Projet ESPRIT ELENA
Réalisations VLSI de réseaux de neurones

Michel Verleysen * et Joan Cabestany **

* Université Catholique de Louvain, Laboratoire de Microélectronique,
3 pl. du Levant, B-1348 Louvain-la-Neuve (Belgique)

** Universitat Politecnica de Catalunya, ETSE Telecomunicacio,
Gran Capita s/n, modul C4, E-08071 Barcelona (Espagne)

Abstract

Le projet ESPRIT ELENA-NervesII, financé par la Communauté Européenne, vise à étudier les architectures neuronales évolutives, c'est-à-dire celles dont la structure (nombre de neurones, connexions,...) varie au cours de l'apprentissage en fonction du problème à traiter. Le projet est structuré sur différents axes pour couvrir les aspects théoriques, de simulation et de réalisation. Le troixième axe de ce projet couvre les réalisations matérielles de ce type de réseau de neurones, principalement par des architectures intégrées VLSI. Ce papier présente les différentes approches étudiées dans le projet ELENA, tant digitales qu'analogiques, et les nouvelles perspectives vers lesquelles la recherche dans ce domaine est en train de s'orienter.

1 Introduction

Une limitation contraignante de la plupart des réseaux de neurones artificiels réside dans le caractère fixe de leur structure. Une fois la structure choisie, l'algorithme d'apprentissage modifie en effet la valeur des connexions entre les neurones, éventuellement la forme des fonctions non-linéaires ou des seuils, mais le nombre de neurones et de connexions reste inchangé. Le problème est d'autant plus important qu'il est en général difficile de fixer le nombre de neurones et de connexions nécessaires pour un problème donné, nombre qui, s'il est trop petit, peut empêcher le réseau de réaliser la fonction qu'on souhaite lui faire réaliser, et qui, s'il est trop grand, peut diminuer les fonctions de généralisation du réseau, un peu comme si l'on voulait faire passer un polynome de degré trop important à travers un trop petit nombre de points.

Les réseaux de neurones évolutifs peuvent apporter une solution à ce type de problème. Ces réseaux, comme leur nom l'indique, ont une structure qui peut évoluer en fonction de l'apprentissage, tant en augmentant qu'en diminuant le nombre de connexions et/ou de neurones. Un exemple de ce type de réseau est le RCE, réseau destiné à la classification, dont le principe est de créer dans l'espace des stimuli autant de zones (de forme sphérique) qu'il est nécessaire pour arriver à une classification correcte de tous les points présentés à l'entrée du réseau. La description complète de cet algorithme peut être trouvée dans [7].

Le projet ESPRIT ELENA-NervesII, financé par la Communauté Européenne pour une durée de trois ans (depuis le 1 juillet 1992), vise à étudier ce type d'algorithmes. Le projet comprend six partenaires:

- l'Institut National Polytechnique de Grenoble, Laboratoire de Traitement d'Images et de Reconnaissance de Formes, France (INPG-LTIRF)
- Thomson-Sintra Acitivités Sous-Marines à Sophia-Antipolis, France (TSA-ASM)
- l'Universitat Politecnica de Catalunya, ETSE Telecomunicacio, Barcelona, Espagne (UPC-DEE)
- l'Université Catholique de Louvain, Laboratoire de Microélectronique, Louvain-la-Neuve, Belgique (UCL-DICE)
- l'Ecole Polytechnique Fédérale de Lausanne, Laboratoire de Microinformatique, Suisse (EPFL-LAMI)
- l'Ecole pour les Etudes et la Recherche en Informatique et Electronique, LERI, Nîmes, France (EERIE-LERI)

Il est divisé en trois axes, qui concernent respectivement la théorie, la simulation logicielle et l'implémentation matérielle des réseaux de neurones évolutifs. Les problèmes de classification de données par réseaux de neurones évolutifs forment le sujet principal du projet.

Dans le premier axe, qui concerne la théorie, les études portent sur les limites de performances que l'on peut atteindre pour un classifieur sur une base de données définie, en essayant d'atteindre un classifieur Bayésien par des estimations de densités de probabilités par noyaux. Les meilleures performances possible sont supposées être atteintes par ce classifieur Bayésien. Les performances d'un classifieur déterminé peuvent alors être comparées à celles du classifieur optimal par l'intermédiaire de la matrice de confusion, donnant les probabilités d'erreurs pour chaque paire de classes (probabilité qu'un vecteur de la classe i soit classé dans la classe j).

Dans le second axe, qui concerne la simulation logicielle des algorithmes de classification, un environnement de programmation, appelé Packlib, est en cours de développement. Il s'agit d'un environnement graphique basé sur l'utilisation de modules, soit fournis de manière standard, soit développés en C par l'utilisateur, qui peuvent être reliés de manière quasi-interactive (sans programmation supplémentaire) afin de programmer ses propres algorithmes. Comme exemple de modules déjà réalisés, on peut citer des modules de prétraitement statistique de données, des algorithmes de quantification vectorielle, des algorithmes neuronaux tels que le RCE, le LVQ ou les cartes de Kohonen, et des modules de représentation graphique de données en deux dimensions ou plus. Cet environnement permet de simuler de façon rapide et conviviale des algorithmes connus ou nouveaux, sans devoir penser aux problèmes d'interfaçages, et permet aussi le test des algorithmes sur un certain nombre de bases de données artificielles ou industrielles qui ont été soit générées soit récoltées dans le cadre du projet.

