REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE

MINISTERE DE L'ENSEIGNEMENT SUPERIEUR

UNIVERSITE SAAD DAHLAB BLIDA

Faculté des Sciences

Département Informatique

Présenté par :

M^elle DJELLOUT Nour El Houda

M^elle BOULARAS Yasmina

Mémoire de fin d'études pour l'obtention du diplôme de Master

Domaine : Mathématique et Informatique
Filière : Informatique
Option : Ingénierie de logiciels

THÈME

Implémentation parallèle de « particule swarm optimisation »pour l'identification paramétrique de la machine asynchrone

Promoteur :

Mr OULD AISSA ahmed

Soutenue le : / / 2012 **devant le jury composé de :**

Président :
Examinateur :
Examinateur :

Année universitaire: 2011/2012

Résumé :

La modélisation mathématique de la machine asynchrone qui représente un modèle proche de la réalité physique, est un système d'équation différentiel de plusieurs inconnus, nécessite un temps de calcule très lent. De ce fait, nous utilisant une méthode d'optimisation par: Méta heuristique PSO.

Ce travail consiste à l'implémentation parallèle de la nouvelle méthode d'optimisation « Particle Swarm Optimisation » pour l'identification paramétrique de la machine asynchrone.

L'application de PSO sous un réseau va permettre de réduire le temps de calcul qui permettre au concepteur des machines de réalisés plusieurs testes pour la validation des résultats obtenus.

Les Mots clé :

Méta-heuristique, optimisation, essaim particulaire, continue, identification paramétrique, la machine asynchrone, réseau.

ملخص

النمذجة الرياضية للآلات المتزامنة،التي تمثل نموذج قريب من الواقع المادي، هو نظام من المعادلات التفاضلية ذات عدة مجاهيل، ويتطلب وقت حساب طويل جدا. ولذلك، فإننا نقترح وسيلة لتحسين: في الشبكة

هذا العمل يتكون من تنفيذ موازي للطريقة الجديد "تحسين سرب الجسيمات"لحساب قيم عناصر الآلات المتزامن

تطبيق يسو في شبكة ينقص من الوقت اللازم للحساب والتي يسمح للمصمم آلات أن يقوم بعدة اختبار للتحقق من صحة النتائج التي حصل عليها،لقد استخدمنا تصميمين على الشبكة ووجدنا إن التصميم الثاني يعطى نتائج افظل من أول وذلك راجع إلى سلوك الجسيم

Abstract :

The mathematic modeling of the asynchronous machine that represent an model close to the physical reality, is an equation deferential system of many unknowns, need a very slow time of calculating. And for all this we propose a method of optimization through "metha heuristique PSO"

This work consists in the parallel implementation of the new method of optimization <practical swarm optimization> for the parametrical identification of the asynchronous machine.

Application PSO IN THE networked reduces the time required for the calculation which allows the machine designer which allows the machines designer to perform many tests to verify the validity of the resultants obtained.

Remerciements

Tout d'abords, il est de notre devoir d'exprimer nos remerciements à Dieu, le tout puissant qui nous a donné, la force pour réaliser ce travail, et puis à nos parents qui nous ont été un grand soutien pendant toute la période de nos études et dés le début de ce travail.

Nous tenons aussi à remercier avec un grand plaisir et un grand respect notre encadreur Mr OULD AISSA Ahmed, promoteur de notre mémoire pour ses conseils, sa disponibilité et ses encouragements qui nous ont permis de réaliser ce travail dans les meilleures conditions.

Nous adressons de même remerciements aux membres du jury pour le temps consacré à la lecture de ce mémoire et leur aimable attention.

Sans oublier de présenter nos grands remerciements et respects à tous ceux qui ont accordé leur soutien.

DEDICACE

À celle qui m'a appris que la patience est la clé du succès et de la Victoire.

À ma mère

À celui qui m'a indiqué la bonne voie en me rappelant que la science et la volonté forgent les grands esprits.

À mon père

Et à ceux qui m'ont aidé et conseillé et qui ont fait preuve d'une patience sans égal face à tous mes caprices d'étudiante exigeante, perfectionniste et infatigable.

À ma sœur et son époux

À mes beaux frères Mounir et ishak.

Ainsi À tous mes amis Ishak, Yasmina, Asma, Wassila pour votre sincère amitié, votre soutien permanent me remonte le moral et vos conseils m'incitent à relever les défis.

Je ne peux nommer ici toutes les personnes qui de près ou de loin m'ont aidé et encouragé mais je les en remercie vivement.

A toute la promotion 2011/2012, ainsi qu'à tous mes professeurs.

Nour EL Houda.

DEDICACE

Grace a dieu, j ai pu réaliser ce travail que je dédie avec mes sentiments les plus profonds :

A mes chères parents qui ont étés toujours présents dans chaque pas que j ai fait dans ma vie.

A mes frères fers ; Khaled et son épouse Asma et à mes sœurs Ahlem ; Abla et son époux islem.

A toute ma famille qui m'a toujours encouragé et soutenue.

A tous mes cousins et cousines.

A mes très chères amis : Nour El Houda, Asma, wassila, Zola, wail, ahmed et Abdou.

A toute la promotion 2011/2012, ainsi qu'à tous mes professeurs.

Yasmina.

Sommaire

Tableaux et figures

Introduction général :

Les machines électriques jouent un rôle fondamental dans les différentes branches de l'industrie. Pour avoir une bonne commande de la machine requière un modèle basé sur des paramètres physique et une méthode d'identification de la machine asynchrone.

La modélisation et l'identification de la machine asynchrone ont un impact non négligeable sur la précision des résultats obtenus. Plusieurs approches d'identification sont mise en œuvre pour le cas de la machine asynchrone :

- Identification classique.
- Identification à l'aide des méthodes numérique.

Ces deux méthodes présentent des inconvénients.

De ce fait, Nous utilisons une méthode Méta heuristique« PSO » pour l'optimisation des paramètres adaptés au problème posé, cette méthode se base sur la collaboration des individus entre eux : chaque particule se déplace et à chaque itération donc l'évolution de la recherche de l'optimum est réalisé après certain nombre d'itérations cela implique un calcul répété de la fonction objective suivant le nombre des particules et le nombre des itérations. Donc le déploiement de l'application sur une seule station nécessite un temps très lent. En effet on propose de répartir l'effort de calcul sur un réseau de station.

L'installation d'un réseau va permettre de faciliter le calcul, plusieurs architecture sont mise en œuvre.

Ce mémoire s'articule autour de quatre chapitres :

Le premier chapitre, présente un état de l'art sur la machine asynchrone triphasée. Nous rappelons le modèle de la machine à cinq puis quatre paramètres électriques.

Dans le deuxième Chapitre, nous commençons par rappeler l'optimisation des problèmes avec une classification suivant les problèmes traités et suivant les techniques de recherche. Et on doit étudier les différentes métas heuristiques, avec une étude détaillée du nouvelle méta heuristique nommé : PSO.

Dans le troisième Chapitre, nous présentons la conception de notre système en utilisent le langage de modélisation UML (*Unified Modeling Language).*

Dans le quatrième chapitre, est consacré a la présentation et teste de notre application Nous terminons ce travail par une conclusion générale.

$$\begin{cases} \dfrac{dI_{ds}}{dt} = -\dfrac{1}{\sigma T_s} I_{ds} + \dfrac{1-\sigma}{\sigma} P\Omega I_{qs} + \dfrac{1-\sigma}{\sigma T_r} I'_{dr} + \dfrac{1-\sigma}{\sigma} P\Omega I'_{qr} + \dfrac{V_{ds}}{\sigma L_s} \\ \dfrac{dI_{qs}}{dt} = -\dfrac{1-\sigma}{\sigma} P\Omega I_{ds} - \dfrac{1}{\sigma T_s} I_{qs} - \dfrac{1-\sigma}{\sigma} P\Omega I'_{dr} + \dfrac{1-\sigma}{\sigma T_r} I'_{qr} + \dfrac{V_{qs}}{\sigma L_s} \\ \dfrac{dI'_{dr}}{dt} = \dfrac{1}{\sigma T_s} I_{ds} - \dfrac{P\Omega}{\sigma} I_{qs} - \dfrac{1}{\sigma T_r} I'_{dr} - \dfrac{P\Omega}{\sigma} I'_{qr} - \dfrac{V_{ds}}{\sigma L_s} \\ \dfrac{dI'_{qr}}{dt} = \dfrac{P\Omega}{\sigma} I_{ds} + \dfrac{1}{\sigma T_s} I_{qs} + \dfrac{P\Omega}{\sigma} I'_{dr} - \dfrac{1}{\sigma T_r} I'_{qr} - \dfrac{V_{qs}}{\sigma L_s} \\ \dfrac{d\Omega}{dt} = \dfrac{P}{J}(1-\sigma)L_s\left(I_{qs}I'_{dr} - I_{ds}I'_{qr}\right) - \dfrac{fr}{J}\Omega \end{cases}$$

L'équation (2.20), montre que le fonctionnement de la machine dépend de quatre paramètres électriques [σ T_s L_s T_r] et de deux mécaniques [J f_r]. [25]

4. Technique d'identification

L'identification consiste à rechercher des modèles mathématiques de systèmes à partir de données expérimentales et de connaissances disponibles a priori [26]. Ces modèles doivent fournir une approximation fidèle du comportement du système physique sous-jacent dans le but d'estimer des paramètres physiques ou de concevoir des algorithmes de simulation, de prévision, de surveillance ou de commande. La démarche classique consiste à formaliser les connaissances disponibles a priori, à recueillir des données expérimentales, puis à estimer la structure, les paramètres et les incertitudes d'un modèle, enfin à valider (ou invalider) celui-ci. Ce champ thématique trouve ses applications dans des domaines très variés.

L'utilisation de données expérimentales pour déterminer les paramètres de modèles athématiques de systèmes dynamiques représente un sujet classique de l'automatique. Les modèles issus de l'identification sont appelés modèles boîte noire, lorsqu'ils sont peu, voire pas du tout, inspirés de la connaissance physique sur le procédé considéré. Durant les quatre dernières décennies, l'identification de modèles boîte noire a connu un développement considérable tant sur le plan de la proposition de nouvelles techniques que sur le plan des applications [27].

En effet, le formidable développement des calculateurs numériques a rendu l'utilisation des modèles à temps discret de plus en plus courante, non seulement en raison de la nature discrète des données acquises, mais surtout à cause de la facilité de l'implantation de l'algorithme d'identification puis de commande ou de diagnostic préventif développés à partir du modèle identifié.

Pour identifier un modèle à temps continu, deux approches principales sont envisageables dans le domaine temporel : l'approche directe et l'approche indirecte. Toutes deux utilisent des données d'entrée/sortie échantillonnées. L'approche indirecte consiste dans un premier temps, à déterminer un modèle à temps discret, puis à convertir ce dernier en un modèle à temps continu. Les propriétés statistiques de ces estimateurs (biais, variance, convergence) sont bien connues et expliquent en grande partie l'attrait pour cette stratégie. Cette approche indirecte fondée sur l'estimation initiale d'un modèle à temps discret fait, en général, appel à des algorithmes d'optimisation itératifs très coûteux en termes de calculs sans garantie de convergence vers l'optimum global. En effet, pour la plupart de ces algorithmes, la procédure d'initialisation conditionne la convergence vers l'optimum global. Les approches directes, ne souffrent pas de cet inconvénient.

Les techniques d'identification de modèles paramétriques linéaires à temps continu reposent principalement sur la minimisation d'un critère fondé soit sur une erreur de sortie, soit sur une erreur d'équation nécessitant l'utilisation d'une transformation linéaire couplée à une méthode issue des moindres carrés. De nombreuses méthodes de type erreur d'équation ont été proposées au cours des trente dernières années [28].

Dans ce contexte, on va essentiellement s'intéresser à l'identification des systèmes à l'aide de modèles à représentation continue. Deux catégories d'algorithmes sont utilisables, que l'on classe suivant la nature des résidus en erreur équation ou en erreur de sortie.

4.1. Définition d'un problème d'identification de paramètres

Soit M un vecteur de IR^{NbMes} contenant NbMes observations d'un système physique et F une application de IR^{Npar} dans IR^{NbMes}. On suppose que F est obtenu à

partir d'une modélisation du système physique observé. Le but est de trouver le jeu de paramètres λ appartenant à IR^{Npar} tel que $F(\lambda) = M$.

Pour identifier un modèle à temps continu, deux approches principales sont envisageables :

4.1.2. Identification de paramètres par analyse direct :

Dans les problèmes directs, les solutions analytiques ou numériques sont trouvées pour les équations différentielles ordinaires ou partielles avec des conditions initiales et aux limites connue et les constantes dans les équations connues. Le point important dans les problèmes directs, les variables dépendantes dans les équations sont calculés comme fonctions de temps et/ou position. Les mesures n'entrent pas dans la solution, exceptée peut-être pour fournir des conditions de la limite (Figure 1.9).

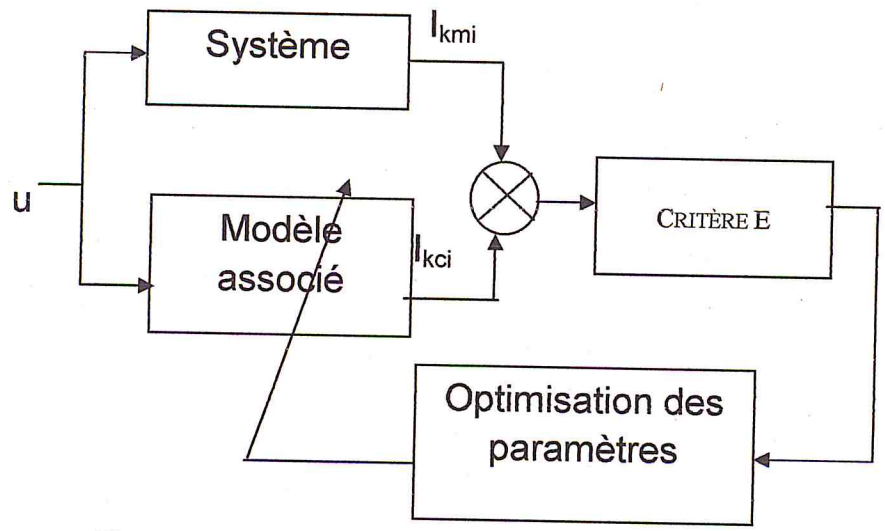


Figure 1.8 Principe d'identification direct

Cette méthode est aussi connue sous l'appellation « Méthode de l'erreur de sortie » ou de méthode du modèle. Elle repose sur la minimisation de l'énergie de l'erreur de sortie. Ce critère se distingue de celui fondé sur une erreur d'équation par sa signification physique et sa sensibilité plus importante aux erreurs de structure de modèles et aux valeurs des paramètres. Elle se différencie également par sa mise en œuvre plus délicate due au non linéarité de l'erreur de sortie par rapport aux

paramètres. Elle consiste à exploiter les propriétés locales du critère au voisinage d'un point courant dans l'espace des paramètres pour en déduire un point suivant meilleur au sens du critère à minimiser.

– Elle permet l'identification de systèmes multi entrées mono sortie représentés par une structure de modèles.

– Le problème principal des méthodes à erreur de sortie est l'existence possible de plusieurs minima locaux vers lesquels peut convergé l'algorithme d'optimisation. L'initialisation des algorithmes joue, en général, un rôle important sur la convergence et il est nécessaire de faire appel à des procédures particulières afin d'éviter les optimums locaux.

5. Conclusion

Dans ce chapitre on a développé un modèle proche de la réalité physique de la machine asynchrone et qui repose sur des paramètres réels. Les paramètres de ce modèle seront identifiés par la technique d'identification directe (erreur de sortie) dans le chapitre suivant.

Chapitre2 :

Les méthodes d'optimisation

II. LES METHODE D'OPTIMISATION

1. Introduction

Les ingénieurs se sont trouvés face à des problèmes à la taille et à la complexité croissante, ce qui fut une motivation pour la recherche de méthodes de résolution fiables et systématiques. La plupart des problèmes à résoudre peut souvent exprimer comme un problème d'optimisation. De ce fait, l'optimisation est une branche des mathématiques qui permet de résoudre des problèmes en déterminant le meilleur élément d'un ensemble selon certains critères prédéfinis.

Dans ce chapitre nous présentant tout d'abord des notions liées aux problèmes et aux méthodes d'optimisation suivant le domaine d'application continu , et en suite en présentant la nouvelle méta heuristique « optimisation par essaim particulaire ». [1]

1.1. Définition d'un problème d'optimisation

Un problème d'optimisation est défini par un ensemble de variables, une fonction objective (fonction de coût) et un ensemble de contraintes. L'espace d'état, appelé aussi domaine de recherche, est l'ensemble des domaines de définition des différentes variables du problème.

La fonction objective définit le but à atteindre, on cherche à minimiser ou à maximiser celle-ci. L'ensemble des contraintes est en général un ensemble d'égalités ou d'inégalités que les variables de l'espace d'état doivent satisfaire. Ces contraintes limitent l'espace de recherche.

