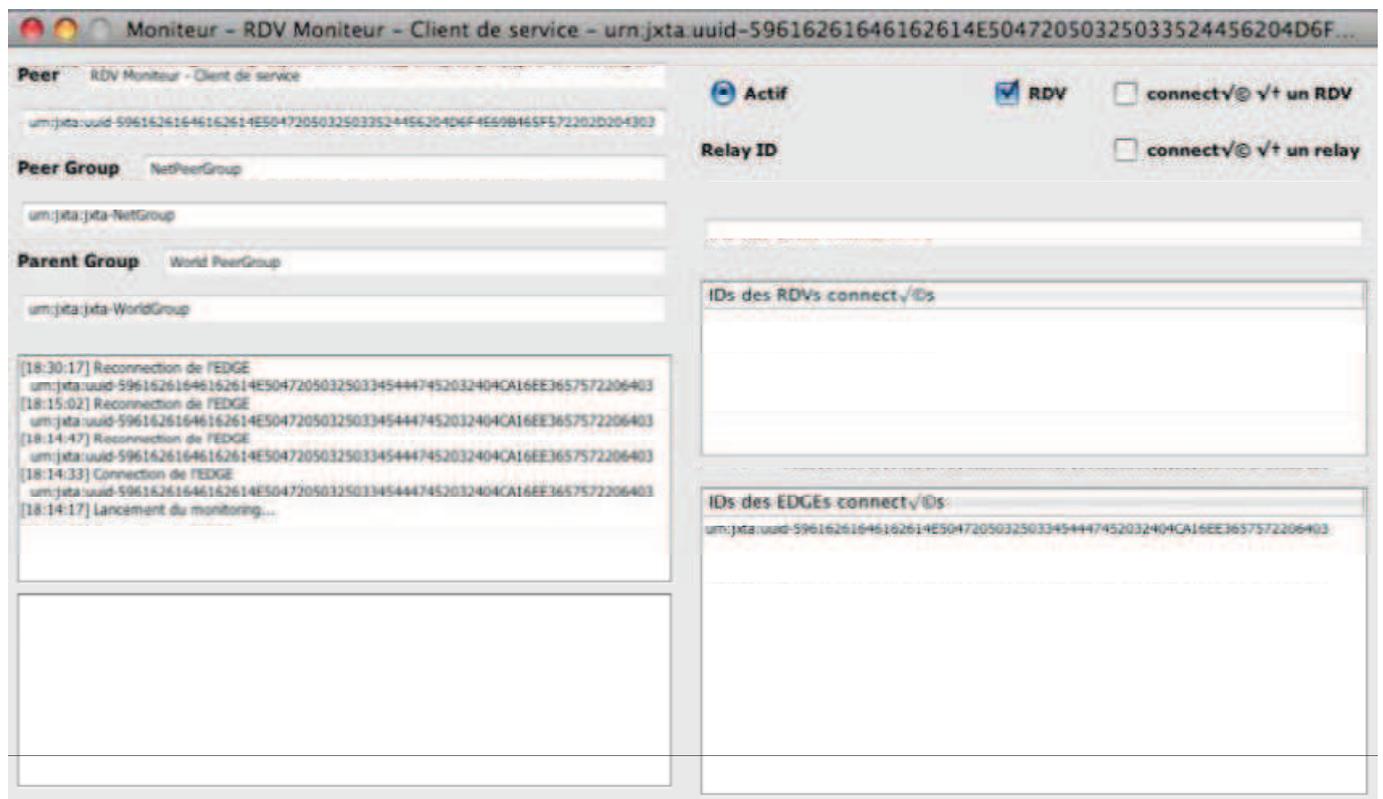


FIGURE 5.7 – Communication entre les pairs.

5.3.2 La distribution de l'algorithme MPSO-SA

La distribution de l'algorithme MPSO-SA est basée sur le principe des voisinages utilisés au sein de l'essaim de particules de la méthode (MPSO-SA). Dans un premier temps on effectue un recensement des tâches répétitives dans chaque voisinage. Elles sont illustrées dans les points suivants (cf figure 5.8) :

- Calcul du X_{Nbest} , le meilleur du voisinage ;

- Calcul de la vitesse et déplacement des particules ;
- Envois de la valeur de X_{Nbest} à l'algorithme du recuit simulé (SA) pour amélioration.

