



# Détermination de l'architecture de modèles neuromimétiques par implantation parallèle

Laurent Bougrain, Yann Boniface

## ► To cite this version:

Laurent Bougrain, Yann Boniface. Détermination de l'architecture de modèles neuromimétiques par implantation parallèle. Journées Scientifiques du Centre Charles Hermite, Jan 2000, none, 3 p. inria-00107849

**HAL Id: inria-00107849**

**<https://inria.hal.science/inria-00107849>**

Submitted on 19 Oct 2006

**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.

# Détermination de l'architecture de modèles neuromimétiques par implantation parallèle

Laurent Bougrain et Yann Boniface  
LORIA, BP 239, 54506 Vandoeuvre-lès-Nancy cedex, France

*e-mail*: bougrain@loria.fr, boniface@loria.fr

## Résumé

La méthode présentée dans cet article permet de déterminer de manière systématique l'architecture idéale pour modéliser une fonction à l'aide de réseaux neuromimétiques. Partant d'une structure volontairement surdimensionnée, ce qui permet de modéliser des fonctions complexes, l'architecture peut ensuite être simplifiée en conservant, voire en améliorant, les capacités de modélisation. L'apprentissage se fait par itérations successives. La méthode nécessite un nombre important d'unités de calcul, de connexions et d'itérations, par conséquent son implantation parallèle est intéressante en vue de réduire le temps de calcul nécessaire à l'élaboration de l'architecture optimale.

**Mots-clés** réseaux de neurones, parallélisme, élagage.

## 1 Introduction

Les réseaux de neurones artificiels sont connus pour être des approximateurs universels de fonctions. Ils sont constitués d'unités de calcul (les neurones formels) qui communiquent entre elles (via les connexions). L'architecture, c'est à dire le nombre de neurones et leur agencement, détermine la famille de fonctions modélisables. Une architecture trop complexe par rapport à la tâche à modéliser entraîne une augmentation du temps de calcul et l'apparition d'un bruit. Une architecture trop simple ne permet pas de modéliser des fonctions complexes. Pour déterminer l'architecture idéale deux méthodes sont possibles : partir d'une architecture élémentaire et augmenter sa complexité ou bien partir d'une architecture complexe et supprimer les unités de calcul et les connexions peu utiles. La deuxième méthode présente l'avantage de donner des renseignements sur la pertinence des données. Dans les deux cas, de nombreuses itérations sont nécessaires à l'élaboration de la configuration idéale. La suite de l'article décrit une méthode nouvelle, appartenant à la deuxième catégorie, qui tire avantage de sa possible implémentation sur des machines parallèles.

## 2 Réseaux neuromimétiques contextuels

Généralement, à un réseau neuromimétique correspond une fonction (les variables d'entrée sont instanciées aux données courantes et le modèle produit un résultat). L'utilisation d'une architecture plus complexe, comme celle d'un Orthogonal Weight Estimator (OWE) couplée à un perceptron multicouches (MLP) [6], permet de modifier les paramètres de la fonction d'après les données courantes [3]. Figure 1, les valeurs constituant le contexte  $\phi$  sont propagées dans les

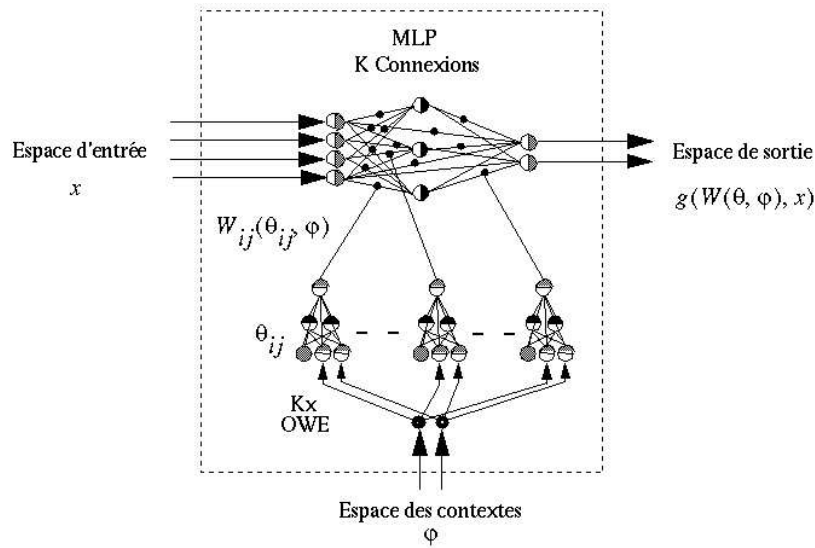


FIG. 1 – Architecture des OWEs associés à un MLP

OWEs de manière à calculer les poids du réseau principal (ici un perceptron multicouches). Une fois les poids calculés, c'est à dire les paramètres de la fonction connus, on propage dans le réseau principal les données courantes pour obtenir la réponse du système. De cette manière, il est possible d'augmenter le nombre de fonctions modélisables par une seule architecture d'une part et d'adapter la fonction aux données courantes d'autre part (l'importance de l'adaptation d'un modèle au contexte dans des problèmes réels a déjà été abordée dans [2]). De plus, il est possible de simplifier l'architecture par suppression des connexions les moins pertinentes (en calculant la variation de l'erreur entre la présence et l'absence d'une connexion et ce pour chaque connexion. L'ordonnancement de leur variation range les connexions de la moins importante à la plus importante [4]). On dispose pour cela d'une grande quantité de données liées à une tâche de prédiction dans le domaine de la téléphonie mobile.