Les méthodes d'optimisation recherchent un point ou un ensemble de points dans l'espace de recherche qui satisfont l'ensemble des contraintes, et qui maximisent ou minimisent la fonction objective. [2]

1.2. Classes de complexité

Les classes de complexité sont définies initialement dans le cadre des problèmes de décision plutôt que dans celui des problèmes d'optimisation.

Tout d'abord, on fait une distinction entre les problèmes décidables et les problèmes indécidables sont ceux pour lesquels aucun algorithme, quel qu'il soit, n'a

été trouvé pour les résoudre ; ainsi, les problèmes décidables sont ceux pour lesquels il existe au moins un algorithme pour les résoudre. [3]

1.3. Classification des problèmes d'optimisation

En distingue deux types de problèmes d'optimisation : les problèmes « discrets » et les problèmes à variables continues. Citons un exemple de chaque type, pour fixer les idées. Parmi les problèmes discrets, on trouve le célèbre problème du voyageur de commerce : il s'agit de minimiser la longueur de la tournée d'un « voyageur de commerce », qui doit visiter un certain nombre de villes, avant de retourner à la ville de départ. Dans la catégorie des problèmes continus, un exemple classique est celui de la recherche des valeurs à affecter aux paramètres d'un modèle numérique de processus, pour que ce modèle reproduise au mieux le comportement réel observé. En pratique, on rencontre aussi des « problèmes mixtes », qui comportent à la fois des variables discrètes et des variables continues. [5]

1.4. Les méthodes approchées(les heuristiques)

Une méthode heuristique (du verbe grec heuriskein, qui signifie « trouver ») permet de guider le processus dans sa recherche des solutions optimales. Feignebaum et Feldman (1963) définissent une heuristique comme une règle d'estimation, une stratégie, une astuce, une simplification, ou toute autre sorte de système qui limite drastiquement la recherche des solutions dans l'espace des configurations possibles. Newell, Shaw et Simon (1957) précisent qu'un processus heuristique peut résoudre un problème donné, et ne peut pas être généralisée à un ensemble de problèmes [10].

1.5. Les méta heuristiques

15.1. Définition

Les métaheuristiques sont apparues au début des années 1980 avec une ambition commune : résoudre au mieux les problèmes dits d'optimisation difficile. En effet, celles-ci s'appliquent à toutes sortes de problèmes discrets, et elles peuvent s'adapter aussi aux problèmes continus.

Les métaheuristiques sont généralement des algorithmes stochastiques itératifs, qui progressent vers un optimum global, c'est-à-dire l'extremum global d'une fonction, par échantillonnage d'une fonction objectif. [11]

1.5.2. Les méthodes de méta-heuristiques

Il existe un grand nombre de méta-heuristiques déférentes, allant des méta-heuristiques à solution unique à des méta-heuristiques à base de population.

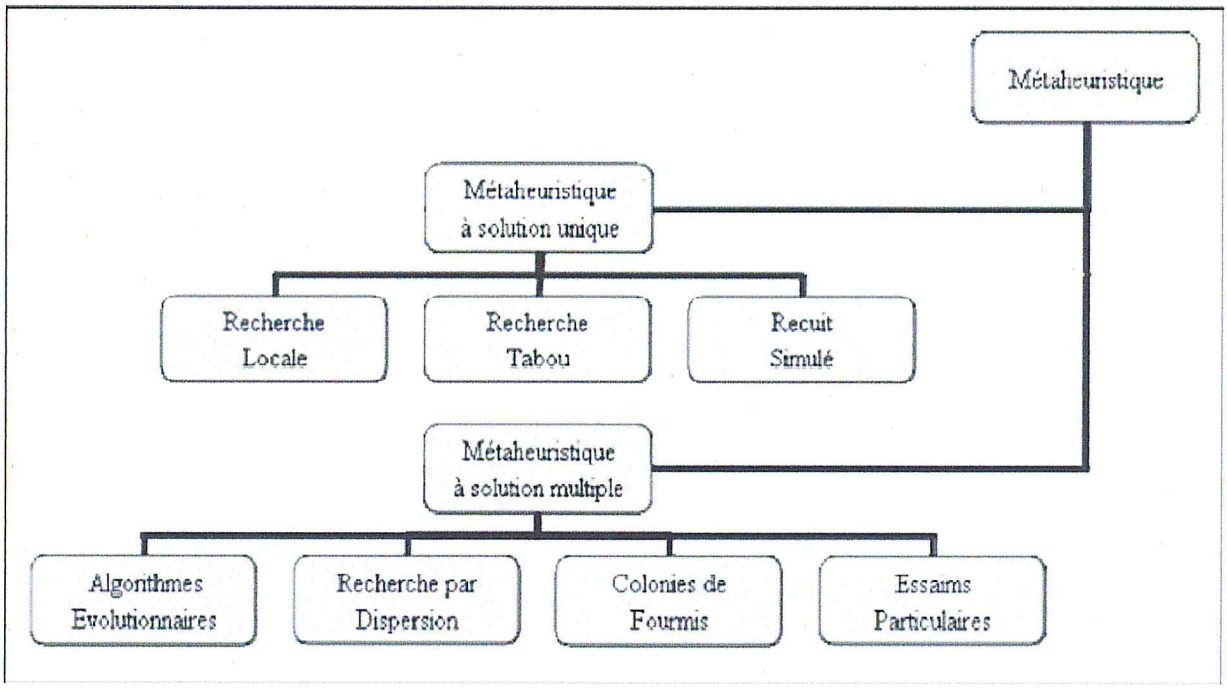


Figure2.1 : Classification des méta-heuristiques.

1.5.2.1. Méta-heuristique à solution unique

Fondés sur la recherche locale de solution, ces algorithmes itératifs procèdent par la génération d'une solution initiale de manière aléatoire .Ils essayent ensuite l'améliorer d'avantage en se basant seulement sur les meilleures solutions se trouvant autour de son voisinage.

1.5.2.2 Méta-heuristique à base de population

Améliorent au fur et à mesure des itérations une population des solutions. Ces algorithmes appelés aussi méthodes évolutionnaires, présentent l'avantage d'être doté d'une certaine intelligence.

Les Algorithmes Evolutionnaires. Comprend principalement, en plus des Algorithmes Génétiques, les Stratégies d'évolution (Evolution Stratégies), la programmation Evolutionnaire (Evolutionnary Programming) et la Programmation Génétique (Genetic Programming) [17].

1.6. Optimisation par essaims particulaires

1.6.1. Introduction

Parmi les méta-heuristiques on trouve la descente stochastique c'est la plus courante, on part d'une solution initiale, on la compare à tous ses voisins en conservant à chaque fois le meilleur résultat.

L'optimisation par essaim particulaire, qui dérive de la descente stochastique, entre dans cette famille d'algorithmes. Elle s'inspire fortement des relations grégaires des oiseaux migrateurs qui doivent parcourir des longues distances et qui doivent donc optimiser leurs déplacements en termes d'énergie dépensée, comme par exemple la formation en V.

Figure2.2 : Voler en formation en V

1.6.2. Origines

L'optimisation par essaims particulaires (OEP ou PSO particle swar optimisation) est une méta-heuristique d'optimisation, inventée par Russel Eberhart (ingénieur en électricité) et James Kennedy (socio-psychologue) en 1995 cherchaient à modéliser des interactions sociales entre des « agents » devant atteindre un objectif donné dans un espace de recherche commun, chaque agent ayant une certaine capacité de mémorisation et de traitement de l'information. La règle de base était qu'il ne devait y avoir aucun chef d'orchestre, ni même aucune connaissance par les agents de l'ensemble des informations, seulement des connaissances locales. Un modèle simple fut alors élaboré. Dès les premières simulations, le comportement collectif de ces agents évoquait celui d'un essaim d'êtres vivants convergeant parfois en plusieurs sous essaims vers des sites intéressants. Ce comportement se retrouve dans bien d'autres modèles, explicitement

inspirés des systèmes naturels. Ici, la métaphore la plus pertinente est probablement celle de l'essaim d'abeilles, particulièrement du fait qu'une abeille ayant trouvé un site prometteur sait en informer certaines de ses consœurs et que celles-ci vont tenir compte de cette information pour leur prochain déplacement. le modèle s'est révélé être trop simple pour vraiment simuler un comportement social, mais par contre très efficace en tant qu'outil d'optimisation.

Fonctionnement fait qu'elle peut être rangée dans les méthodes itératives (on approche peu à peu de la solution) et stochastiques (on fait appel au hasard). Sous ce terme un peu technique, on retrouve un comportement qui est aussi vieux que la vie elle-même: améliorer sa situation en se déplaçant partiellement au hasard et partiellement selon des règles prédéfinies.[32]

1.6.3. Les éléments de P.S.O

Pour appliquer la PSO il faut définir un espace de recherche constitué de particules et une fonction objective à optimiser. Le principe de l'algorithme est de déplacer ces particules afin qu'elles, trouvent l'optimum.

Chacune de ces particules est dotée :

- D'une position, c'est-à-dire ses coordonnées dans l'ensemble de définition.
- D'une vitesse qui permet à la particule de se déplacer. De cette façon, au cours des itérations, chaque particule change de position. Elle évolue en fonction de son meilleur voisin, de sa meilleure position, et de sa position précédente. C'est cette évolution qui permet de tomber sur une particule optimale.
- D'un voisinage, c'est-à-dire un ensemble de particules qui interagissent directement sur la particule, en particulier celle qui a le meilleur critère.

A tout instant, chaque particule connait :

-Sa meilleure position visitée. On retient essentiellement la valeur du critère calculée ainsi que ses coordonnées.

- La position du meilleur voisin de l'essaim qui correspond à l'ordonnancement optimal.

- La valeur qu'elle donne à la fonction objectif car à chaque itération il faut une comparaison entre la valeur du critère donnée par la particule courante et la valeur optimale.

1.6.4. Principe

Le principe est considère dans l'espace de recherche, un essaim de particules.

Un essaim est composé d'un ensemble de particules qui représentent chacune une solution. Chaque particule est en train de bouger, c'est-à-dire qu'elle a une vitesse. Egalement, chaque particule a une petite mémoire, lui permettant de se souvenir de sa meilleure performance, en position et en valeur.

Enfin, chaque particule dispose d'un groupe d'informatrices, historiquement appelé son voisinage.

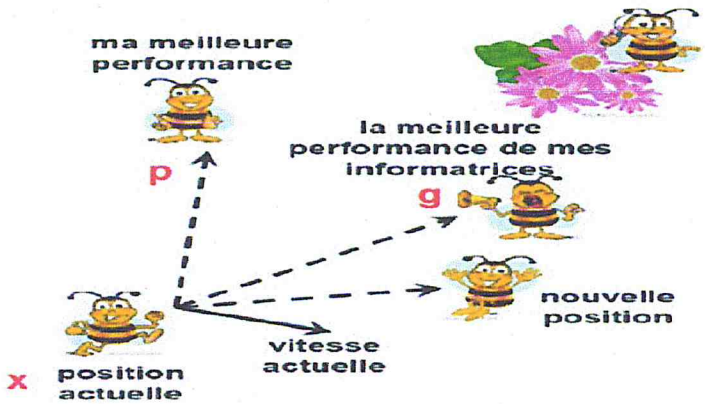


Figure2.3 : principe de PSO

A partir des quelques informations dont elle dispose, une particule doit décider de son prochain mouvement, c'est-à-dire décider de sa nouvelle vitesse.

Pour ce faire, elle combine linéairement trois informations :

- sa vitesse actuelle
- sa meilleure performance
- la meilleure performance de ses voisines (ses informatrices).

À l'aide de trois paramètres parfois appelés coefficients de confiance, qui pondèrent trois tendances :

- tendance à suivre sa propre voie.

- tendance conservatrice (revenir sur ses pas).
- tendance « panurgienne » (suivre le meilleur voisin).[33]

1.6.5. Qualité d'une particule

Chaque particule a une position courante et une « meilleure performance », qui est mémorisée. C'est donc d'abord à ce niveau de détail que l'on peut dire s'il y a progrès ou non.une particule sera dite bonne si elle vient d'améliorer sa meilleure performance, neutre sinon.et celle qui a la moins performance. Elle sera appelée la pire.

On peut voir chaque particule de l'essaim comme le sommet d'un graphe, on peut représenter le lien d'information par un arc de B ver A. L'arc inverse, de A ver B, peut exister [13].

1.6.6. Suppression d'une particule

Le but est de trouver l'optimum, si possible a moindre faires, c'est-à-dire en effectuant le moins possible d'évaluations de la fonction.par des que l'occasion se présente de supprimer a peu prés sans particule, il faut la saisir. Notons qu'il vaut mieux conserver a tort une particule qu'en éliminer une a tort. C'est pourquoi seule une bonne tribu pourra éventuellement éliminer une de ses particules et uniquement la pire d'entre elle .Dans le cas d'une tribu mono particule, l'élimination ne se fera que si un des informateurs a une meilleure performance .En effet, on veut être sur de conserver au moins une information de meilleure qualité que celle que l'on va éliminer

1.6.7. Les caractéristiques

- Caractéristique d'un essaim particule est:
  - N : le nombre de particule
  - V_max : la vitesse maximale d'une particule ;
  - C1 :l'inertie d'une particule ;
  - C2, C3 : les coefficients de confiance qui pondère le comportement conservateur et le panurgisme.
- Caractéristique d'une particule :
  - Xi(t) : sa position dans l'espace de recherche.

- Vit(t) : sa citasse.
- X_pbesti : la position de la meilleure solution par laquelle elle est passée.
- X_vbesti :la position de la meilleure solution connue de son voisinage.
- Pbesti :la valeur de fitness de sa meilleur solution.
- Vbesti.la valeur de fitness de la meilleure solution connue du voisinage.

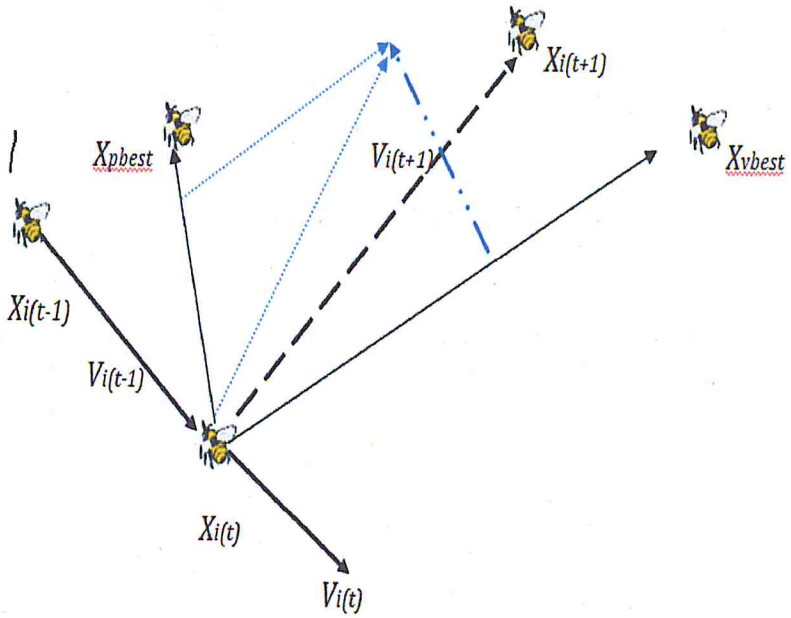


Figure2.4 : le déplacement des particules

Dans les faits, on calcule la nouvelle vitesse à partir de la formule suivante :

$$Vi(t+1)=c1* Vi(t) +C2 * rand*(Xpbest - Xi) +C3*rand*(Xvbest - Xi)$$

On peut ensuite déterminer la position suivante de la particule grâce à la vitesse que l'on vient de calculer :

$$Xi(t+1)=Xi(t)+V(t+1)$$

1.6.8. Configuration de la méthode

- **Coefficients de confiance :**

Deux autres paramètres importants sont les coefficients de confiance que l'on nomme précédemment c2 et c3. Ils permettent de pondérer les tendances des particules à suivre leur instinct de conservation ou leur panurgisme. De manière générale, ces variables aléatoires sont évaluées à chaque itération suivant une loi uniforme sur le domaine de définition.

C2 ET c3 sont des constantes positives terminées de façon empirique et suivant la relation :

$$C2+C3 \leq 4$$

- **Le coefficient d'inertie :**

De même un paramètre important à prendre en compte est le coefficient d'inertie appelé c1 dans la formule vue auparavant. Il permet de définir la capacité d'exploration de chaque particule en vue d'améliorer la convergence de la méthode. Fixer ce paramètre revient à trouver un compromis entre une exploration globale (c1 > 1) et une exploration locale (c1 < 1). Il représente l'instinct aventureux de la particule.

- **Topologie du voisinage :**

Le voisinage d'une particule est le sous-ensemble de particules de l'essaim avec lequel il a une communication directe. Ce réseau de rapports entre toutes les particules est connu comme la sociométrie, ou la topologie de l'essaim.