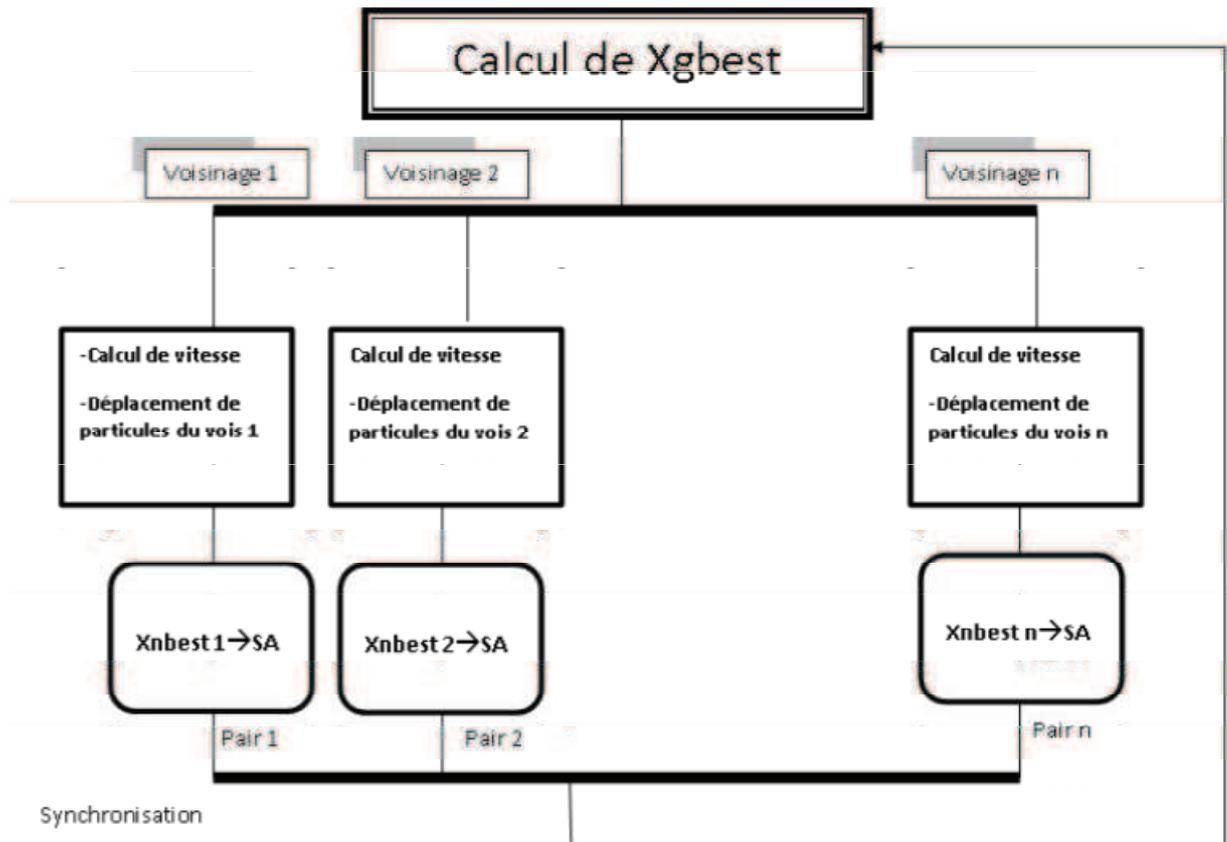


FIGURE 5.8 – L'algorithme MPSO-SA sur l'architecture pair à pair.

L'idée de base de la distribution de l'algorithme est de mettre toutes les opérations répétitives de chaque voisinage dans un pair (Comme pour le calcul des décimales de π). Le nombre de pairs est équivalent au nombre de voisinages. L'étape de synchronisation vient au moment où les différents pairs rendent leurs résultats. A ce moment le calcul de X_{gbest} , le meilleur de tout l'essaim devient possible. Ce processus est répété itérativement jusqu'à satisfaction du critère d'arrêt.

CONCLUSION

Le but de ce chapitre vise à adapter l'algorithme MPSO-SA sur des architectures distribuées pour atteindre des résultats plus rapidement voire meilleurs grâce à l'apport combiné de l'algorithme et du calcul distribué.

A cet effet, il était primordial de faire une incursion dans le monde du pair à pair et la

plateforme/protocole de développement JXTA qui est implémentée en JAVA : JXSE. Nous avons mis en place une plateforme expérimentale basée sur l'architecture pair à pair sur laquelle pourrait être déployée un algorithme de calcul distribué. Dans un premier temps un exercice de l'algorithme BBP (Bailey-Borwein-Plouffe) du calcul de π a été effectué avec succès et nous faisons évoluer l'algorithme distribué MPSO-SA vers la phase d'exécution des calculs.

Comme perspective, on propose la distribution de l'algorithme MPSO-SA qui peut être basée sur le principe des voisinages utilisé au sein de l'essaim de particules de la méthode (MPSO-SA). En effet, il s'agit de mettre toutes les opérations répétitives de chaque voisinage dans un pair (Comme pour le calcul des décimales de π). Le nombre de pairs est équivalent au nombre de voisinages. De cette manière, le temps de calcul sera en moyenne divisé par le nombre de voisinage.

La perspective de la distribution amène à d'autres approches algorithmiques. La notion même de voisinage pourrait être raffinée selon les pairs. Une approche motivante consisterait à passer la main à l'algorithme pour définir dynamiquement le nombre de voisinages, et donc les pairs dont le système aurait besoin.

CONCLUSION GÉNÉRALE

Cette thèse de doctorat en cotutelle, a été développée dans le cadre de la collaboration scientifique entre le Laboratoire d'Étude et Recherche en Mathématiques Appliquées (LERMA, École Mohammadia d'Ingénieurs au Maroc) et le Laboratoire d'Informatique et Traitement de l'Information et des Systèmes LITIS "EA 4108" de l'Institut National des Sciences Appliquées de Rouen.

Les travaux de recherche ont visé à contribuer au développement des méthodologies pour la modélisation et l'optimisation des systèmes. On a cherché à proposer des améliorations des algorithmes d'optimisation utilisés dans l'analyse des systèmes à travers une évaluation des caractéristiques et performances des algorithmes méta heuristiques.

Les méthodes utilisées sont basées sur, d'une part, les phénomènes biologiques : les simulations informatiques de vols groupés d'oiseaux et de bancs de poissons (PSO) et d'autre part, les processus physiques qui consistent à ordonner les atomes d'un cristal afin de former une structure cristalline parfaite (SA), dans le cadre de la méthode proposée dénommée MPSO-SA.

Les travaux de recherche concernent la proposition d'une nouvelle méthode hybride dénommée MPSO-SA et son application à des problèmes concrets en mécanique des structures et à l'optimisation de l'interopérabilité des systèmes d'information. Une distribution du calcul d'optimisation MPSO-SA à des fins d'amélioration des performances en utilisant une plate forme des réseaux peer to peer.

Au début des travaux de recherche, une revue détaillée de la littérature à propos de l'objet de la thèse a été réalisée. Notre travail avait pour objectif, la contribution dans le domaine d'optimisation heuristique. Nous avons proposé une nouvelle méthode d'optimisation

globale dénommée MPSO-SA. Cette méthode hybride est le résultat du couplage d'une variante de l'algorithme Essaim de Particules nommé PSO (Particle Swarm Optimisation) avec la méthode du recuit simulé nommée SA (Simulted Annealing).

Les principales contributions de cette thèse sont scindées en cinq chapitres :

Le premier dresse un état de l'art sur l'optimisation globale. Il a classifié les méthodes d'optimisation et a présenté les notions de base ainsi que les définitions de quelques méthodes métaheuristiques. Ces méthodes ont pour but la recherche de bonnes solutions (optimum global) aux problèmes d'optimisation dans les différents domaines.

Le deuxième chapitre était consacré à la présentation d'une nouvelle méthode d'optimisation hybride MPSO-SA ainsi que son implémentation avec différents langages de développement (MATLAB, JAVA). Cette nouvelle méthode hybride est basée sur un couplage de l'algorithme d'optimisation par essaim de particules modifiées MPSO (Modified Particle Swarm Optimisation) avec la méthode du recuit simulé (SA). Le processus d'hybridation utilise MPSO comme mécanisme de recherche évolutif pour acquérir une exploration efficace des solutions prometteuses de tout le domaine de recherche. Au cours de ce processus, SA est utilisée pour effectuer une amélioration des solutions trouvées. Plusieurs combinaisons ont été testées pour retenir un modèle approprié.

Le troisième chapitre a présenté une application de notre méthode à des problèmes concrets en mécanique des structures. La méthode combine la résolution des problèmes de fiabilité des structures mécaniques et la méthode hybride d'optimisation proposée dans le second chapitre. Nous avons présenté une nouvelle approche. Elle consiste à utiliser notre méthode MSPO-SA dans l'approche hybride dynamique RBDO. Avec ces différentes approches hybrides, nous avons proposé un outil d'optimisation efficace pour le contrôle de conception des structures mécaniques. À travers les deux exemples présentés (Plaque vibrante et ressort d'embrayage) nous avons montré l'efficacité de la méthode hybride proposée RBDO-MPSO-SA. Les résultats sont satisfaisants pour ces deux applications.