### 3 Parallélisation

L'intérêt de l'implantation parallèle des réseaux de neurones a été abordée [5]. Puisque l'on procède par simplification, l'une des difficultés est engendrée par la grande taille du réseau de départ et le nombre d'opérations nécessaires à l'obtention de l'architecture optimale recherchée (l'ordre de grandeur pour notre tâche de prédiction est de 15000 neurones, 100000 connexions et 50000 données en 32 dimensions). L'apprentissage de ce type de réseaux est donc extrêmement lent, tandis que l'utilisation du réseau final élagué sera très rapide. Nous avons implanté notre réseau à l'aide d'une librairie de développement parallèle de réseaux de neurones [1] qui nous offre deux avantages. Premièrement, le temps d'implantation s'est trouvé considérablement réduit par l'utilisation des fonctions offertes. Deuxièmement, cette librairie nous permet, avec le même code, d'effectuer l'apprentissage sur machines parallèles MIMD à mémoire partagée, et d'utiliser le réseau émergent sur machines séquentielles classiques. L'accélération obtenue sur l'origine 2000 en multi-utilisateurs est de deux sur trois/quatre processeurs et de six sur une vingtaine de processeurs (ces résultats s'expliquent par le caractère périodiquement séquentiel de notre type d'apprentissage) (Figure 2). Au delà de l'accélération, le gain en temps de calcul est très appréciable (en terme d'heures).

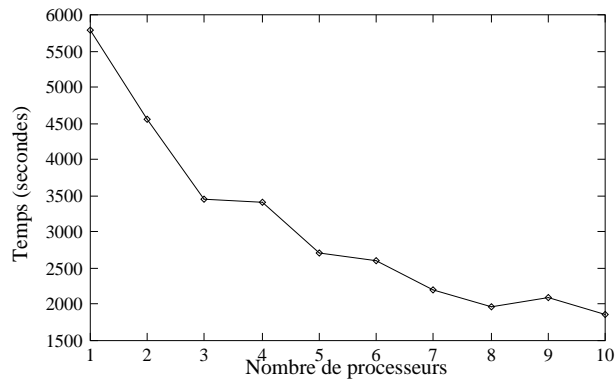


FIG. 2 – Performances en parallélisation

## 4 Conclusion

L'application d'une méthode d'élagage, telle que Optimal Brain Damage, à l'architecture complexe d'un Orthogonal Weight Estimator permet de cumuler les performances de ces deux méthodes. Leur complémentarité permet d'obtenir une architecture de réseaux neuromimétiques idéale. De plus, l'implémentation parallèle de ce couplage réduit le temps de calcul durant la phase de détermination pour que cette méthode puisse être réellement appliquée. L'implantation a été facilitée par l'utilisation d'une librairie spécifique aux réseaux de neurones, développée par un des auteurs. Une fois l'architecture obtenue, la phase d'utilisation peut être exécutée soit sur machines parallèles, soit sur machines séquentielles.

## Références

- [1] Y. Boniface, F. Alexandre, and S. Vialle. A bridge between two paradigms for parallelism : Neural networks and general purpose mind computers. In *IJCNN*, 1999.
- [2] L. Bougrain and F. Alexandre. Unsupervised connectionist clustering algorithms for a better supervised prediction: Application to a radio communication problem. In *Proc. of International Joint Conference on Neural Networks*, Washington, 1999.
- [3] L. Bougrain, N. Pican, and F. Alexandre. Rôle du contexte dans le modèle owe: un réseau de neurones artificiels utilisant des connexions axo-synaptiques. In *Proc. of NeuroSciences pour Ingenieur*, Munster, 1998.
- [4] Y. Le Cun, J.S. Denker, and S.A. Solla. Optimal brain damage. In D.S. Touretzky, editor, *Advances in Neural Information Processing Systems II (Denver 1989)*, pages 598–605, San Mateo, 1990. Morgan Kaufmann.
- [5] M. Misra. Parallel environments for implementing neural network. *Neural Computing Survey*, 1:48–60, 1996.
- [6] N. Pican. An orthogonal delta weight estimator for mlp architectures. *ICNN'96. Washington, DC, USA*, 1996.