Il existe deux principaux types de voisinage :

✓ **Les voisinages géographiques :** les voisins sont considérés comme les particules les plus proches. Cependant, à chaque itération, les nouveaux voisins doivent être recalculés à partir d'une distance prédéfinie dans l'espace de recherche. C'est donc un voisinage dynamique.

Dans cet exemple, le voisinage de la particule est composé des deux particules les plus proches.

- ✓ **Les voisinages sociaux :** les voisins sont définis à l'initialisation et ne sont pas modifiés ensuite. C'est le voisinage le plus utilisé, pour plusieurs raisons :
- Il est plus simple à programmer.
- Il est moins coûteux en temps de calcul.

En cas de convergence, un voisinage social tend à devenir un voisinage géographique [14]

– Topologie en étoile : le réseau social est complet, chaque particule est attirée vers la meilleure particule notée gbest et communique avec les autres.

– Topologie en anneau: chaque particule communique avec n

(n = 3)voisines immédiates. Chaque particule tend à se déplacer vers la meilleure dans son voisinage local notée lbest.

– Topologie en rayon: une particule "centrale" est connectée à tous les autres. Seule cette particule centrale ajuste sa position vers la meilleure, si cela provoque une amélioration l'information est propagée aux autres.

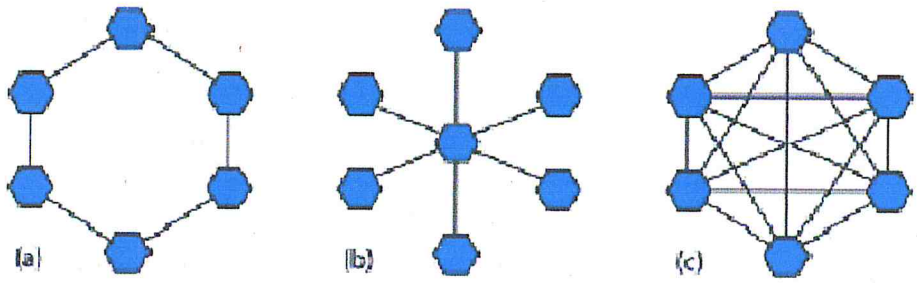


Figure2.5 : Trois topologies différentes

1.6.9. Organigramme de la méthode des essaims particulaires

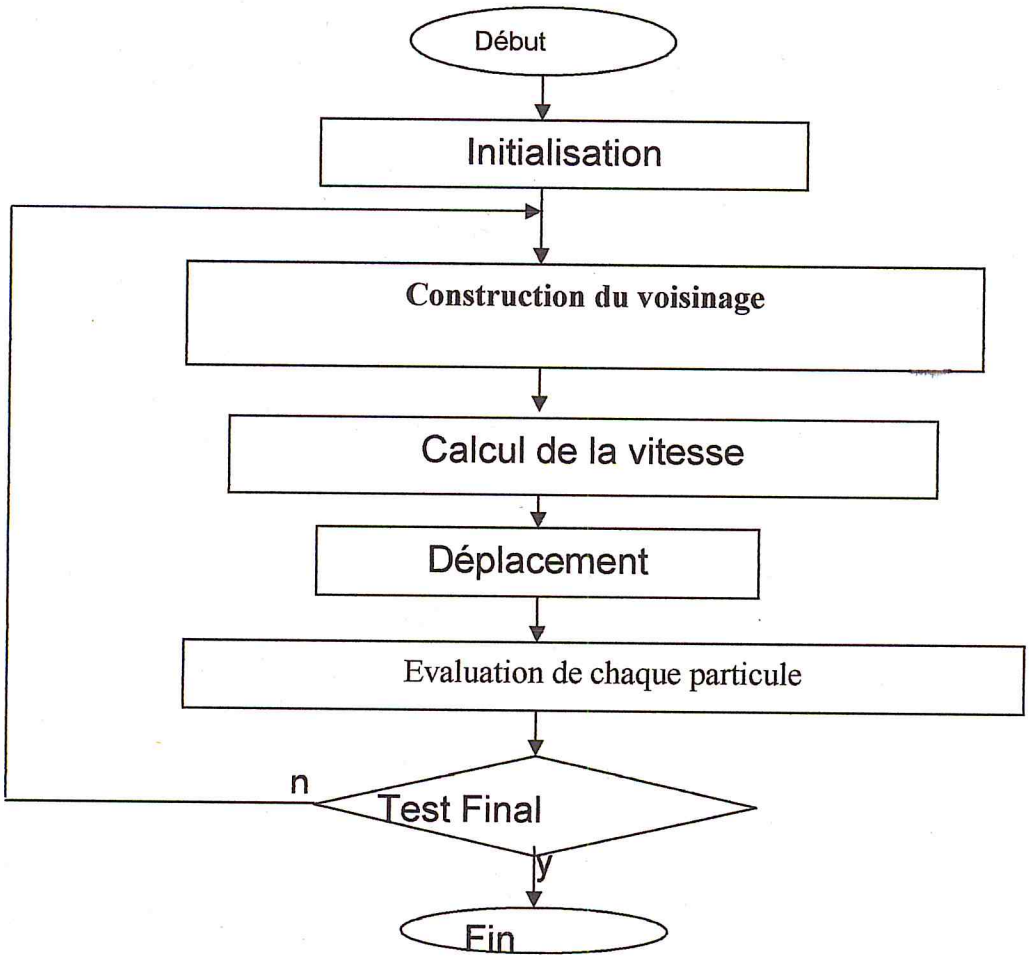


Figure 2.6 : Algorithme du PSO standard

7. Conclusion

Dans ce chapitre, nous avons présenté les différentes méthodes d'optimisation avec une classification suivant les problèmes traités et suivant les techniques de recherche.

Nous avons étudié les différentes des métas heuristiques dans la littérature du domaine d'optimisation, avec une étude détaillée de la nouvelle méta heuristique nommé : PSO

Chapitre 3 :

Conception

III.CONCEPTION

1. Introduction

Dans ce chapitre nous présentons la modélisation UML (*Unified Modeling Language).*

UML est un langage de modélisation graphique. Il est apparu dans le monde de la conception et du développement des systèmes informatiques, principalement dans le cadre de la «conception orientée objet ».

Aujourd'hui, UML est naturellement utilisé dans les projets logiciels, mais on peut l'appliquer pour spécifier toutes sortes de systèmes sans se limiter au domaine informatique.

2. Modélisation UML

La modélisation consiste à créer une représentation simplifiée d'un problème, la modélisation comporte deux composantes :

- L'analyse, c'est-a-dire l'étude du problème ;
- La conception, soit la mise au point d'une solution au problème.

3. Objectifs d'UML

Dans le cadre spécifique de la création d'un langage commun à divers processus de développement de projets, UML a été conçu pour répondre aux objectifs suivants :

- comprendre des problèmes ;
- spécifié des modèles ;
- construire des solutions.

Par extension, dans une approche plus large, UML insiste sur la modélisation et satisfait aux exigences suivantes :

- définir un ensemble commun d'éléments de modélisation indépendant des domaines d'application, c'est-à-dire permettant de modéliser facilement toutes sortes de systèmes, mêmes non logiciels ;
- fournir aux utilisateurs un langage de modélisation fondamentalement basé sur l'approche objet ;
- fournir un langage graphique pour décrire des modèles.

4. Différentes vues d'une modélisation UML

Toutes les vues proposées par UML sont complémentaires les unes des autres, elles permettent de mettre en évidence différents aspects d'un logiciel à réaliser. On peut organiser une présentation d'UML autour d'un découpage en vues, ou bien en différents diagrammes, nous commençons par présenter les différentes vues, qui sont les suivantes :

1- la vue fonctionnelle, interactive, qui est représentée à l'aide de diagrammes de cas et de diagrammes des séquences.

2 - la vue structurelle, ou statique, réunit les diagrammes de classes et les diagrammes de packages.

3 - la vue dynamique, montrant le fonctionnement du système qui est exprimée par les diagrammes d'états, et diagramme d'activité.

5. Diagramme

5.1. Définition des besoins

Les besoins de notre système sont définis dans les objectives : Optimisé les paramètres de la machine asynchrone avec une implémentation parallèle de la méthode PSO.

5.2. Diagramme de cas d'utilisation

Le diagramme de cas d'utilisation représente la structure des grandes fonctionnalités nécessaires aux utilisateurs du système.

C'est le premier diagramme du modèle UML, celui où s'assure-la relation entre l'utilisateur et les objets que le système met en œuvre.

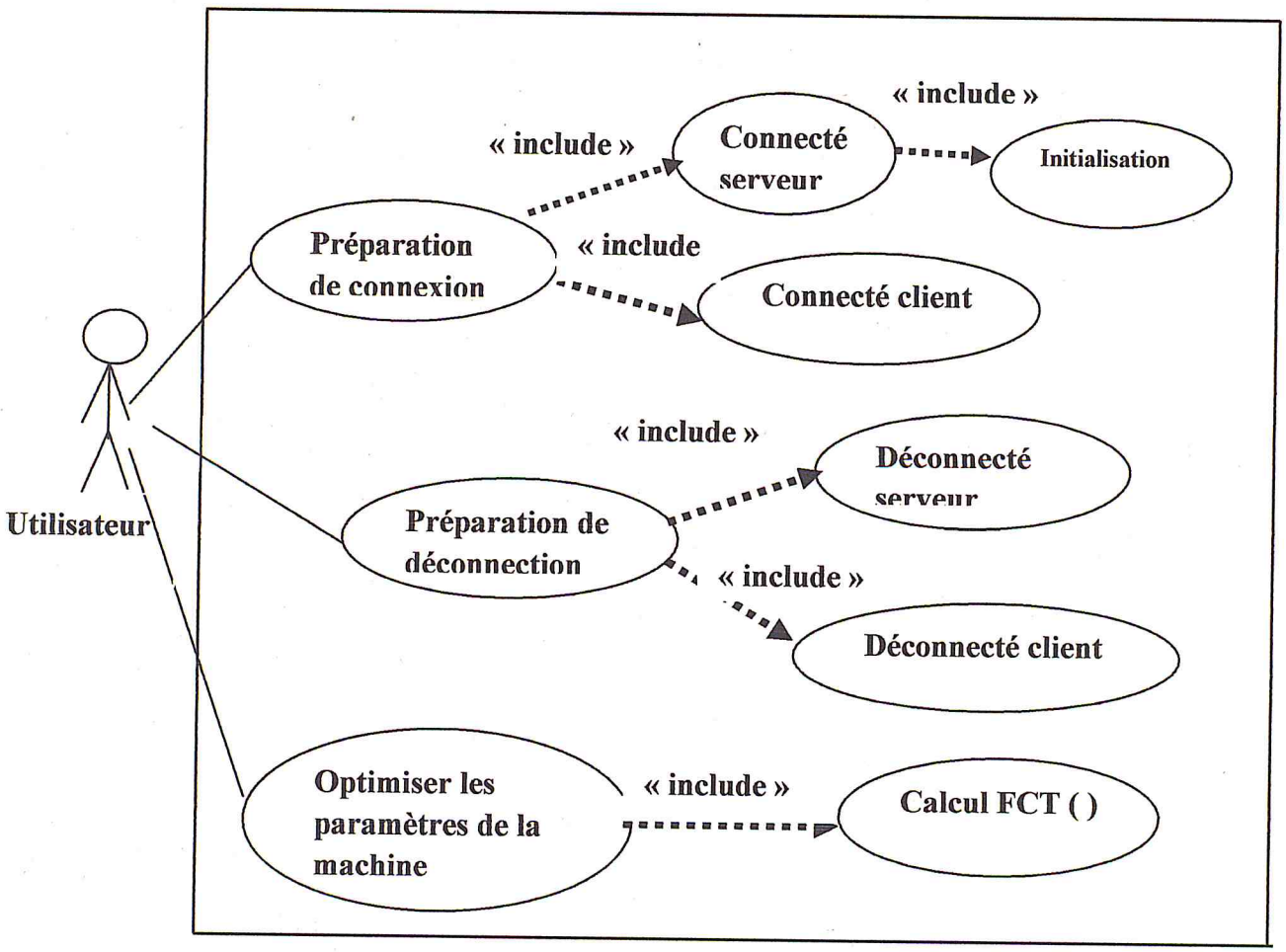


Figure3.1 : Diagramme de cas d'utilisation global.

5.3. Diagramme de séquence

Le diagramme de séquence représente la succession chronologique des opérations réalisées par un acteur. Il indique les objets que l'acteur va manipuler et les opérations qui font passer d'un objet à l'autre.