Le quatrième chapitre contribue aux applications de l'optimisation dans le domaine de l'interopérabilité des systèmes d'information (SI). La méthode MPSO-SA proposée est

utilisée pour trouver la meilleure répartition et optimiser la matrice d'effort nécessaire dans une situation de travail en réseau. Dans un premier lieu, une modélisation pour la représentation de l'évolution d'interopérabilité des SI en utilisant la métrique RatIop est effectuée. Puis en utilisant MPSO-SA, nous avons proposé un effort optimal théorique nécessaire pour optimiser un multi projet de mise en œuvre de l'interopérabilité (Interopérabilité de quatre et dix systèmes d'information). Les résultats obtenus permettent la comparaison de la distribution d'effort avec les propositions des architectes d'intégration et œuvrer ainsi dans l'établissement des actions concrètes pour une meilleure stratégie de gouvernance des SI.

le cinquième chapitre, propose une distribution du calcul d'optimisation à des fins d'amélioration des performances, et la constitution d'une plateforme pour le calcul scientifique dans une architecture réseau dite Peer to Peer. Cette architecture de calcul distribué va être testée pour une reconduite projetée sur l'algorithme d'optimisation MPSO-SA. Le principe de base de la distribution est de mettre toutes les opérations répétitives de chaque voisinage dans un pair (Comme pour le calcul des décimales de π). Le nombre de pairs est équivalent au nombre de voisinages. De cette manière, le temps de calcul sera en moyenne divisé par le nombre de voisinage.

ANNEXES



SOMMAIRE

A.1	LISTE DES PUBLICATIONS	114
A.2	FONCTIONS TESTS MATHÉMATIQUES	115
A.2.1	Fonction de Shubert 3 DIM	115
A.2.2	Fonction de Shubert 4 DIM	115
A.2.3	Fonction de Camel	116
A.2.4	Fonction de Dejoint with Noise	116
A.2.5	Fonction de Goldstein and Price	116
A.2.6	Fonction de Rastrigin	117
A.2.7	Fonction de Shekel	117
A.2.7.1	Fonction de Shekel m=5 SHK5	117
A.2.7.2	Fonction de Shekel m=7 SHK7	118
A.2.7.3	Fonction de Shekel m=10 SHK10	118
A.2.8	Fonction de Rosenbrock	119
A.2.9	Fonction de Branins	120
A.2.10	Fonction de Griewank	120
A.2.11	Fonction de Bohachevsky	121
A.2.11.1	Fonction de Bohachevsky BO1	121
A.2.11.2	Fonction de Bohachevsky BO2	121
A.2.12	Fonction de Hansen	121
A.2.13	Fonction de Cosine mixture	121
A.2.14	Fonction de Hartman	122
A.2.14.1	Hartman pour 3 dimension3 HA3	122
A.2.14.2	Hartman pour 6 dimensions HA6	123
A.3	MODÈLES DE RÉFÉRENCE DE L'INTEROPÉRABILITÉ DES SI	124
A.3.1	Military Communications and Information Systems Interoperability (MCISM)	124
A.3.2	NATO C3 Technical Architecture Reference Model for Interoperability	124
A.3.3	Layered Interoperability Score (i-Score)	124
A.3.4	Evaluation de l'interopérabilité appliqués aux systèmes d'entreprises	124
A.3.5	Interoperability Assessment Methodology (IAM)	125
A.3.6	Levels of Conceptual Interoperability Model (LCIM)	125
A.3.7	Layers of Coalition Interoperability (LCI)	126
A.3.8	System-of-Systems Interoperability Model (SoSI)	126
A.3.9	Organizational Interoperability Agility Model (OIAM)	126

A.1 LISTE DES PUBLICATIONS

Norelislam Elhami, Rachid Ellaia, Mhamed Itim, Hybrid Evolutionary Optimization Algorithm MPSO-SA, International Journal of Simulation and Multidisciplinary Optimisation (IJSMDO) ASMDO 2010 DOI : 10.1051=ijsmdo :2010004, Volume 4, pp 27-32 (2010).

Norelislam Elhami, Mhamed Itim, Rachid Ellaia Reliability-Based Design and Heuristic Optimization MPSO-SA of Structures , Advanced Materials Research, ISBN : 978-3-03785-163-0, Volume 274, pp95-106 (2011)

Norelislam Elhami, Rachid Ellaia, Mhamed Itim A New Methodology for the Reliability Based Particle Swarm Optimization with Simulated Annealing , Advanced Materials Research, ISBN : 978-3-03785-163-0, Volume 274, pp107-118 (2011)

Norelislam ELHAMI, Rachid ELLAIA et Mhamed ITMI, Optimisation globale par essaim de particules couplée avec la méthode du recuit simulé, Conférence Internationale CIRO'10 Marrakech 24-27 Mai (CIRO 2010).

EL HAMI Norelislam, ITMI Mhamed, ELLAIA Rachid, : Hybrid Algorithm Mpsa-Sa Combining Particle Swarm Optimization With Simulated Annealing, congrès de la Société française de Recherche Opérationnelle et d'Aide à la Décision, N° 330, Page II-573, (ROA-DEF 2011).

Norelislam Elhami, Rachid Ellaia, Mhamed Itim et Mohamed Zeriab Es-Sadek, Logistics Problems Based on MPSO-SA hybrid Optimisation Algorithms, 5^{me} conférence international, Tendances dans les Applications Mathématique en Tunisie, Algérie, Maroc, TAMTAM en Sousse Tunisie 23-26 Avril (2011).

Elmir B., El Hami N., Bounabat B., Ellaia R., Itmi M. "Optimal Control of Information System Interoperability Implementation". International Journal of Design and Innovation Research, Volume X - n°Y / 2011. (Sous revue).

A.2 FONCTIONS TESTS MATHÉMATIQUES

A.2.1 Fonction de Shubert 3 DIM

$$SH3(x) = \prod_{i=1}^3 \sum_{j=1}^5 j \cos[(j+1)x_i + j]$$

avec

$$-10 \leq x_i \leq 10$$

Cette fonction de 3 dimensions (3 variables) possède un minimum global $SH(x) = -2709.07$

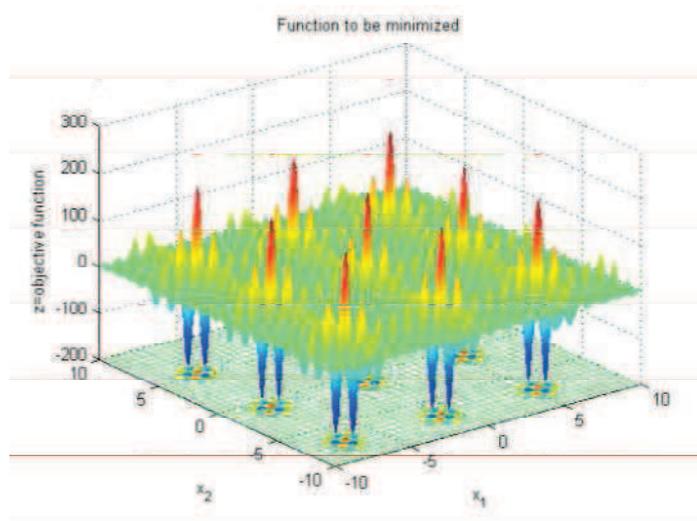


FIGURE A.1 – Représentation graphique de Shubert de 2 dimensions

A.2.2 Fonction de Shubert 4 DIM

$$SH4(x) = \prod_{i=1}^4 \sum_{j=1}^5 j \cos[(j+1)x_i + j]$$

avec

$$-10 \leq x_i \leq 10$$

Cette fonction de 4 dimensions (4 variables) possède un minimum global $SH(x) = -39303.17$

A.2.3 Fonction de Camel

$$CA(x) = 4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$$

avec

$$-10 \leq x_i \leq 10$$

Cette fonction possède un minimum $CA(x) = -1.0316285$ en $x = (\pm 0.089842, \mp 0.712656)^T$.

A.2.4 Fonction de Dejoint with Noise

$$DE(x) = \sum_{i=1}^d x_i^4 + \text{random}(0, 1)$$

avec

$$-1.28 \leq x_i \leq 1.28$$

Cette fonction possède un minimum global $DE(x) = 0$ en $x = (0, 0, \dots, 0)^T$. Elle est convexe, mais graphiquement, la zone du minimum est plus "plate", donc a priori, la minimisation est plus longue et plus difficile.

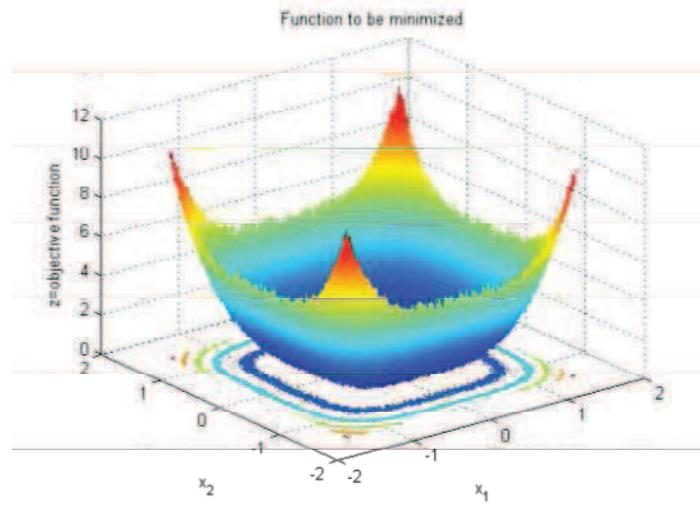


FIGURE A.2 – Représentation graphique de la fonction de Dejoint with Noise à 2 dimensions

A.2.5 Fonction de Goldstein and Price

$$GP(x) = [1 + (x_1 + x_2 + 1)^2(19 - 14x_1 + 3x_1^2 - 14x_2 + 6x_1x_2 + 3x_2^2)] * [30 +$$

$$(2x_1 - 3x_2)^2(18 - 32x_1 + 12x_1^2 + 48x_2 - 36x_1x_2 + 27x_2^2)]$$

avec

$$-5 \leq x_i \leq 5$$

Cette fonction possède un minimum global $GP(x) = 3$ en $x = (0, -1)^T$.

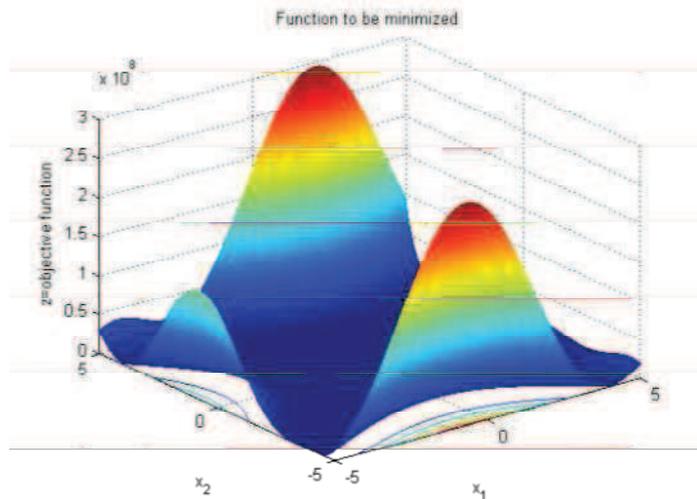


FIGURE A.3 – Représentation graphique de la fonction de Goldstein and Price à 2 dimensions

A.2.6 Fonction de Rastrigin

$$RA(x) = 10d + \sum_{i=1}^d [x_i^2 - 10\cos(2\pi x_i)]$$

avec

$$-5.12 \leq x_i \leq 5.12$$

Cette fonction possède un minimum global $RA(x) = 0$ en $x = (0, 0)^T$.

A.2.7 Fonction de Shekel

A.2.7.1 Fonction de Shekel m=5 SHK5

$$SHK5(x) = - \sum_{i=1}^5 \frac{1}{(x - a_i)^T (x - a_i) + c_i} \quad x \in \mathbf{R}^d \text{ et } m = 5$$

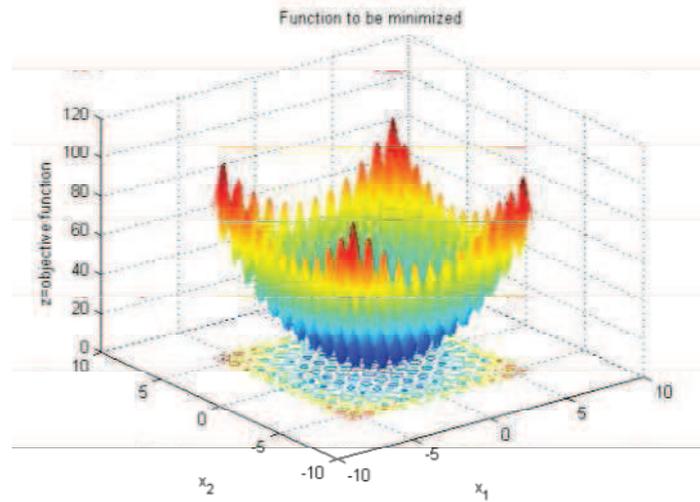


FIGURE A.4 – Représentation graphique de Rastrigin de 2 dimensions

Cette fonction possède un minimum global pour $d = 4$ $SHK5(x) = -10.1532$ en $x = (4, 4, 4, 4)^T$.