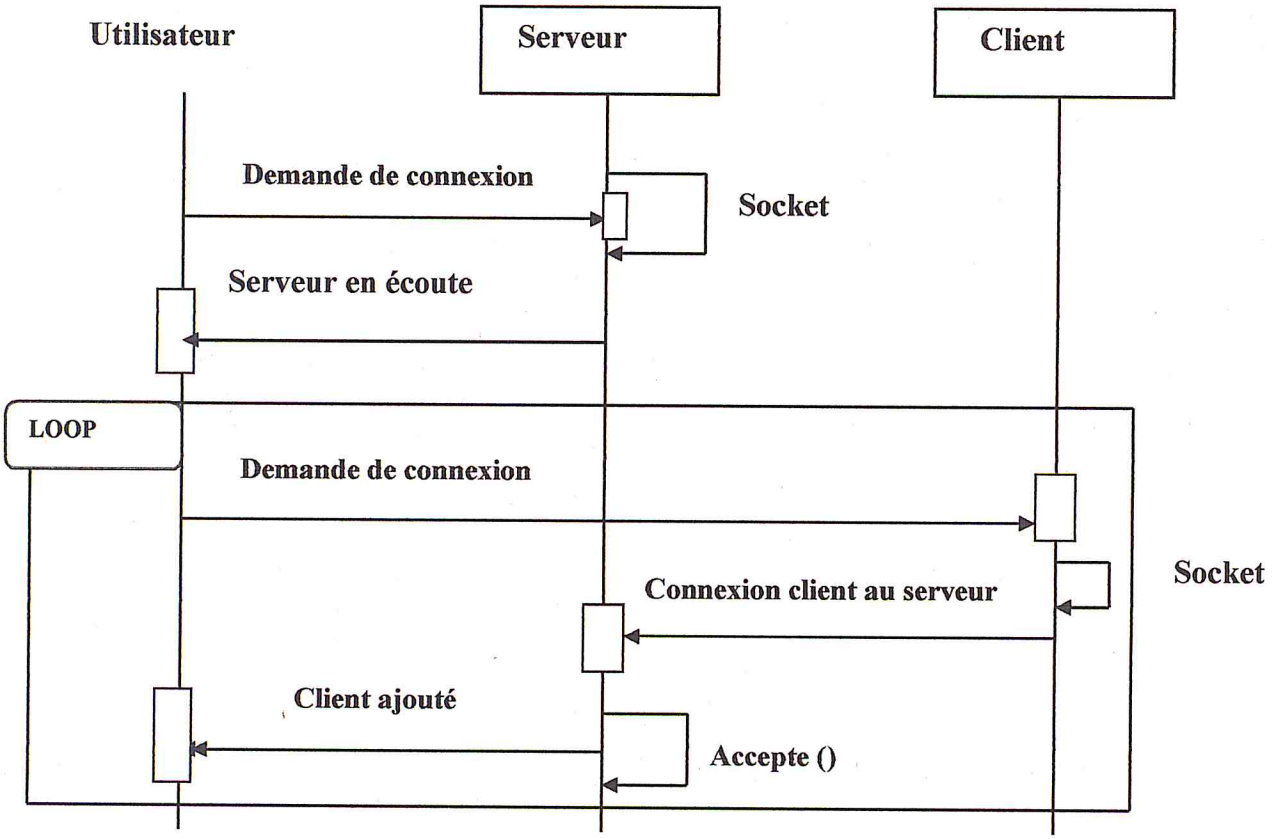


Figure3.2 : Diagramme de séquence pour la préparation de connexion

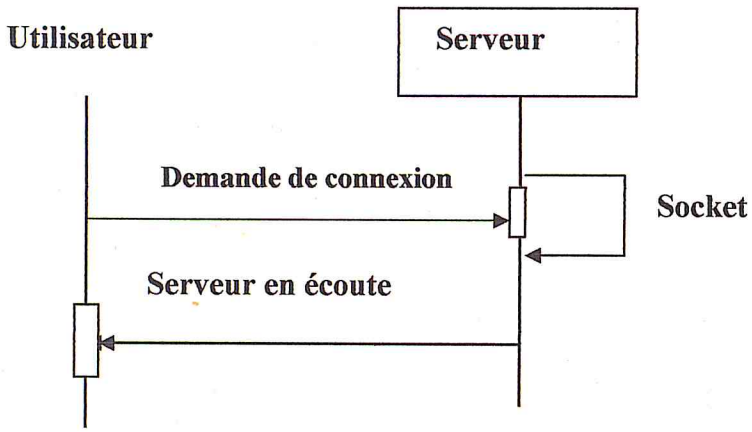


Figure3.3 : Diagramme de séquence pour la préparation de connexion2

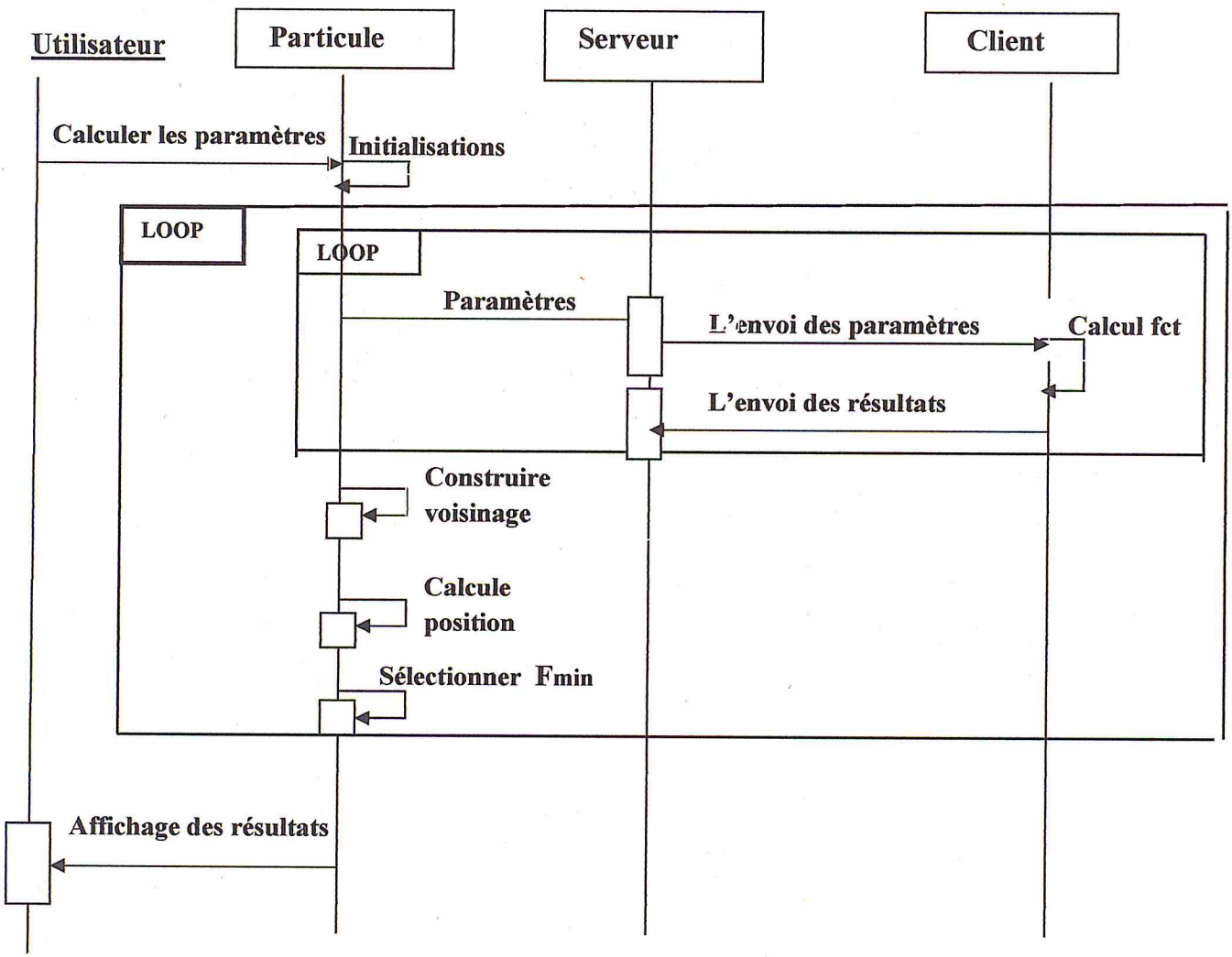


Figure3.4 : Diagramme de séquence pour PSO1

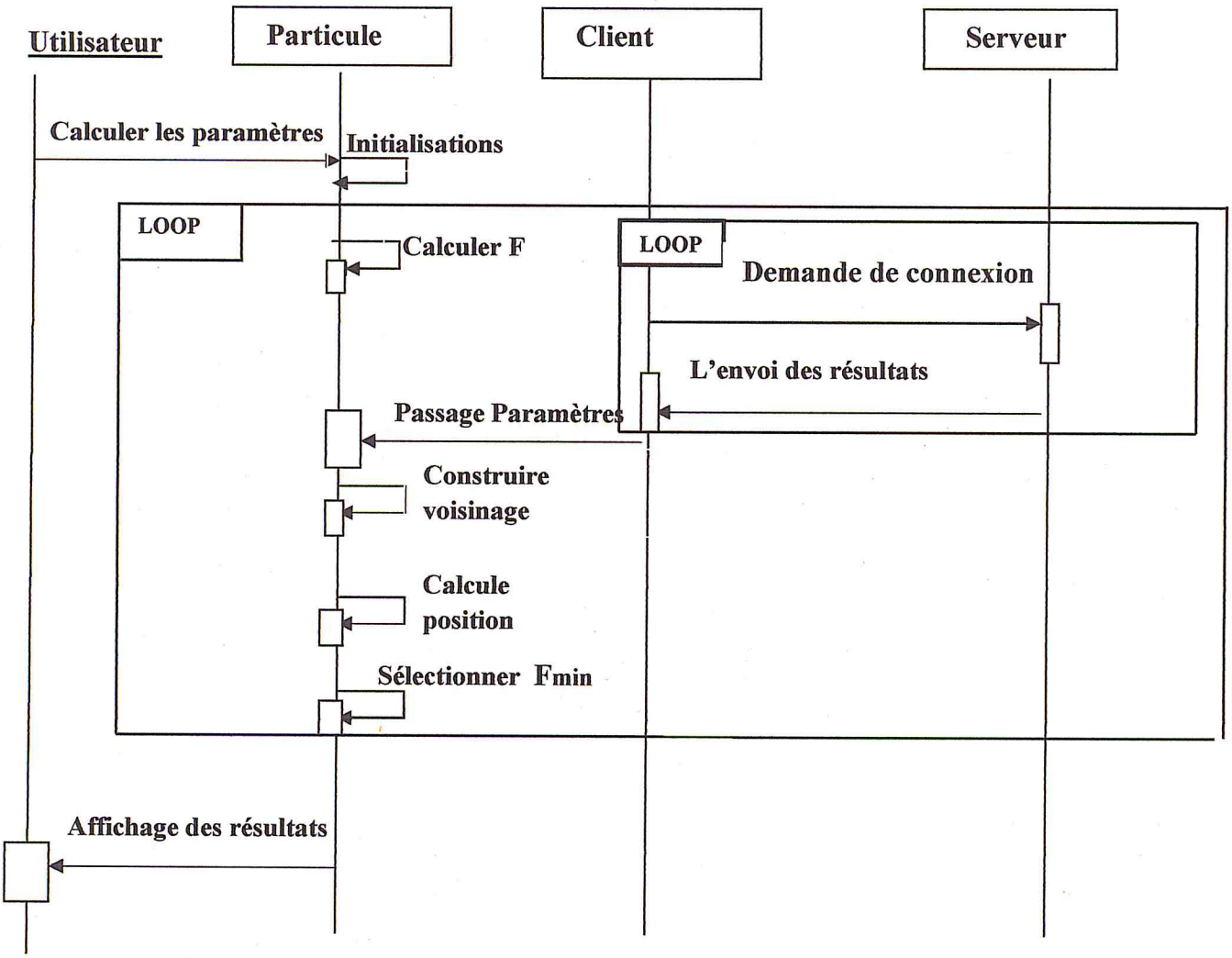


Figure3.5 : Diagramme de séquence pour PSO2

5.4. Diagramme de classes

Le diagramme de classes est généralement considéré comme le plus important dans un développement orienté objet. Il représente l'architecture conceptuelle du système : il décrit les classes que le système utilise, ainsi que leurs liens.

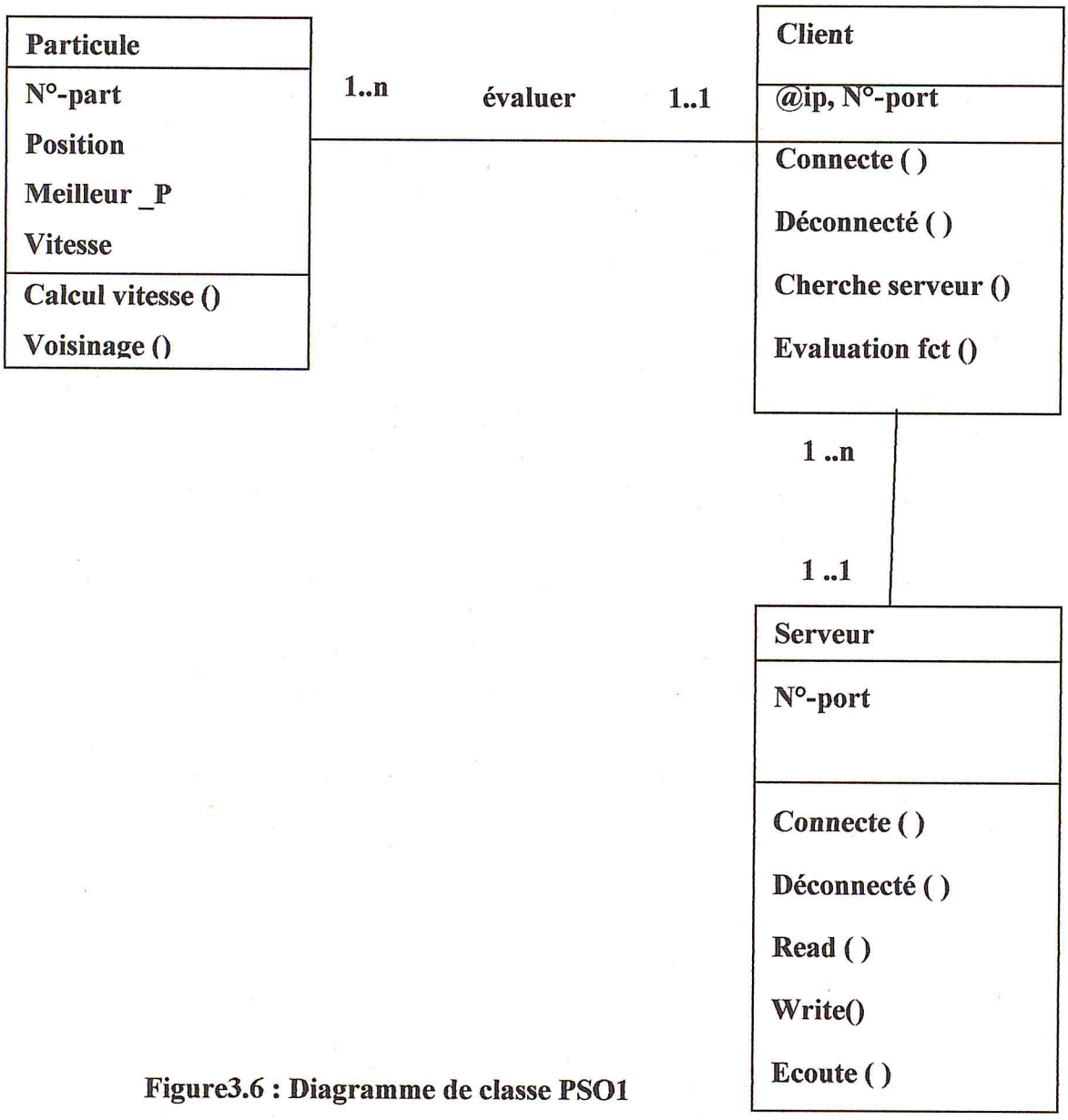


Figure3.6 : Diagramme de classe PSO1

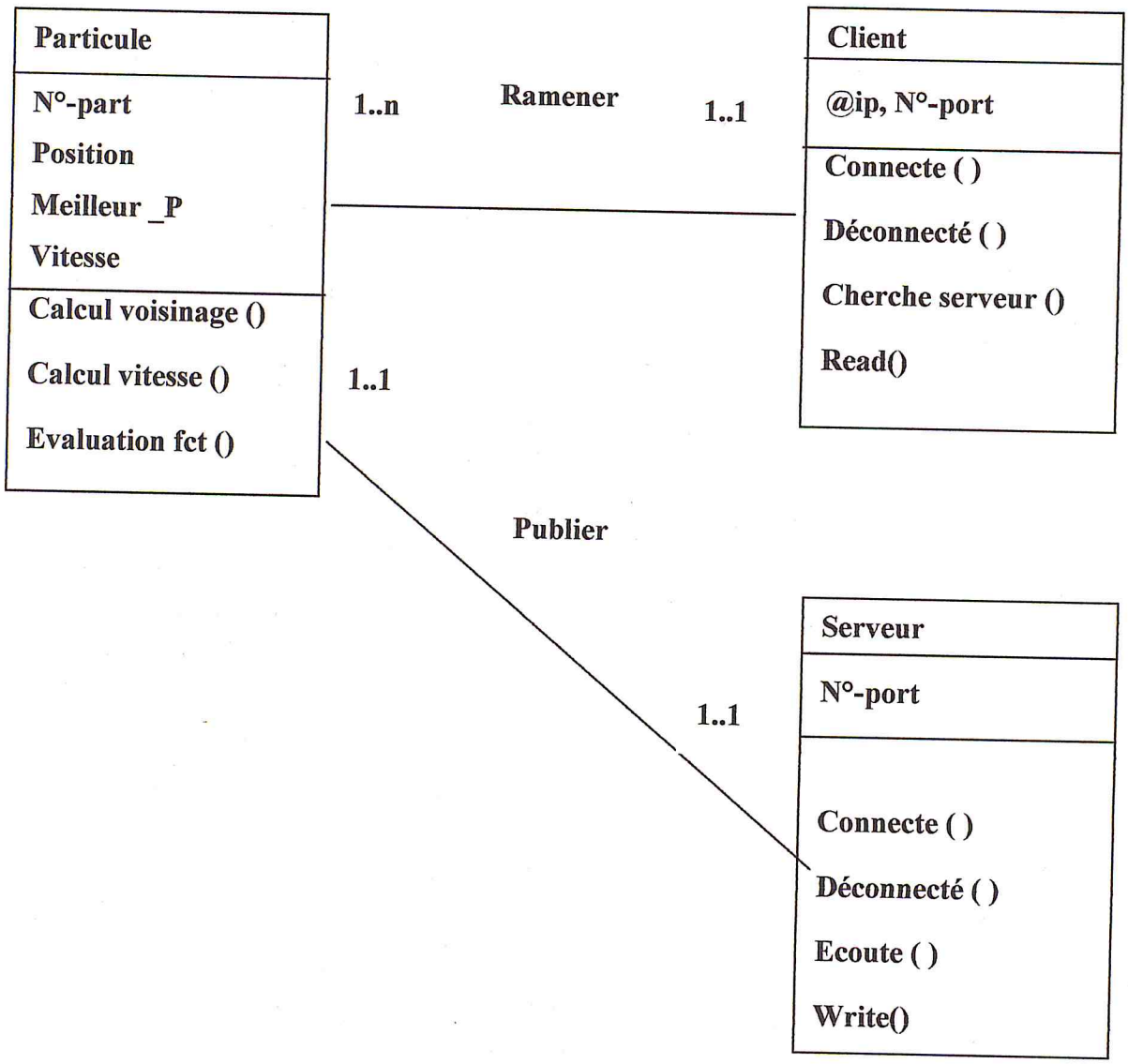


Figure3.7 : Diagramme de classe PSO2

5.5. Diagramme de déploiement

Un diagramme de déploiement décrit la disposition physique des ressources matérielles qui composent le système et montre la répartition des composants sur ces matériels. Chaque ressource étant matérialisée par un nœud, le diagramme de déploiement précise comment les composants sont répartis sur les nœuds et quelles sont les connexions entre les composants ou les nœuds.

Description :

- Pour la 1er architecteur on doit installer un réseau qui contient un certain nombre de poste suivant le nombre des particules.
  - ➢ Un poste est considéré comme serveur pour la gestion central des particules à savoir : position, vitesse, déplacement, voisinage et meilleure record.
  - ➢ Les autres postes comme des clients, où chaque client réalise la tache d'évaluation de la fonction objective pour une particule.