A.2.7.2 Fonction de Shekel m=7 SHK7

$$SHK7(x) = - \sum_{i=1}^7 \frac{1}{(x - a_i)^T (x - a_i) + c_i} \quad x \in \mathbb{R}^d \text{ et } m = 7$$

Cette fonction possède un minimum global pour $d = 4$ $SHK7(x) = -10.40294$ en $x = (4, 4, 4, 4)^T$.

A.2.7.3 Fonction de Shekel m=10 SHK10

$$SHK10(x) = - \sum_{i=1}^{10} \frac{1}{(x - a_i)^T (x - a_i) + c_i} \quad x \in \mathbb{R}^d \text{ et } m = 10$$

Cette fonction possède un minimum global pour $d = 4$ $SHK10(x) = -10.53641$ en $x = (4, 4, 4, 4)^T$.

Les coefficients de la fonction de Shekel a_{ij} , c_i sont :

i/j	$a_{i,1}$	$a_{i,2}$	$a_{i,3}$	$a_{i,4}$	c_i
1	4.0	4.0	4.0	4.0	0.1
2	1.0	1.0	1.0	1.0	0.2
3	8.0	8.0	8.0	8.0	0.2
4	6.0	6.0	6.0	6.0	0.4
5	3.0	7.0	3.0	7.0	0.4
6	2.0	9.0	2.0	9.0	0.6
7	5.0	5.0	3.0	3.0	0.3
8	8.0	1.0	8.0	1.0	0.7
9	6.0	2.0	6.0	2.0	0.5
10	7.0	3.6	7.0	3.6	0.5

TABLE A.1 – *four-dimensional Shekel function*

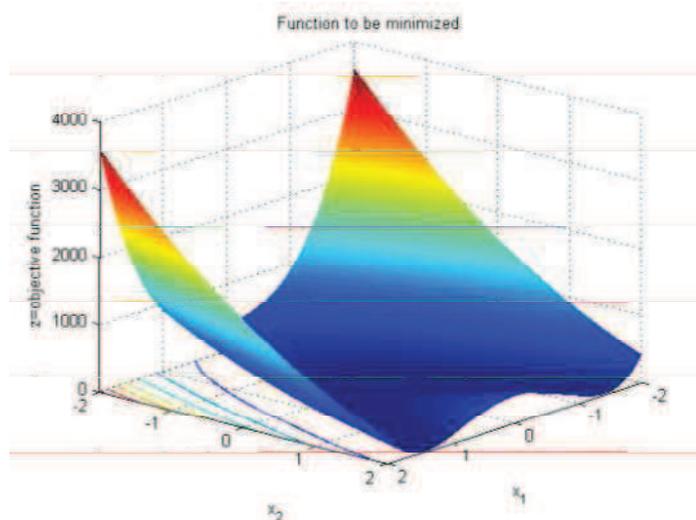
A.2.8 Fonction de Rosenbrock

$$RO(x) = 100(x_2 - x_1^2)^2 + (1 - x_1)^2$$

avec

$$-30 \leq x_i \leq 30$$

Cette fonction possède un minimum $RO(x) = 0$ en $x = (1, 1, \dots, 1)^T$. La difficulté pour la recherche de ce point est liée au fait qu'il se trouve à l'intersection de deux vallées de pentes très faibles.

FIGURE A.5 – *Représentation graphique de la fonction de Rosenbrock à 2 dimensions*

A.2.9 Fonction de Branins

$$BR(x) = a(x_2 - bx_1^2 + cx_1 - d)^2 + e(1 - f)\cos(x_1)$$

Il est recommandé d'affecter aux paramètres de la formule les valeurs suivantes :

$$a = 1, b = 5.1/4\pi^2, c = 5/\pi, d = -, e = 10 \text{ et } f = 1/8\pi$$

La valeur optimal de la fonction est $f(x_1, x_2) = 0.397887$

Trois positions : $(x_1, x_2) = (-\pi, 12.275), (\pi, 2.275), (9.42478, 2.475)$.

A.2.10 Fonction de Griewank

$$GR(x) = \frac{1}{4000} \sum_{i=1}^d x_i^2 - \prod_{i=1}^d \cos\left(\frac{x_i}{\sqrt{i}}\right) + 1$$

avec

$$-600 \leq x_i \leq 600$$

Cette fonction possède un minimum global $GR(x) = 0$ en $x = (0, 0)^T$.

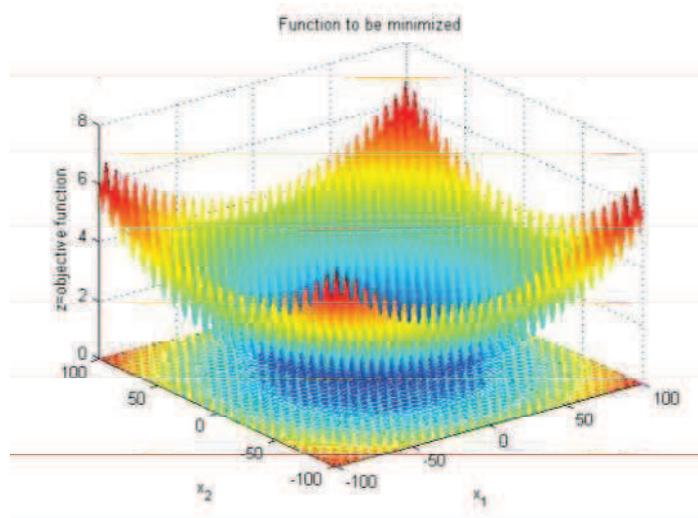


FIGURE A.6 – Représentation graphique de la fonction de Griewank à 2 dimensions

A.2.11 Fonction de Bohachevsky**A.2.11.1 Fonction de Bohachevsky BO1**

$$BO1(x) = \sum_{i=1}^{n-1} [x_i^2 + 2x_{i+1} - 0.3\cos(3\pi x_i) - 0.4\cos(4\pi x_{i+1}) + 0.7]$$

Le domaine de recherche est $-100 \leq x_i \leq 100, i = 1, 2$

L'optimum est $(0, 0)$, la valeur de la fonction optimal est 0.

A.2.11.2 Fonction de Bohachevsky BO2

$$BO2(x) = \sum_{i=1}^{n-1} [x_i^2 + 2x_{i+1} - 0.3\cos(3\pi x_i) * \cos(4\pi x_{i+1}) + 0.3]$$

Le domaine de recherche est $-100 \leq x_i \leq 100, i = 1, 2$

L'optimum est $(0, 0)$, la valeur de la fonction optimal est 0.

A.2.12 Fonction de Hansen

$$HAN(x) = \sum_{i=0}^4 (i+1)\cos(ix_0 + i+1) \sum_{j=0}^4 (j+1)\cos((j+2)x_1 + j+1)$$

$-10 \leq x_i \leq 10$

Le minimum global de cette fonction est $HAN(x) - 176.54$.

A.2.13 Fonction de Cosine mixture

$$CM(x) = \sum_{i=1}^d x_i^2 - 0.1 \sum_{i=1}^d \cos(5\pi x_i)$$

avec

$$-1 \leq x_i \leq 1$$

Cette fonction possède un minimum $CM(x) = -0.1 d$ en $x = (0, 0, \dots, 0)^T$. Avec d les dimensions du problème

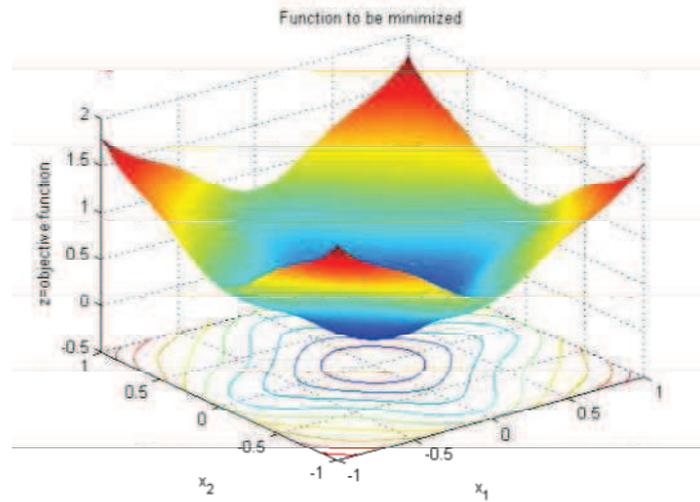


FIGURE A.7 – Représentation graphique de la fonction de Cosine mixture à 2 dimensions

A.2.14 Fonction de Hartman

A.2.14.1 Hartman pour 3 dimension3 HA3

Pour $n = 3$:

$$HA3(x) = - \sum_{i=1}^4 \alpha_i \exp \left[- \sum_{j=1}^3 A_{ij} (x_j - P_{ij}^2) \right]$$

$$\alpha = [1, 1.2, 3, 3.2]^T$$

$$A = \begin{pmatrix} 3.0 & 10 & 30 \\ 0.1 & 10 & 35 \\ 3.0 & 10 & 30 \\ 0.1 & 1 & 35 \end{pmatrix} \quad P = 10^{-4} \begin{pmatrix} 6890 & 1170 & 2673 \\ 4699 & 4387 & 7470 \\ 1091 & 8732 & 5547 \\ 381 & 5743 & 8828 \end{pmatrix}$$

$$0 \leq x_i \leq 1, i = 1, 2, 3$$

Le minimum global est : $X_* = (0.114614, 0.555649, 0.8525447)$

La valeur de fonction objectif $f(X_*) = -3.86278$

A.2.14.2 Hartman pour 6 dimensions HA6**Pour $n = 6$:**

$$HA6(x) = - \sum_{i=1}^4 \alpha_i \exp\left[- \sum_{j=1}^6 B_{ij}(x_j - Q_{ij}^2)\right]$$

$$\alpha = [1, 1.2, 3, 3.2]^T$$

$$B = \begin{pmatrix} 10 & 3 & 17 & 3.05 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix} Q = 10^{-4} \begin{pmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{pmatrix}$$

$$0 \leq x_i \leq 1, i = 1, 2, 3, \dots, 6$$

Le minimum global est : $X_* = (0.20169, 0.150011, 0.476874, 0.275332, 0.311652, 0.6573)$

La valeur de fonction objectif $f(X_*) = -3.32237$

A.3 MODÈLES DE RÉFÉRENCE DE L'INTEROPÉRABILITÉ DES SI

A.3.1 Military Communications and Information Systems Interoperability (MCISM)

Appliquée au domaine de la collaboration militaire, ce modèle consiste en la confection d'une matrice tridimensionnelle où sont répertoriés le croisement entre le niveau de commandes, services et le média de communication [Amanowicz et Gajewski (1996)]. Elle utilise des indicateurs rouges, jaunes et verts pour indiquer l'inexistence, l'existence partielle, ou l'interopérabilité complète. Une fois cette matrice élaborée, il s'agit de calculer les distances entre les différents composants mis en jeu en utilisant des outils mathématique d'analyse adéquats.

A.3.2 NATO C3 Technical Architecture Reference Model for Interoperability

La quatrième version 4 de ce modèle s'est rapprochée du modèle LISI [NATO (2003)]. Ce modèle cinq degrés d'interopérabilité :

- Aucune interopérabilité ;
- Échange non structuré de données ;
- Échange structure de données ;
- Échange transparent et automatisé de données ;
- Échange transparent et automatisé des informations.

A.3.3 Layered Interoperability Score (i-Score)

i-Score [Ford et al., 2007b] est une méthode de quantification de l'interopérabilité faisant appel à des techniques mathématiques et qui prend en considération les différents aspects accompagnant une situation de coalition militaire : Aspects techniques, Aspects biologiques, Aspects organisationnels, Aspects environnementaux.

A.3.4 Évaluation de l'interopérabilité appliqués aux systèmes d'entreprises

Après les domaines militaires et de recherches scientifiques, les acteurs des SI portent les résultats obtenus et les méthodes existantes au domaine des entreprises et des organisations en général.

A.3.5 Interoperability Assessment Methodology (IAM)

La méthodologie IAM s'inspire du modèle QoIM [Leite, 1998]. IAM contient neuf composants caractérisant l'interopérabilité (exigences, standards, éléments de données, connectivité des nœuds, protocoles, flux d'information, latence, interprétation, utilisation de l'information). IAM introduit la notion de degrés d'interconnexion : connectivité, disponibilité, interprétation, compréhensibilité, utilité, exécution, et le feedback. Les nouveautés de la méthode IAM permettent d'étendre le modèle standard LISI [Kasunic et Anderson (2004)].