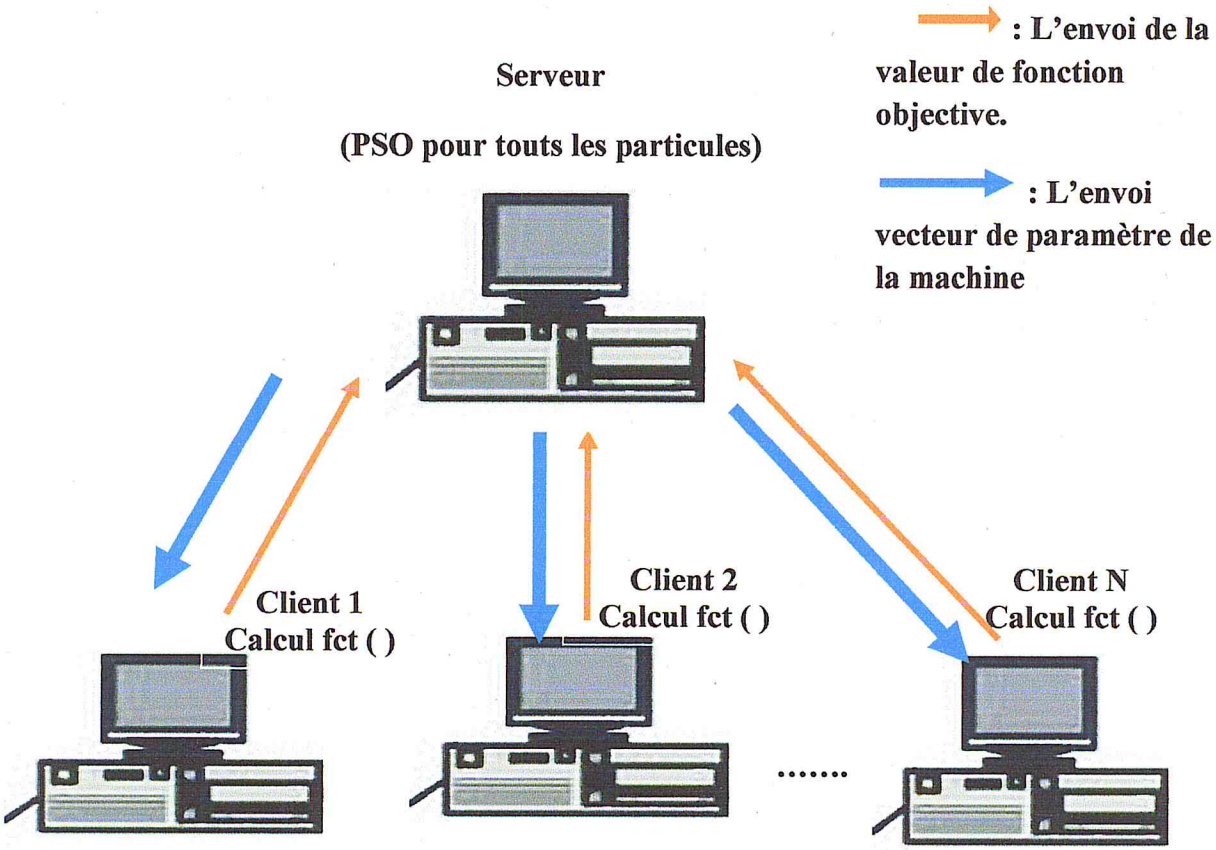


Figure 3.8 : schéma de description 1

A tout instant après lancement du calcul, un client peut s'ajouté ou se retiré du group sans que la méthode s'initialise. Le serveur accepte touts les connections et transfère les informations à la classe particule. Les particules peuvent évoluer sans rendre compte qu'une particule est ajoutée ou supprimé.

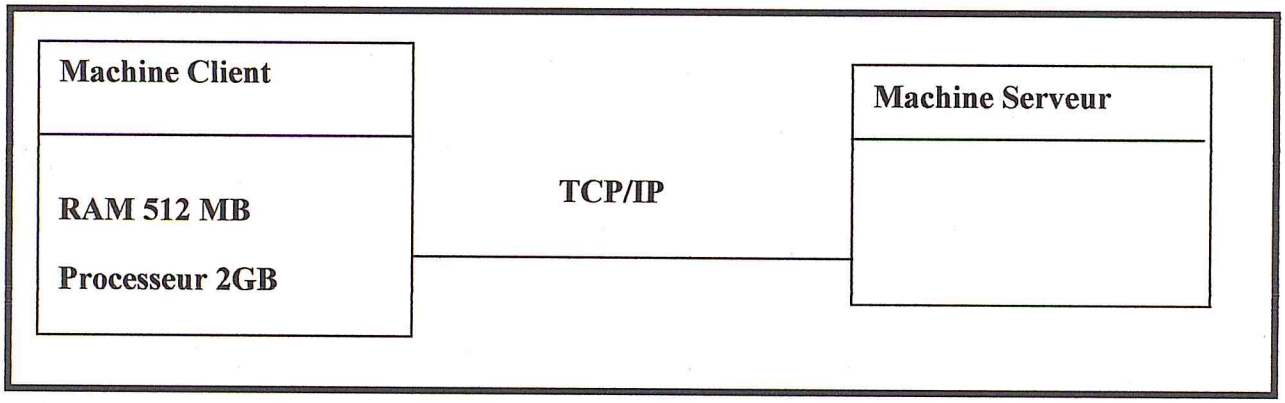


Figure 3.9 : Diagramme de déploiement

- **Pour la 2eme architecture :**
  - Chaque poste est considéré comme une particule complètement autonome en déplacement et en évaluation. Donc chaque poste possède une partie serveur pour informé les autres et une partie client pour demander les informations. Cette architecture va nous permettre de tester le déplacement asynchrone des particules.

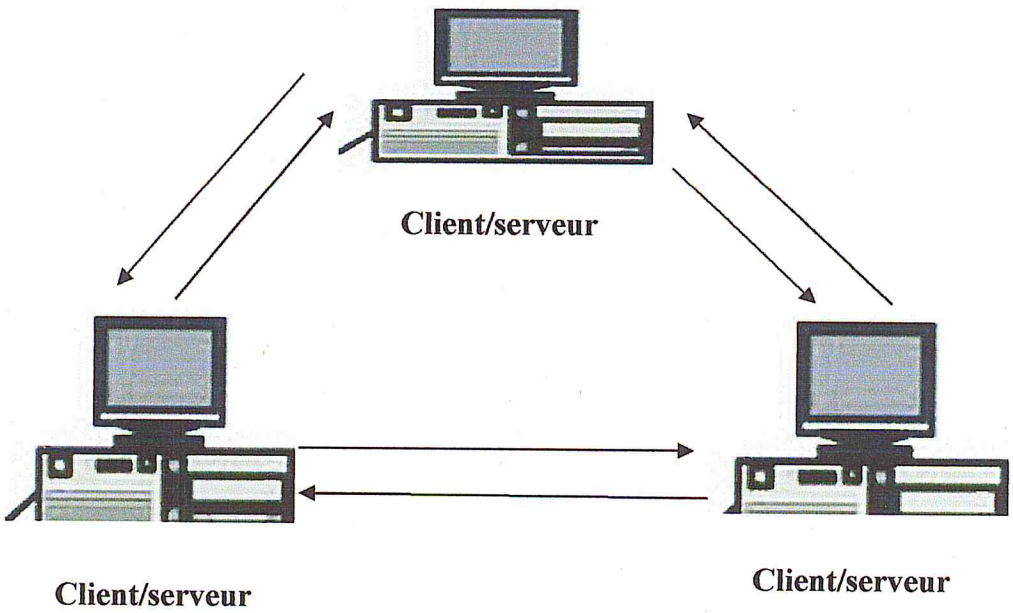


Figure 3.10 : schéma de description 2

6. Conclusion

On a présenté dans ce chapitre le langage UM L et nous avons modélisé nos traitements .Grace a notre conceptions nous pouvons entamer la partie réalisation de notre projet avec teste et implémentation.

Chapitre4 :

TEST

ET

IMPLEMENTATION

IV.CHAPITRE : TEST ET IMPLEMENTATION

1. Introduction

Ce dernier chapitre est consacré à la présentation de l'application, cette application a été développée en c++ sous l'environnement builder c++. Elle permet de définir les paramètres de la machine asynchrone.

Nous commençons par la présentation de l'interface et ces différentes fonctionnalités.

Et nous procédons par la suite aux différents tests effectués.

2. l'environnement c++ builder 5

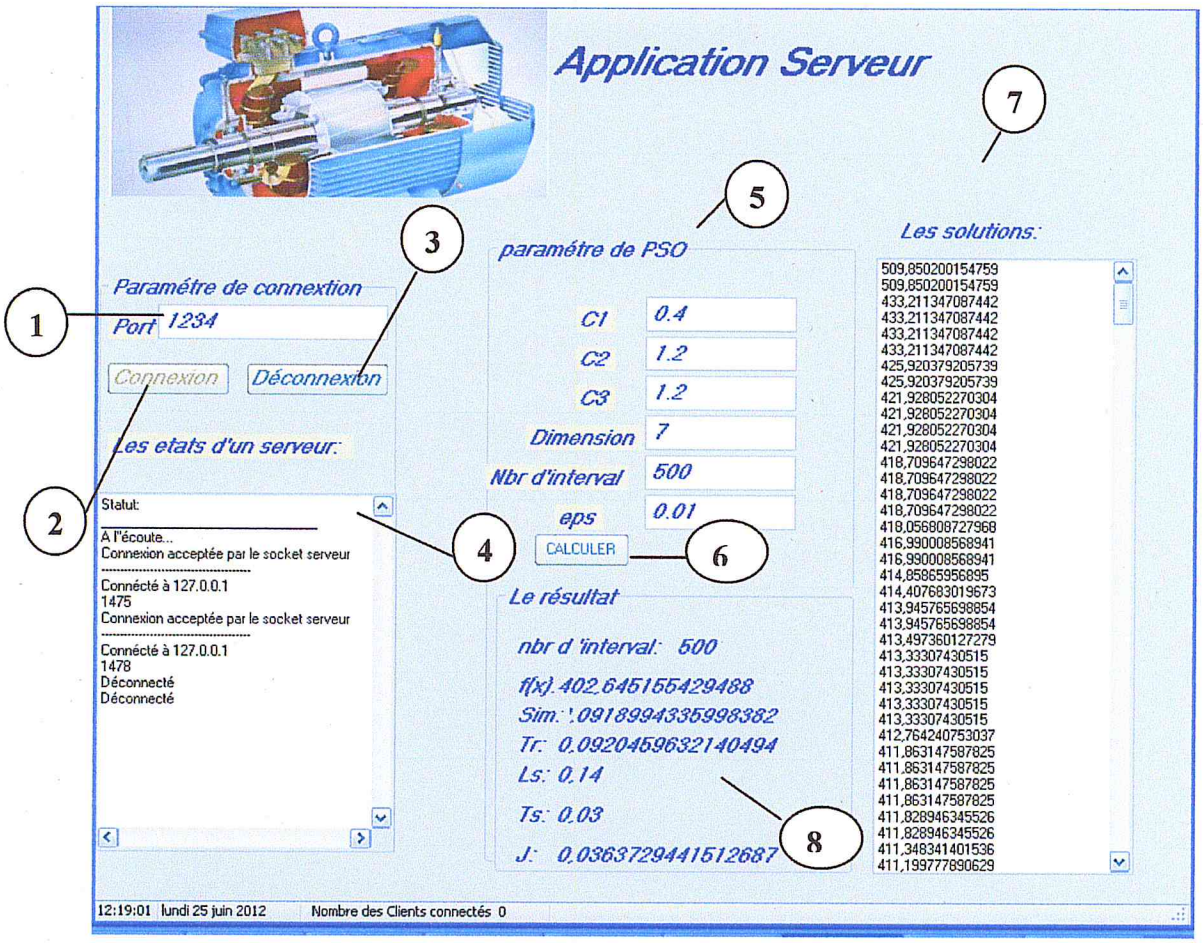
C++builder5 est un environnement de programmation visuel orienté objet pour le développement rapide d'applications.il offre une série de composant qui permettent la réalisation des applications interactif, et la partie fonctionnelle d'application exemple réseau.

3. Interface

3.1. Interface pour la 1er architecture

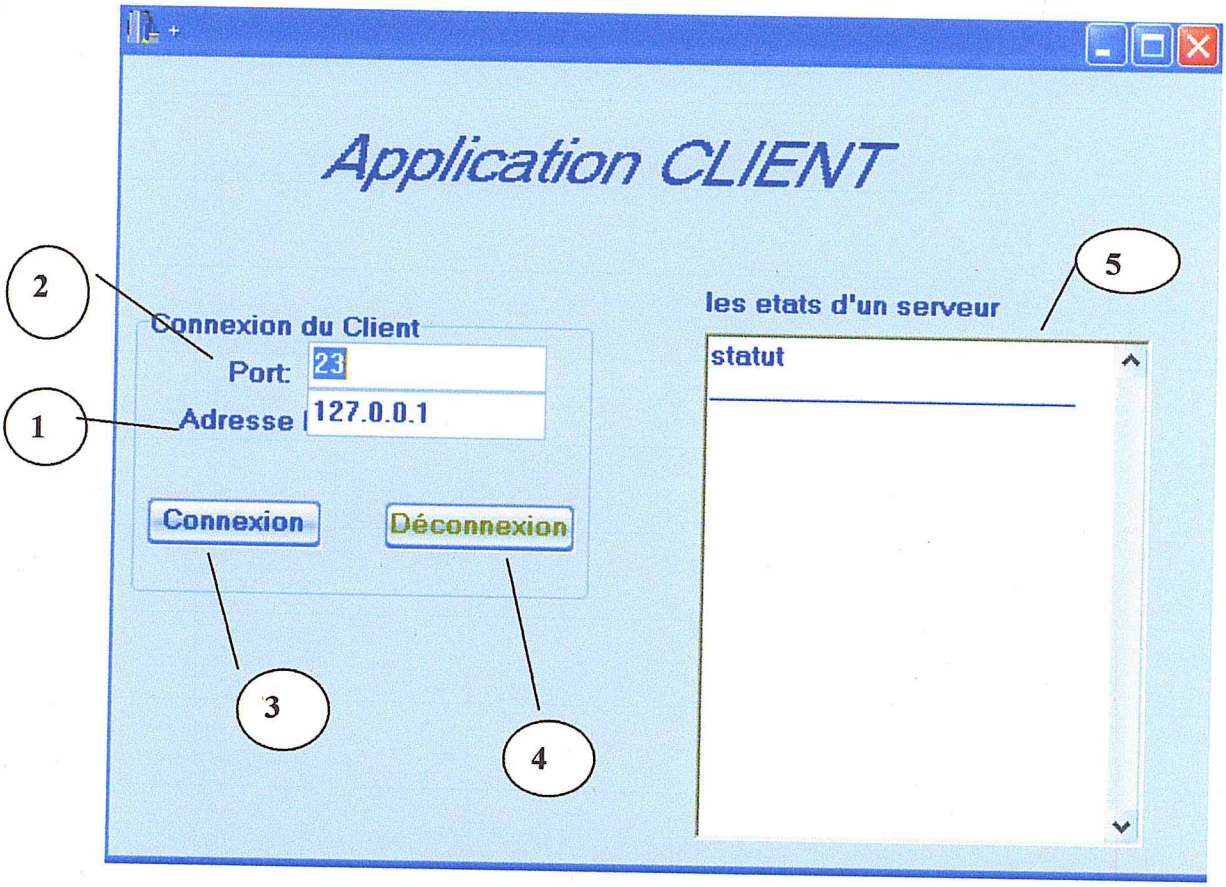
Pour cette architecture on a une interface pour l'application serveur et une interface pour l'application client.

- **L'interface serveur :**



1 : numéro de port de serveur.
2 : connexion de serveur pour le mettre en écoute.
3 : déconnexion de serveur
4 : le statut de serveur et l'affichage des clients connecté.
5 : les paramètres de PSO.
6 : lancement de calcule.
7 : l'affichage du meilleur f.
8 : l'affichage du meilleur résultat obtenu.

- **l'interface client :**



1 : saisie de l'@ IP de serveur.

2 : saisie de numéro de port de serveur.