A.3.6 Levels of Conceptual Interoperability Model (LCIM)

Le modèle LCIM (Levels of Conceptual interoperability Model) [Tolk et al., 2003] propose une approche plus conceptuelle. Il se focalise sur la qualité de documentation des données qui vont être échangées, et sur la qualité des interfaces entre les systèmes inter opérants. LCIM dispose de beaucoup de similitudes avec LISI et OIMM sauf qu'il est utilisé essentiellement lors des phases de modélisation afin de détecter les opportunités d'interopérabilité entre les systèmes. Il propose cinq niveaux conceptuels :

- Données spécifiques aux systèmes ;
- Données documentées ;
- Données alignées statiquement ;
- Données alignées dynamiquement ;
- Données harmonisées.

Cette classification a été par la suite étendue et change dans une version avancées par :

- Aucune interopérabilité ;
- Technique ;
- Syntaxique ;
- Sémantiques ;
- Pragmatique ;
- Dynamique ;
- Conceptuel.

A.3.7 Layers of Coalition Interoperability (LCI)

LCI est un modèle combinant les caractéristiques opérationnelles et techniques [Tolk, 2003]. En utilisant neuf niveaux :

- Physique ;
- Protocoles ;
- Données et objets ;
- Informations ;
- Connaissance ;
- Alignement des procédures ;
- Alignement des opérations ;
- Alignement stratégique ;
- Partage des objectifs politiques.

A.3.8 System-of-Systems Interoperability Model (SoSI)

le modèle SoSI est développé par le Software Engineering Institute afin de supporter leur approche de systèmes de systèmes pour caractériser l'interopérabilité [Morris (2004)]. Ce modèle est constitué de trois niveaux :

- Opérationnel
- Constructionnel
- Programmatique

A.3.9 Organizational Interoperability Agility Model (OIAM)

Le modèle OIAM se base sur le modèle OIMM. Il s'agit d'une extension qui s'intéresse aux aspects d'agilité des systèmes dans le cadre de la collaboration [Kingston et al. (2005)]. OIAM propose lui aussi cinq niveaux :

- Statique ;
- Souple ;
- Accommode ;
- Ouvert ;
- Dynamique.

OIAM utilise les quatre attributs de l'OIMM tout en combinant les deux attributs caractérisant la préparation et la compréhension en un seul attribut.

LISTE DES ABRÉVIATIONS

BO	Fonction Bohachevsky
BR	Fonction Branin
CA	Fonction Camel
CM	Fonction Cosine Mixture
DE	Fonction Dejong
ERP	Endpoint Routing Protocol
FORM	First Order Reliability Method
GP	Fonction Goldstein Price
GR	Fonction Griewank Price
HAN	Fonction Hansen
HA	Fonction Hartman
GIMM	Governmental Interoperability Maturity Model
LISI	(Levels of Information System Interoperability Model)
MEF	Méthodes des Eléments Finis
MEFP	Méthodes des Eléments Finis Probabilistes
MEFS	Méthodes des Eléments Finis Stochastiques
MPSO	Modified Particle Swarm Optimisation
OIMM	Organizational Interoperability Maturity Model
PSO	Particle Swarm Optimisation
QoIM	Quantification of Interoperability Methodology
RA	Fonction Rastrigin
RBDO	Reliability Based Design Optimization

PBP	Peer Pipe Binding
PEP	Peer End point Protocol
P2P	Peer to Peer
PDP	Peer Discovery Protocol
PIP	Peer Information Protocol
PRP	Peer Resolver Protocol
RO	Fonction Rosenbrock
RVP	Rendez-Vous Protocol
SA	Simulted Annealing
SH	Fonction Shubert
SI	Systèmes d'Information
SK	Fonction Shekel
SI	Systèmes d'Information
SoIM	Spectrum of Interoperability Model
SORM	Second Order Reliability Method

BIBLIOGRAPHIE

- Mebarki A. l'approche probabiliste de la fiabilité des structures de génie civil : laméthode de l'hypercône et ses algorithmes d'application. *Thèse de doctorat d'état (Es Sciences) de l'Université Paul Sabatier de Toulouse, France*, 1990. (Cité page 62.)
- M.M. Ali et M.N. Gabere. A simulated annealing driven multi-start algorithm for bound constrained global optimization. *Journal of Computational and Applied Mathematics*, page 2661–2674, 2010. (Cité page 33.)
- M. Amanowicz et P. Gajewski. Military communications and information systems interoperability. *Proceedings of the 1996 IEEE Military Communications Conference*, 1996. (Cité page 124.)
- D.P. Bertsekas. Dynamic programming and optimal control. *Athena Scientific*, 1 and 2 2nd ed, 2000. (Cité page 20.)
- B. Bochenek et P. Fory's. Structural optimization for post buckling behavior using particle swarms. *Struct Multidisc Optim*, pages 521–531, 2006. (Cité pages 33 et 40.)
- Adnane CABANI. Réseaux pair-à-pair et simulation distribuée. application à la simulation multi agents. *Thèse de l'INSA de Rouen*, 2007. (Cité page 96.)
- S. Chekroun. Contribution a l'étude des moteurs asynchrones a haut rendement. *thèse de doctorat, Université de Batna, Algérie*, 2011. (Cité pages 19, 20 et 21.)
- D. Chen, Vallespir B, et Daclin N. An approach for enterprise interoperability measurement. *Proceedings of MoDISE-EUS France*, 2008. (Cité page 86.)
- T. Clark et R. Jones. Organisational interoperability maturity model for c2. *Proceedings of the 9th International Command and Control Research and Technology Symposium. Newport*, 1999. (Cité page 80.)
- Y. COOREN. Perfectionnement d'un algorithme adaptatif d'optimisation par essaim particulaire applications en génie médical et en électronique. *PhD thesis, Université Paris 12*, 2008. (Cité page 20.)
- S. Datta et McLaughlin S.W. An enumerative method for runlength limited codes : Permutation codes. *IEEE Trans. Information Theory*, 45 :2199–2204, 1999. (Cité page 20.)
- F. Deheeger. Couplage mécano fiabiliste : 2smart méthodologie d'apprentissage stochastique en fiabilité. *Thèse présentée pour obtenir le grade de Docteur de Université BLAISE PASCAL - Clermont II*, 28 janvier 2008. (Cité page 59.)
- De DepartmentOfDefense. Levels of information system interoperability (lisi). *C4ISR Architecture Working Group Final Report*, Washington DC : OSD(ASD(C3I)) C4ISR AWG, 1998. (Cité page 79.)