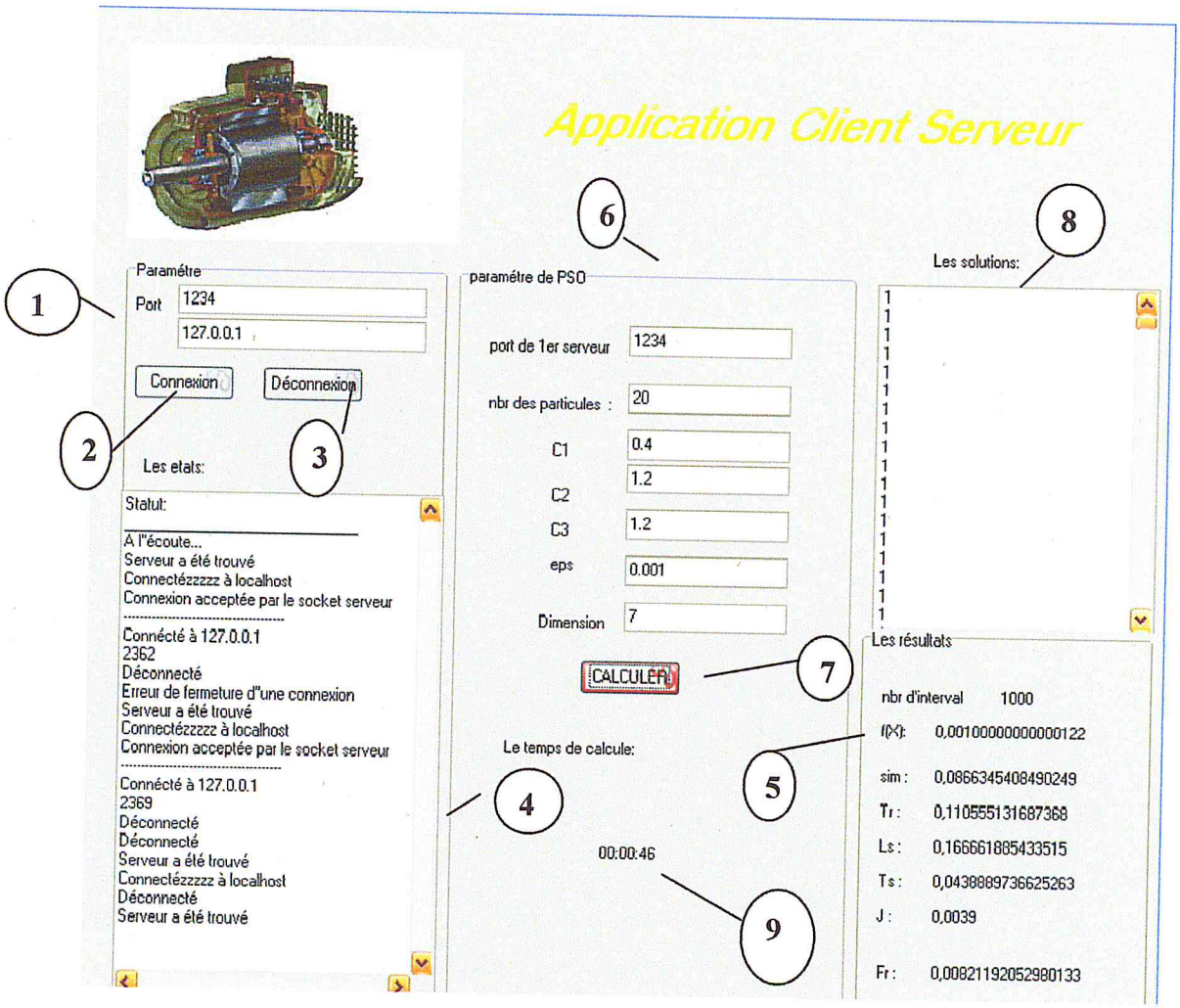
3 : connexion de client.

4 : déconnexion de client.

5 : l'état de client (connecté, déconnecté).

3.2. Interface pour la 2eme architecture

Pour cette architecture on a une seule interface qui contient des composants serveur et des composants client.



1 : paramètre de connexion.
2 : connexion.
3 : déconnexion.
4 : statut
5 : meilleur résultat obtenu.
6 : paramètre de PSO.
7 : lancement de calcule.
8 : l'affichage du meilleur résultat obtenu.
9 : le temps de calcule.

4. Validation du modèle

Pour valider le modèle mathématique on procède par l'injection des paramètres d'un moteur connu et on compare le courant calculé avec le courant mesuré et on peut déduire l'écart entre mm avec des suppositions de simplification et le modèle réel

5. Configuration de PSO

Avant l'identification des paramètres de la machine on doit d'abord choisir les bons paramètres de PSO à savoir les coefficients de l'équation de vitesse (c1, c2, c3) et la meilleur structure de voisinage, donc pour cela on doit effectuer plusieurs teste avec des valeurs simulé.

5.1. Changement de c1, c2, c3

Les facteurs c1, c2, c3 joue un rôle très important pour la convergence de la méthode PSO. Après plusieurs tests on a abouti que :

c1=1.2-0.6*(nb_eval/eval_max)

c2=c3=0.4.

5.2. Confinement d'intervalle

C'est le comportement d'une particule au cour de son déplacement dans un sens et dé qu'elle arrive à la limite de l'intervalle elle doit changé de sens et avec une décélération de la vitesse.

Après certain nombre de teste, on a opté pour la formule suivante :

V[n]= c1*V[n]/ 2;

5.3. Le choix de voisinage

Le choix de cet algorithme (l'algorithme à deux structures de voisinage) provient de faite que cette dernière basée sur une recherche globale et une recherche locale :

- La recherche globale est présentée lorsque chaque particule demande information aléatoirement des autres particules de son voisinage formé.

- La recherche locale est présentée lorsque chaque particule demande d'information aux particules les plus proches.

6. Tests et résultats

6.1. Résultat simulé

Dans ce tableau, en a illustré les paramètres de la machine asynchrone :

Paramètre	valeur
Sigma	0.09
Tr	0.123
Ls	0.159
Ts	0.054
J	0.038
Fr	0.008

Figure 4.1 : Paramètre de simulation

A l'aide de données simulées nous nous proposons de:

- Valider les programmes développés.
- Montrer que la connaissance du courant et de la tension est suffisante pour déterminer simultanément les paramètres électriques et mécaniques caractérisant la machine.
- Montrer que la fonction représentant l'erreur quadratique entre les grandeurs simulé et les grandeurs calculées présente un minimum global, en d'autres termes, que le vecteur paramètre représentant le minimum de cette fonction est unique.

Les données simulées sont obtenues par résolution numérique du système décrivant le fonctionnement de la machine) par la méthode de Runge-Kutta en utilisant les paramètres d'un moteur asynchrone triphasé connu.

Les données sont obtenues sont présenté dans figure 4.2

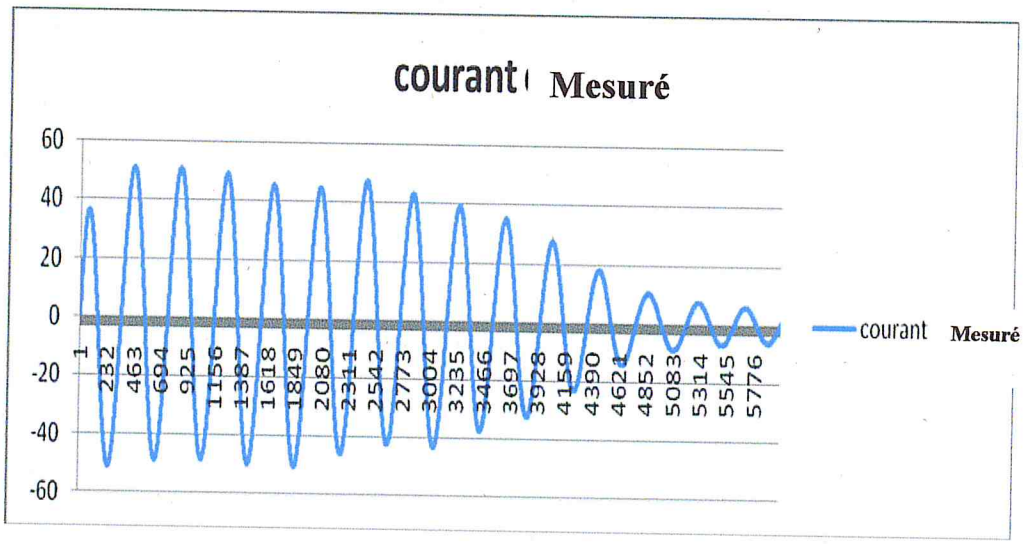


Figure 4.2 : Courant obtenu par les Paramètres de simulation

6.2. Résultat calculé

Pour la confirmation de la validation de notre travail on a entamé plusieurs tests.

6.2.1. Tests

En utilisant les données simulées illustrées par la figure 4.1 nous estimons les paramètres du moteurs a l'aides de deux architecteurs, pour chaque architecteurs nous avons effectue plusieurs tests sur un espace de recherche. Nous présentons l'évolution des paramètres en fonction des itérations.

1er architecture :

- **Nombre d'itération=300**

Nbr client	Fmin	sigma	Tr	Ls	Ts	j	Fr	temps
1	**1596.82**	**0.11**	**0.11**	**0.15**	**0.049**	**0.033**	**0.004**	**00 :00 :05**
3	**103.47**	**0.086**	**0.133**	**0.15**	**0.049**	**0.037**	**0.007**	**00 :00 :11**
7	**5.76**	**0.091**	**0.119**	**0.155**	**0.05**	**0.038**	**0.007**	**00 :00 :28**
19	**4.66**	**0.091**	**0.121**	**0.156**	**0.053**	**0.037**	**0.008**	**00 :01 :06**
25	**3,84**	**0.09**	**0.123**	**0.157**	**0.053**	**0.038**	**0.01**	**00 :01 :27**
40	**4.1**	**0.089**	**0.123**	**0.159**	**0.054**	**0.037**	**0.008**	**00 :02 :18**
80	**0.8**	**0.087**	**0.122**	**0.157**	**0.053**	**0.038**	**0.008**	**00 :15 :00**

Figure 4.3 : résultat1 selon l'évolution du nombre de particule à l'itération 300

- **Nombre d'itération=500**

Nbr client	Fmin	sigma	Tr	Ls	Ts	j	Fr	temps
1	**115.48**	**0.1**	**0.12**	**0.15**	**0.049**	**0.033**	**0.003**	**00 :00 :25**
3	**103.47**	**0.086**	**0.133**	**0.15**	**0.045**	**0.038**	**0.007**	**00 :00 :28**
7	**2.88**	**0.09**	**0.121**	**0.157**	**0.53**	**0.038**	**0.008**	**00 :01 :28**
19	**1.55**	**0.09**	**0.12**	**0.158**	**0.053**	**0.037**	**0.007**	**00 :01 :56**
25	**1,63**	**0.09**	**0.123**	**0.158**	**0.053**	**0.037**	**0.008**	**00 :02 :27**
40	**0.68**	**0.09**	**0.122**	**0.158**	**0.053**	**0.037**	**0.007**	**00 :03 :18**
80	**0.3**	**0.087**	**0.122**	**0.157**	**0.053**	**0.038**	**0.008**	**00 :05 :00**

Figure 4.4 : résultat1 selon l'évolution du nombre de particule à l'itération 500

- **Nombre d'itération=1500**

Nbr client	Fmin	sigma	Tr	Ls	Ts	j	Fr	temps
1	**63.06**	**0.1**	**0.11**	**0.15**	**0.049**	**0.037**	**0.003**	**00 :00 :45**
3	**48.66**	**0.093**	**0.136**	**0.15**	**0.045**	**0.033**	**0.003**	**00 :02 :28**
7	**0.31**	**0.09**	**0.121**	**0.158**	**0.53**	**0.037**	**0.003**	**00 :05 :28**
19	**0.069**	**0.09**	**0.122**	**0.158**	**0.053**	**0.038**	**0.007**	**00 :08:56**
25	**0.15**	**0.09**	**0.123**	**0.158**	**0.053**	**0.037**	**0.008**	**00 :15 :27**
44	**0.89**	**0.09**	**0.122**	**0.158**	**0.053**	**0.037**	**0.007**	**00 :20 :18**
80	**0.22**	**0.09**	**0.122**	**0.157**	**0.053**	**0.038**	**0.008**	**00 :22 :00**

Figure 4.5 : résultat1 selon l'évolution du nombre de particule à l'itération 1500

2eme architecture :

- **Nombre d'itération=300**

Nbr client	Fmin	sigma	Tr	Ls	Ts	j	Fr	Temps
1	**583.69**	**0.01**	**0.12**	**0.15**	**0.048**	**0.033**	**0.004**	**00 :00 :03**
3	**99.20**	**0.086**	**0.133**	**0.156**	**0.050**	**0.037**	**0.007**	**00 :00 :09**
7	**2.89**	**0.091**	**0.119**	**0.155**	**0.053**	**0.038**	**0.007**	**00 :00 :21**
19	**1.21**	**0.091**	**0.121**	**0.156**	**0.052**	**0.038**	**0.008**	**00 :00:57**
25	**0.8**	**0.09**	**0.122**	**0.157**	**0.053**	**0.038**	**0.01**	**00 :01 :20**
40	**0.6**	**0.089**	**0.122**	**0.159**	**0.054**	**0.037**	**0.008**	**00 :02 :00**
80	**0.2**	**0.09**	**0.122**	**0.159**	**0.054**	**0.038**	**0.008**	**00 :14:00**

Figure 4.6 : résultat2 selon l'évolution du nombre de particule à l'itération 300

- **Nombre d'itération=500**

Nbr client	Fmin	sigma	Tr	Ls	Ts	j	Fr	Temps
1	**106.30**	**0.1**	**0.12**	**0.15**	**0.049**	**0.033**	**0.003**	**00 :00 :15**
3	**80.45**	**0.086**	**0.133**	**0.15**	**0.045**	**0.038**	**0.007**	**00 :00 :19**
7	**1.59**	**0.09**	**0.121**	**0.157**	**0.53**	**0.038**	**0.008**	**00 :01 :07**
19	**0.9**	**0.09**	**0.122**	**0.158**	**0.053**	**0.037**	**0.007**	**00 :01 :43**
25	**0.4**	**0.09**	**0.123**	**0.158**	**0.053**	**0.037**	**0.008**	**00 :02 :03**
40	**0.13**	**0.09**	**0.122**	**0.158**	**0.053**	**0.037**	**0.007**	**00 :03 :20**
80	**0.10**	**0.09**	**0.122**	**0.159**	**0.053**	**0.038**	**0.008**	**00 :18 :00**

Figure 4.7 : résultat2 selon l'évolution du nombre de particule à l'itération 500

- **Nombre d'itération=1500**

Nbr client	Fmin	sigma	Tr	Ls	Ts	j	Fr	Temps
1	**40.06**	**0.1**	**0.122**	**0.150**	**0.049**	**0.037**	**0.003**	**00 :00 :20**
3	**32.66**	**0.093**	**0.136**	**0.15**	**0.045**	**0.032**	**0.006**	**00 :01 : 30**
7	**0.20**	**0.09**	**0.122**	**0.158**	**0.53**	**0.037**	**0.007**	**00 :03:45**
19	**0.019**	**0.09**	**0.122**	**0.158**	**0.053**	**0.038**	**0.008**	**00 :08:00**
25	**0.016**	**0.09**	**0.123**	**0.158**	**0.053**	**0.038**	**0.008**	**00 :15 :20**
44	**0.010**	**0.09**	**0.122**	**0.158**	**0.054**	**0.038**	**0.007**	**00 :20 :30**
80	**0.001**	**0.09**	**0.123**	**0.159**	**0.054**	**0.038**	**0.008**	**00 :25 :59**

Figure 4.8: résultat2 selon l'évolution du nombre de particule à l'itération 1500

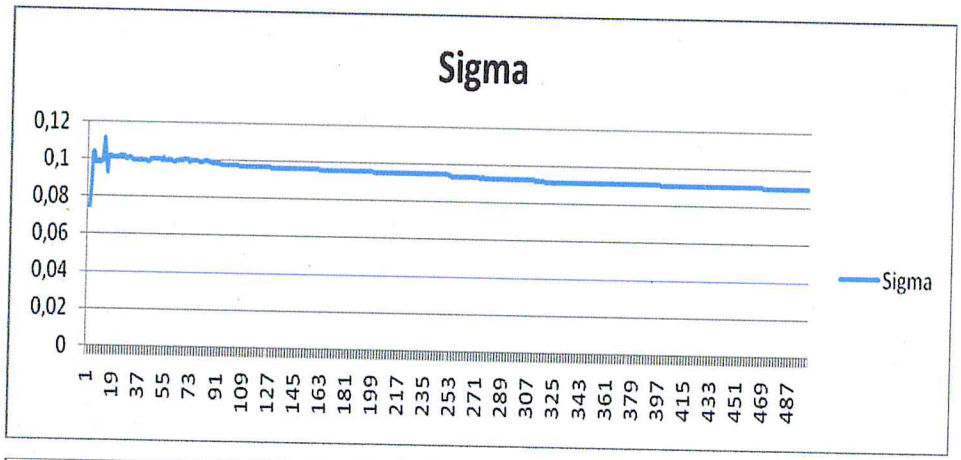
6.2.2. Interprétation des résultats

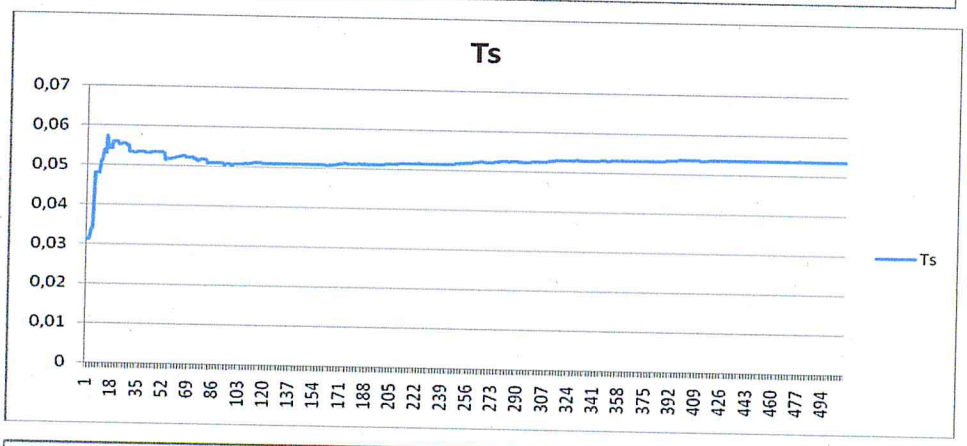
Après les résultats illustrés dans les tableaux précédentes en conclus :

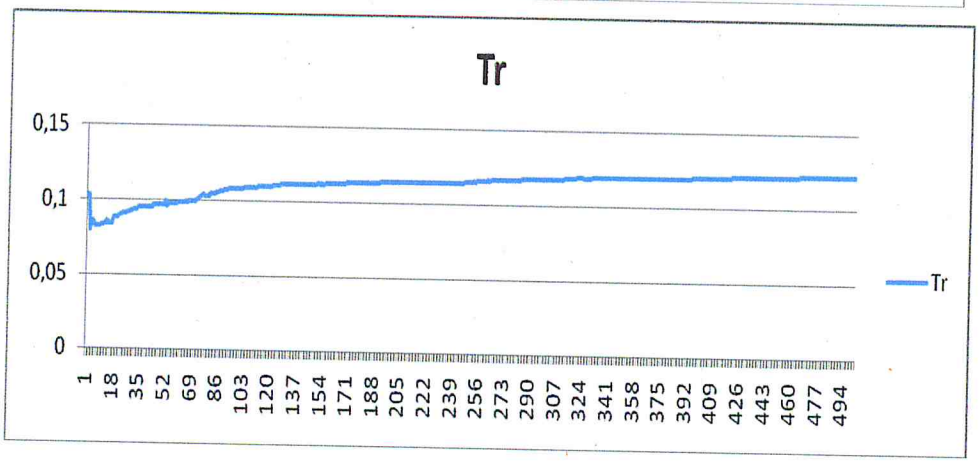
Quand le nombre de particule est grand on atteindre généralement l'optimum et que l'augmentation de nombre d'itération donne plus de précision.