- D. Devictor. Fiabilité et mécanique : méthodes form/sorm et couplages avec des codes d'éléments finis par des surfaces de réponse adaptatives. *Thèse présentée pour obtenir le grade de Docteur d'Université*, 1996. (Cité pages 62 et 64.)
- M. Dorigo. Optimization, learning and natural algorithms. *Ph.D. Thesis, Politecnico di Milano, Italy*, 1992. (Cité page 28.)
- F. Duprat. Probabilistic approach to corrosion risk due to carbonation via an adaptive response surface method. *Probabilistic Engineering Mechanics*, 21 :207–216, 2006. (Cité page 59.)
- R.C. Eberhart et J. Kennedy. A new optimizer using particle swarm theory, in : Proceedings sixth symposium on micro machine and human science. *IEEE Service Center, Piscataway*, pages 39–43, 1995. (Cité pages 3 et 33.)
- N. Elhami, R. Ellaia, et M. Itim and. Hybrid evolutionary optimization algorithm mpso-sa. *Int. J. Simul. Multidisci.Des. Optim ASMDO 2010 DOI : 10.1051=ijsmdo :2010004*, 4 : 27–32, 2010. (Cité pages 43 et 45.)
- N. Elhami, M. Itim, et R. Ellaia. A new methodology for the reliability based particle swarm optimization with simulated annealing. *Advanced Materials Research ISBN : 978-3-03785-163-0*, 274 :107 118, 2011a. (Cité page 65.)
- N. Elhami, M. Itim, et R. Ellaia. Reliability-based design and heuristic optimization mpso-sa of structures. *Advanced Materials Research ISBN : 978-3-03785-163-0*, 274 :95 106, 2011b. (Cité page 69.)
- B. Elmir et B. Bounabat. Integrated public e-services interoperability assessment. *International Journal of information Science and Management Special issue*, page 1 12, 2010. (Cité page 83.)
- T. Ford. Interoperability measurement department of the air force air university, air force institute of technology. *Ph.DDissertation*, 2008. (Cité page 83.)
- A.S. Fraser. Simulation of genetic systems by automatic digital computers. *Australian Journal of Biological Sciences*, 10 :484 491, 1957. (Cité page 23.)
- A. Gherbi, Francois B, , et Belkacemi M. Methods for power system transient stability analysis : State of the art. *Electrical and Computer Engineering, Canadian Journal*, 31 : 3–13, 2006. (Cité page 19.)
- F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 13 :533–549, 1986. (Cité page 30.)
- F. GLOVER. Tabu search and adaptive memory programming - advances, applications and challenges. *applications and challenges. Interfaces in Computer Science and Operations Research*, page 1 75, 1996. (Cité page 30.)
- P. Gottschalk. Maturity levels for interoperability in digital government. *Government Information Quarterly*, 26(1) :75–81, 2009. (Cité page 82.)
- Joseph D. Gradecki. Mastering jxta : Building java peer-to-peer applications. *John Wiley & Sons ISBN : 0471250848*, 2002. (Cité page 101.)

- Luis Guijarro. Interoperability frameworks and enterprise architectures in e-government initiatives in europe and the united states. *Government Information Quarterly*, 24 :89–101, 2007. (Cité page 76.)
- A. Haldar. Reliability assessment using stochastic finite element. *John Wiley and Sons, Inc. New York*, 2000. (Cité page 57.)
- HASOFER, A.M., et N.C LIND. Exact and invariant second-moment code format. *Journal of Engineering Mechanics*, 100 :111 121, 1974. (Cité page 63.)
- J. Héliot, C. Faidy, et B. Drubay. Trse-m code progress in the field of examination evaluation and flaw acceptance criteria. *Records of the 13th SMIRT Conference, Porto Alegre, Brazil*, 2 :647 652, 1995. (Cité page 62.)
- M. Kasunic et W. Anderson. Measuring systems interoperability : Challenges and opportunities. *Conference on the Acquisition of Software Intensive Systems, CMU/SEI-2004-TN-003*, 2004. (Cité pages 86 et 125.)
- G. Kingston, Fewell S, et Richer W. An organisational interoperability agility model. *Proceedings of the 10th International Command and Control Research and Technology Symposium. McLean, VA*, 2005. (Cité page 126.)
- S. Kirkpatrick, C. D. Gelatt, et M. P. Vecchi. Optimization by simulated annealing. *Science*, 220 :671–680, 1983. (Cité page 34.)
- H. Kubicek et R. Cimander. Three dimensions of organizational interoperability. *eGovInterop'07 Conference*, page 19, 2007. (Cité page 81.)
- L. Lamberti. An efficient simulated annealing algorithm for design optimization of truss structures. *Computers and Structures*, page 1936 1953, 2008. (Cité page 33.)
- G. LaVean. Interoperability in defense communications. *Communications, IEEE Transactions*, 28.9 :1445–55, 1980. (Cité page 79.)
- Pi. Liu, A. Der Kiureghian. Optimization algorithms for structural reliability. *Structural Safety*, 9 :161–177, 1991. (Cité page 64.)
- D. G. Luenberger. Introduction to linear and nonlinear programming. *Addison Wesley*, 1973. (Cité page 17.)
- D. Mensh, R. Kite, et P. Darby. A methodology for quantifying interoperability. *Naval Engineers Journal*, 101.3 :251, 1989. (Cité page 79.)
- E. Morris. System of systems interoperability (sosi), final report. cmu/sei-2004-tr-004. *Proceedings of the 10th International Command and Control Research and Technology Symposium. McLean, VA*, 2004. (Cité page 126.)
- J. Muzeau, Mohamed A., et J. Mitteau J.C. Fiabilité et sûreté de fonctionnement des systèmes mécaniques. *Journal of Engineering Mechanics*, 1996. (Cité page 63.)
- NATO. Nato c3 technical architecture. *NATO Allied Data Publication 34*, 2, 2003. (Cité page 124.)
- R. Nelder. A simplex method for function minimization. *Computer Journal*, 7 :308 313, 1965. (Cité page 20.)

- J. Nocedal et S.J. Wright. Numerical optimization. *Springer*, 1999. (Cité page 20.)
- Flores Macias O. Modèle fiabilistes et mécaniques : éléments finis stochastiques. méthode de coulage et applications. *Thèse de Doctorat de l'Université Blaise Pascal Clermont Ferrand II, France*, 1994. (Cité page 62.)
- H. Proccacia. Méthodes avancées d'analyse des bases de données du retour d'expérience industriel. *Collection de la direction des études et recherches d'Electricité de France Eyrolles*, 1996. (Cité page 63.)
- S. S. Rao. Engineering optimization, theory and practice. *Wiley*, 1996. (Cité page 17.)
- V. Savsani, R.V. Rao, et D.P. Vakharia. Optimal weight design of a gear train using particle swarm optimization and simulated annealing algorithms. *Mechanism and Machine Theory*, page 531 541, 2010. (Cité page 33.)
- A. Schrijver. Theory of linear and integer programming. *John Wiley and sons*, 1998. (Cité page 20.)
- S. Sitarz. Ant algorithms and simulated annealing for multicriteria dynamic programming. *Computers and Operations Research*, page 433 441, 2009. (Cité page 33.)