7. Evolution de paramètre

1er architecture :







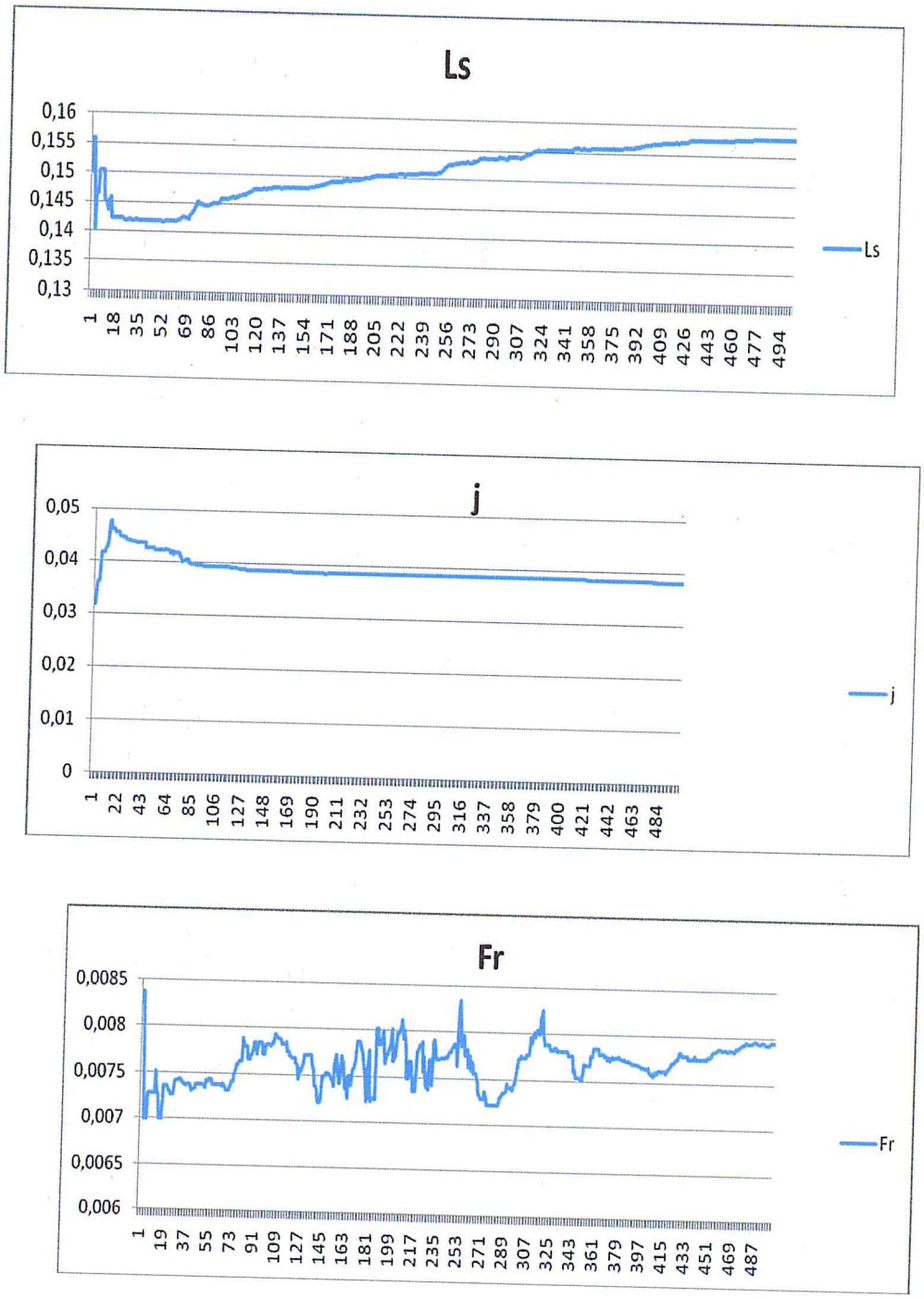
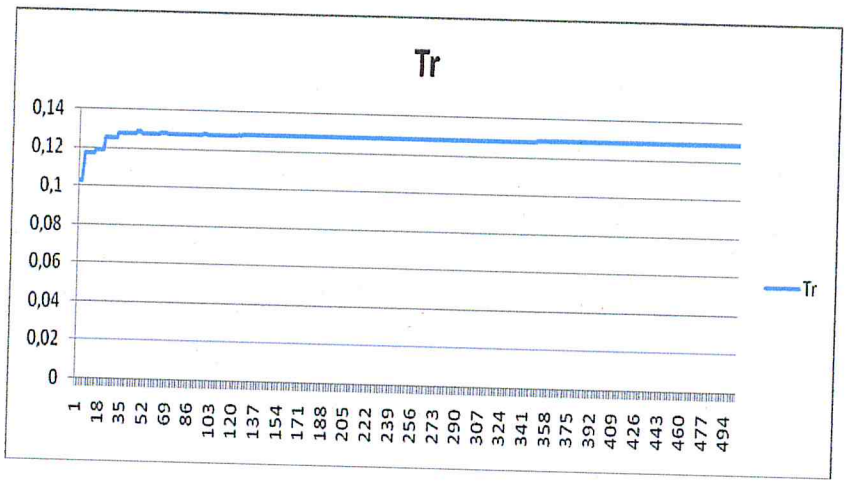
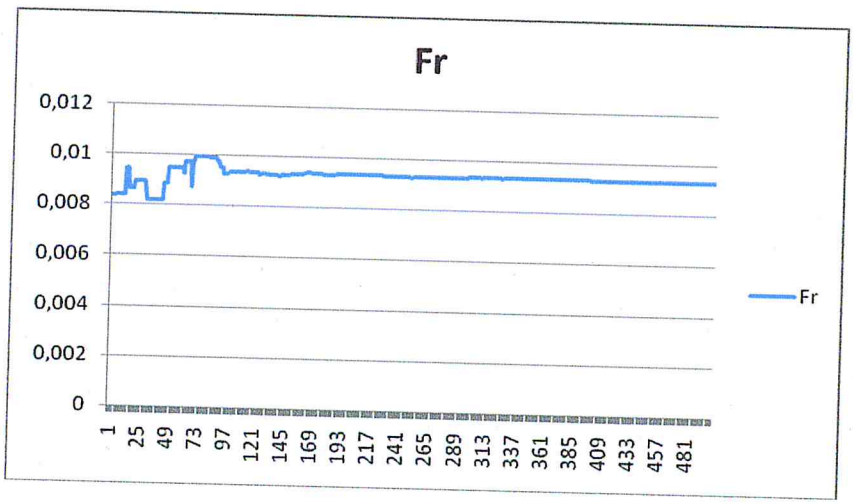
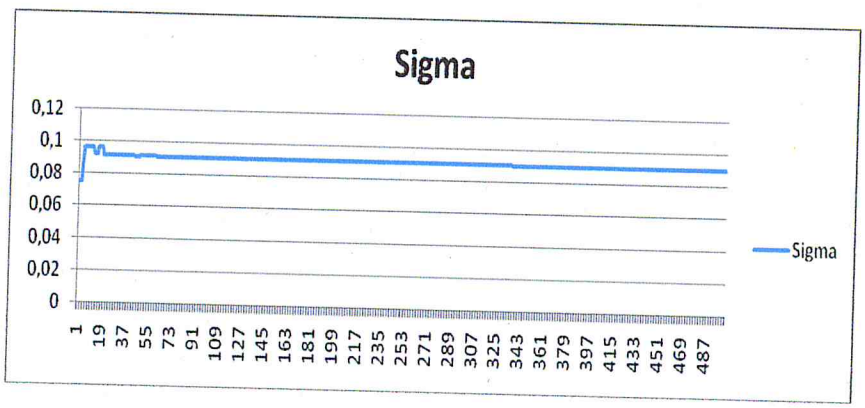


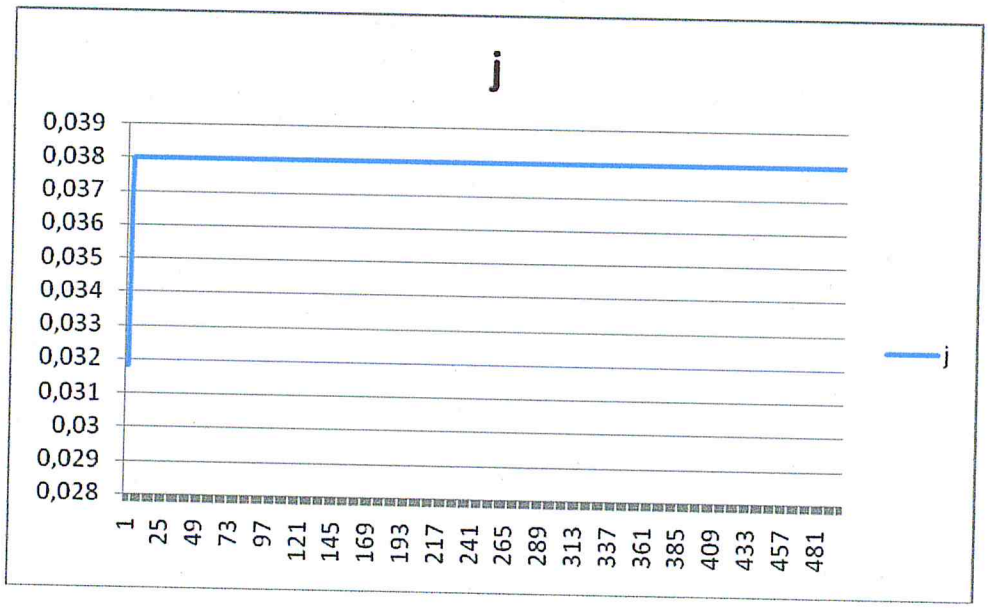
Figure 4.9 L'évolution des paramètres du Moteur simulé pour la 1er architecture

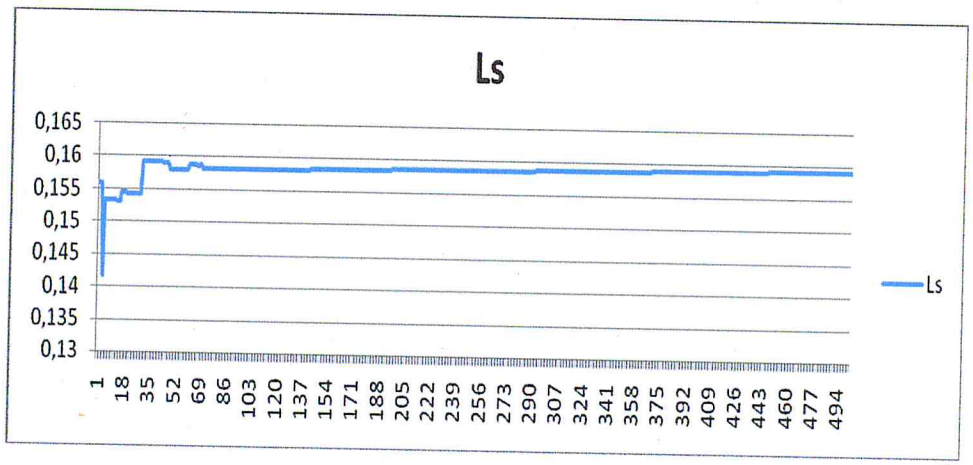
2eme architecture :











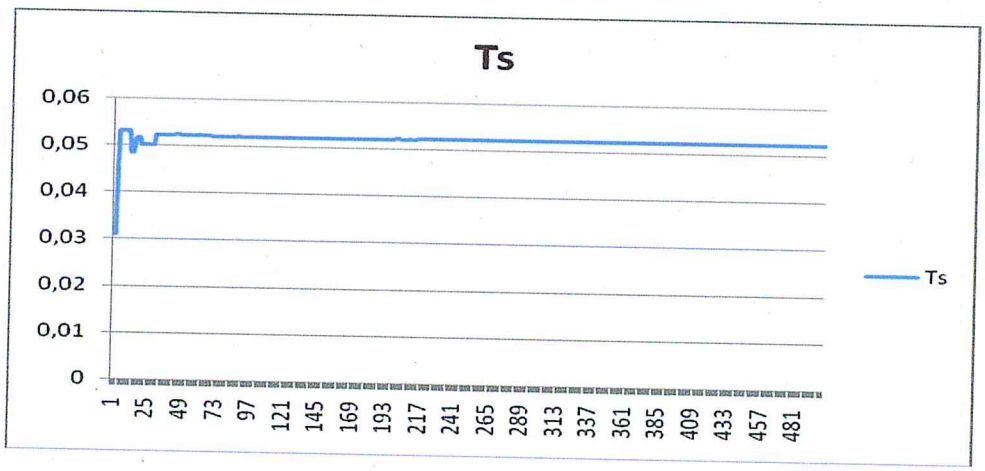


Figure 4.10L'évolution des paramètres du Moteur simulé pour la 2eme architecture.

8. Signal obtenu par calcule

Après les testes on utilise les paramètres estimés et on désigne le courant et on compare avec le courant simulé :

- 1er architecteur :

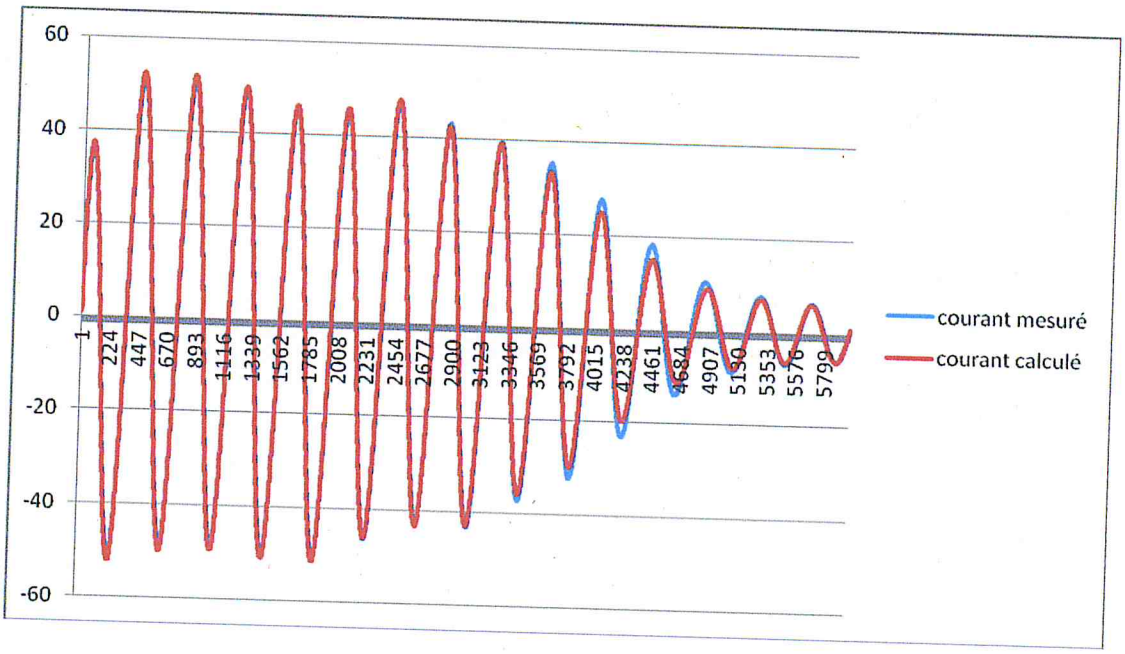


Figure 4.11 : Courant obtenu pour la 1ère architecture comparée avec le courant obtenu par les paramètres de simulation.

On remarque d'après figure (4.9) que le modèle simulé superpose sur le modèle connu avec une légère déformation et une faible erreur sur les paramètres du moteur.

- 2eme architecteur :

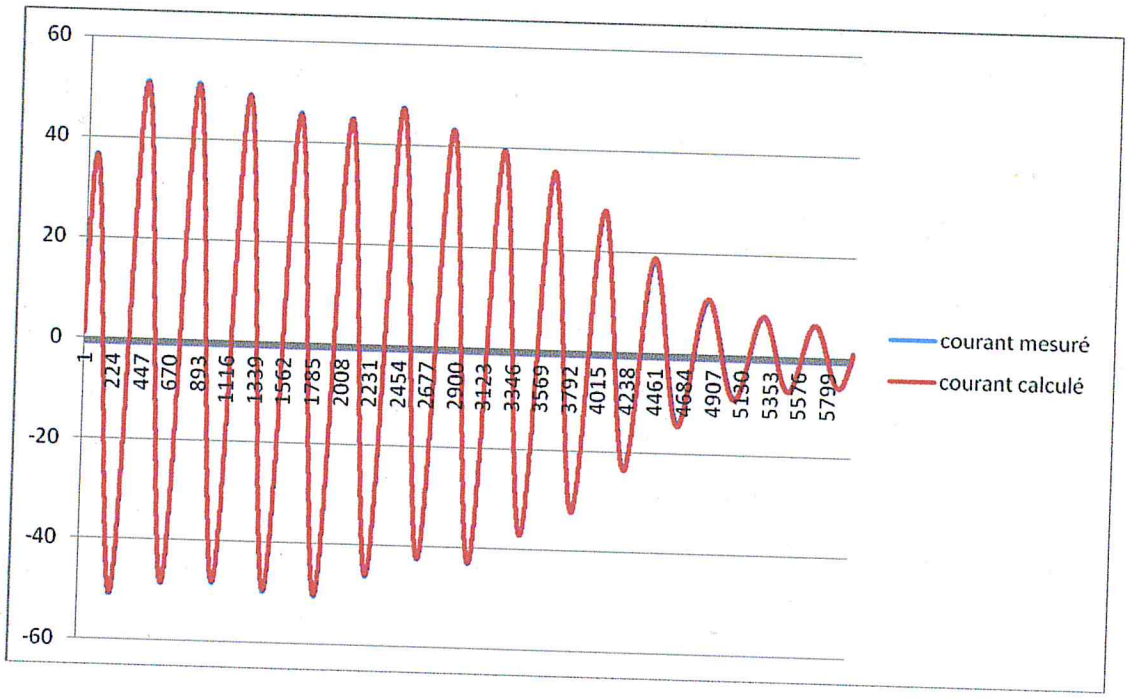


Figure 4.12 : Courant obtenu pour la 2eme architecture comparée avec le courant obtenu par les paramètres de simulation

On remarque d'après figure (4.10) La superposition des deux courants : mesuré et calculé.

Comparaisons entre les 2 architectures

Après Comparaisons entre les 2 architectures. On peut conclue les points suivants :

1) La 1er architecteur va améliorai le temps de calcule mais la structure n'est pas indépendant (les particules nous sons pas indépendant) par contre la 2eme architecteur est indépendant pour la structures et le calcule.
2) Le comportement des particules joue un rôle très importent pour la réalisation des résultats dans la 1er architecteurs a chaque déplacement les particules cherche la meilleure solution connue de son voisinage.
   La 2eme architecteur, les particules réalisons plusieurs itérations sans mise à jour de l'information issue de voisinage ce qui permet une fort recherche local.

3) on remarque bien que le signal obtenu pour la 2eme architecture est plus superposé au signal mesuré. cela justifie l'erreur quadratique minimale du deuxième architecture est plus minimum que l'erreur quadratique minimale du première architecture. et les valeurs des paramètres trouvés par la deuxième architecture sont très proches que les paramètres trouvent par premier architecture.

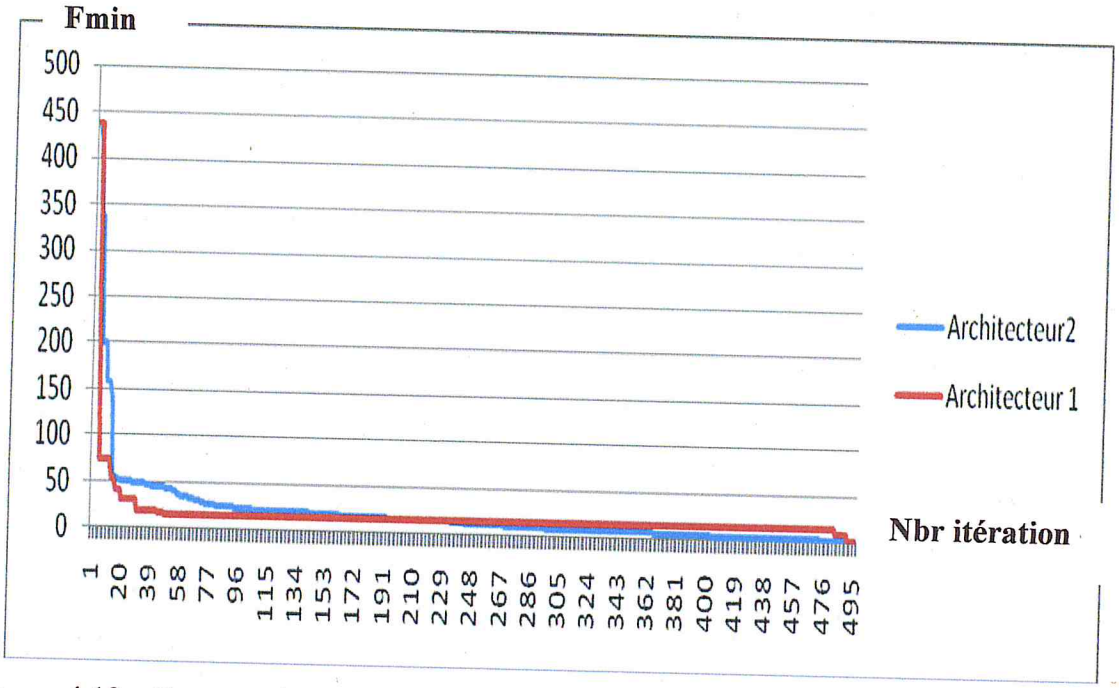


Figure 4.13 : Comparaison entre le courant calculé de 1er architecture et 2eme architecture.

9. conclusion

On a présenté dans ce chapitre le logiciel dédié a la résolution du problème par PSO. La 2eme partie de ce chapitre illustre quelque teste effectués sur deux architecture, évidement plusieurs autres peuvent être réalisés.

Nous avons validé les algorithmes par des donnes simulés, les résultats obtenus sont très satisfaisante.

Conclusion générale

Le travail présenté est une contribution à l'identification paramétrique de la machine asynchrone. Il s'appuie sur les étapes suivantes :

- Le choix de la méthode d'optimisation.
- Le chois de la modèle de la machine.

Nous avons testés notre approche d'identification en utilisant l'algorithme PSO appliqué sous mono poste et sous réseau.

Cette nouvelle méthode d'optimisation PSO est bien adaptée à l'identification de la machine asynchrone qui est régie par un système d'équation non linéaire ne permettant pas l'obtention d'une expression mathématique de l'erreur quadratique.

Ce travail va permette de réduire le temps de calcul qui permettre au concepteur des machines de réalisés plusieurs testes pour la validation des résultats obtenus.

Le travail que nous avons réalisé, nous a permis d'approfondir nos connaissances dans:

La modélisation, l'optimisation, et l'implémentation sous réseau.

- Principaux symboles :

R_s, (R_r) : Résistance par phase d'un enroulement au stator (respectivement rotor)

I_s, L_r : Inductance propre d'un enroulement stator (respectivement rotor)

M_{rs} : Mutuelle inductance cyclique.

L_r, L_s : Inductance propre d'un enroulement rotor (respectivement stator)

m_s, m_r : Mutuelle inductance entre deux enroulements au stator (respectivement rotor)

f : Coefficient des frottements visqueux.

P : Nombre de paire de pôles.

J : Inertie du moteur chargé.

- Principales grandeurs :

Ω : Vitesse de rotation mécanique de rotor.

V: Tension simple (valeur efficace du fondamental)

I : Courant de ligne (valeur efficace du fondamental)

Cem : Couple électromagnétique.

Ω :Vitesse de rotation mécanique du rotor.

□ : Vitesse angulaire électrique.

θ :Angle entre un enroulement rototique et son homologue statorique.

- **Rappel sur les calculs matriciels :**

1. Déterminant d'une matrice carrée :

1.1. Déterminant d'ordre 2 : $\begin{vmatrix} a & b \\ c & d \end{vmatrix}$ = ad-bc

1.2. Déterminant d'ordre 3 : $\begin{vmatrix} a & b & c \\ d & e & f \\ g & h & i \end{vmatrix}$ = aX+bN+cM

$$X=\begin{vmatrix} e & f \\ h & i \end{vmatrix} ; N=-\begin{vmatrix} d & f \\ g & i \end{vmatrix} ; M=\begin{vmatrix} d & \\ e & \end{vmatrix}$$

1.3. Inverse d'une matrice :

Pour qu'une matrice soit inversible, il faut que son déterminant soit non nul

$$A^{-1}=\frac{1}{\text{Det A}} . \text{com A}$$

Com A est la matrice obtenue en remplaçant chaque coefficient par son cofacteur. Dans certains cas, ce calcul peut heureusement se simplifier.

1.4. Produit scalaire de deux vecteurs :

Soient u (x ; y ; z) et v (x' ; y' ; z') deux vecteurs.

On appelle produit scalaire de u et v le réel noté r

et défini par : r= xx' + yy' + zz'

Exemple : avec u (1 ; 2 ; 3) et v (2 ; 3 ; 6), on obtient :

r = 2 + 6 + 18 = 26

- **Méthode de Runge-Kutta :**

Les méthodes de Runge-Kutta sont des méthodes d'analyse numérique d'approximation de solutions d'équations différentielles. Elles ont été nommées ainsi en l'honneur des mathématiciens 74Carl Runge et Martin Wilhelm Kutta lesquels élaborèrent la méthode en 1901.

Ces méthodes reposent sur le principe de l'itération, c'est-à-dire qu'une première estimation de la solution est utilisée pour calculer une seconde estimation, plus précise, et ainsi de suite.

La méthode de Runge-Kutta d'ordre 1 (RK1)

Cette méthode est équivalente à la méthode d'Euler, une méthode simple de résolution d'équations différentielles du 1er degré.

Considérons le problème suivant :

$$y_{n+1} = y_n + \frac{h}{6}(k_1 - 2k_2 + 2k_3 + k_4)$$

où

$$k_1 = f(t_n, y_n)$$

$$k_2 = f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_1\right)$$

$$k_3 = f\left(t_n + \frac{h}{2}, y_n + \frac{h}{2}k_2\right)$$

$$k_4 = f(t_n + h, y_n + hk_3)$$

L'idée est que la valeur suivante (y_{n+1}) est approchée par la somme de la valeur actuelle (y_n) et du produit de la taille de l'intervalle (h) par la pente estimée. La pente est obtenue par une moyenne pondérée de pentes :

- k_1 est la pente au début de l'intervalle ;
- k_2 est la pente au milieu de l'intervalle, en utilisant la pente k_1 pour calculer la valeur de y au point $t_n + h/2$ par le biais de la méthode d'Euler ;
- k_3 est de nouveau la pente au milieu de l'intervalle, mais obtenue cette fois en utilisant la pente k_2 pour calculer y;
- k_4 est la pente à la fin de l'intervalle, avec la valeur de y calculée en utilisant k_3.

Dans la moyenne des quatre pentes, un poids plus grand est donné aux pentes au point milieu.

$$\text{pente} = \frac{k_1 + 2k_2 + 2k_3 + k_4}{6}.$$

La méthode RK4 est une méthode d'ordre 4, ce qui signifie que l'erreur commise à chaque étape est de l'ordre de h^5, alors que l'erreur totale accumulée est de l'ordre de h^4.

$$y' = f(t, y), \quad y(t_0) = y_0$$

La méthode RK1 utilise l'équation

$$y_{n+1} = y_n + h f\,(t, y_n)$$

où h est le pas de l'itération.

La méthode de Runge-Kutta d'ordre 2 (RK2)

La méthode RK2 du point milieu est une composition de la méthode d'Euler:

$$y_{n+1} = y_n + h f\left(t + \frac{h}{2}, y_n + \frac{h}{2} f\,(t, y_n)\right)$$

où h est le pas de l'itération.

Elle consiste à estimer la dérivée au milieu du pas d'intégration:

$$y_{n+\frac{1}{2}} = y_n + \frac{h}{2} f\,(t, y_n)$$

$$y'_{n+\frac{1}{2}} - f\left(t + \frac{h}{2}, y_{n+\frac{1}{2}}\right)$$

et à refaire le pas d'intégration complet à partir de cette estimation :

$$y_{n+1} = y_n + h y'_{n+\frac{1}{2}}$$

C'est le cas particulier pour $\alpha = \frac{1}{2}$ de la méthode plus générale

$$y_{n+1} = y_n + h\left[\left(1 - \frac{1}{2\alpha}\right) f\,(t, y_n) + \frac{1}{2\alpha} f\,(t + \alpha\, h, y_n + \alpha\, h f\,(t, y_n))\right]$$

C'est une méthode d'ordre 2 car l'erreur est de l'ordre de h^3.

La méthode de Runge-Kutta classique d'ordre quatre (RK4)

C'est un cas particulier d'usage très fréquent, noté RK4.

Considérons le problème suivant :

$$y' = f(t, y), \quad y(t_0) = y_0$$

La méthode RK4 est donnée par l'équation :

Référence

[1] : BOMBRUN Maxime et SENE Abdoulaye, 2011. L'optimisation par essaim particulaire pour des problèmes d'ordonnancement Rapport d'ingénieur, Projet de 2ème année, Filière 4 : Calculs et modélisations scientifiques.

[2] : TFAILI Walid, 2007. Conception d'un algorithme de colonie de fourmis pour l'optimisation continue dynamique, thèse de doctorat de l'université paris, Spécialité : sciences de l'ingénieur.

[3] : AYARI Naouel, mémoire mastere, 2010, métaheuristiques parallèles hybrides pour l'optimisation combinatoire.

[4] : AMIRAT Imene, 2010, mémoire ingéniorat, spécialité : SI, université SAAD DAHLEB, Application de « particule swarm optimisation » pour l'identification paramétrique de la machine asynchrone.

[5] : Maurice Clerc et Patrick Siarry, une nouvelle métaheuristique pour l'optimisation ; difficile : la méthode des essaims particulaires.

[6] : OULD-AISSA Ahmed, Mémoire Magister, 2010, spécialité : signaux et systéme, université SAAD DAHLEB, Application de « particule swarm optimisation » pour l'identification paramétrique de la machine asynchrone »

[7]: C.H.papadimitriou, K.Steiglitz, combinatorial optimization-algoritms and complexity.prentice hall, 1982.

[8]: C.C.Ribeiro,N.Maculan (Eds.),applications of combinatorial optimization.annals of operations research 50,1994.

[9] : G.Laporte, I.H.Osman, Metaheuristicsbin combinatorial optimization,annals of operations research 63,J.C.baltzer science publishers,basel,Switzerland,1996.

[10]: Johann Dréo, Alain Pétrowski, Patrick Siarry, Eric Taillard. Métaheuristiques pour l'optimisation difficile. Eyrolles, 2005.

[11] : Sébastien Verel, Métaheuristiques pour l'Optimisation Difficile, équipe ScoBi, Université de Nice Antipolis 2008, cours.

[12]: J. Kennedy, R.C. Eberhart. (1995).Particle Swarm Optimisation.. Proc.IEEE Int. Conf. On Neural Networks, November 27-December 1, 1995,pp.1942-1948, Perth, Australia.

[14] : Maxime BOMBRUN Abdoulaye SENE, ,2011 Rapport d'ingénieur, Projet de 2ème année, Filière 4 : Calculs et modélisations scientifiques, L'optimisation par essaim particulaire pour des problèmes d'ordonnancement.

[15] : H.Omessaad"Contribution au devloppement de méthods d'optimisation stochastique. Application a la concepation des dispositif electrotechniques" Mémoire de Doctorat, Ecole central de lille, 2003