Le troisième axe enfin concerne les implémentations matérielles de réseaux de neurones, et en particulier des algorithmes évolutifs étudiés dans le cadre de ce projet. Cet axe est détaillé au paragraphe suivant.

2 Implémentations matérielles

Les algorithmes neuronaux de classification, et en particulier les algorithmes évolutifs, peuvent être divisés en deux classes principales: il s'agit d'une part des algorithmes de type ROI (Region-Of-Influence), et d'autre part des algorithmes PLS (Picewise Linear Separation).

Les algorithmes ROI fonctionnent, comme leur nom l'indique, par le principe de régions d'influences. Celles-ci consistent en général en des fonctions radiales de base, de la dimension de l'espace des données, et qui sont positionnées dans l'espace suivant un algorithme donné (supervisé). Lors de la reconnaissance d'un vecteur, les régions auxquelles ce vecteur appartient, et les classes associées à ces régions, seront déterminées; une décision d'appartenance à une classe (ou à aucune) sera alors prise. L'exemple peut-être le plus simple d'algorithme ROI (mais sans doute un des moins performants ...) est l'algorithme RCE [7]. En phase d'apprentissage, celui-ci consiste à placer autant de régions d'influences de type hypersphères centrées sur les données à apprendre avec un rayon initial fixé à priori, qu'il est nécessaire pour classer correctement l'ensemble des données. Lors de la détection d'une mauvaise classification, dûe à une région d'influence trop grande d'une mauvaise classe, la ou les régions d'influence fautives sont réduites pour éviter cette mauvaise classification. Le double processus de création de nouvelles classes et de diminution de leur taille est itéré jusqu'à un taux de reconnaissance de 100% de l'ensemble d'apprentissage; il est évident que ce processus conduira à une capacité de généralisation du réseau extrèmement mauvaise en cas de recouvrement de classes dans l'espace des données.

Les algorithmes PLS, eux, fonctionnent de manière différente. Lors de l'apprentissage, c'est l'espace tout entier qui est divisé de manière générale par des hyper-plans en un certain nombre de régions, chacune associée à une classe. Le nombre de divisions, et donc le nombre de neurones réalisant une partition binaire d'une portion de l'espace, est limité par un critère d'arrêt de l'algorithme d'apprentissage. Des exemples d'algorithmes PLS sont les arbres de décision binaire et les algorithmes "upstart" et "tiling" [2, 5].

La différence entre les algorithmes ROI et PLS se fait sentir au niveau même de leur structure. Les premiers peuvent de manière générale être représentés par le modèle de la figure 1, les deuxièmes ayant eux une structure non-déterminée, de type perceptron multi-couches, ou en arbre, comme représenté à la figure 2. L'équivalence entre els deux structures est bien établie.

Il existe un troisième type de structure neuronale, qui peut être rendue évolutive. Il s'agit des algorithmes à noyaux, principalement étudiés de manière théorique dans le premier axe du projet. Ce type de réseau n'essaye plus de déterminer directement des classes dans l'espace des données, par la création de régions de taille finie (ROI) ou infinie (PLS), mais essayent au contraire d'estimer les densités de probabilités de chaque classe, pour en déduire un classifieur bayésien quasi-optimal. L'estimation des densités de probabilités de chaque classe peut être faite au travers d'estimateurs à noyaux (par exemple gaussiens) qui sont centrés soit sur chacune des données d'apprentissage, soit sur des points déterminés par une méthode de quantification vectorielle. La somme des noyaux associés à chacune des classes donne alors (sous certaines conditions) une estimation des densités de probablité des classes, densités dont les intersections (après multiplication éventuelle par les probabilités a priori) donnent les frontières bayésiennes.

Valgo, ISSN 1243-4825
Numéro 94-1 : les processeurs neuronaux - juin 1994

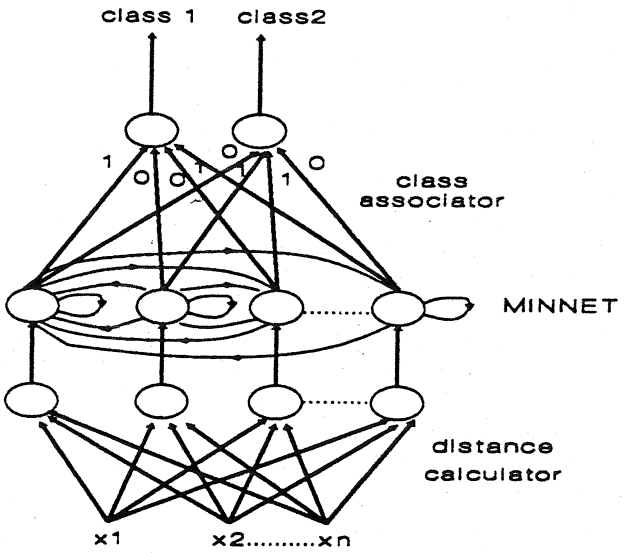

Figure 1: Structure générale d'un réseau ROI

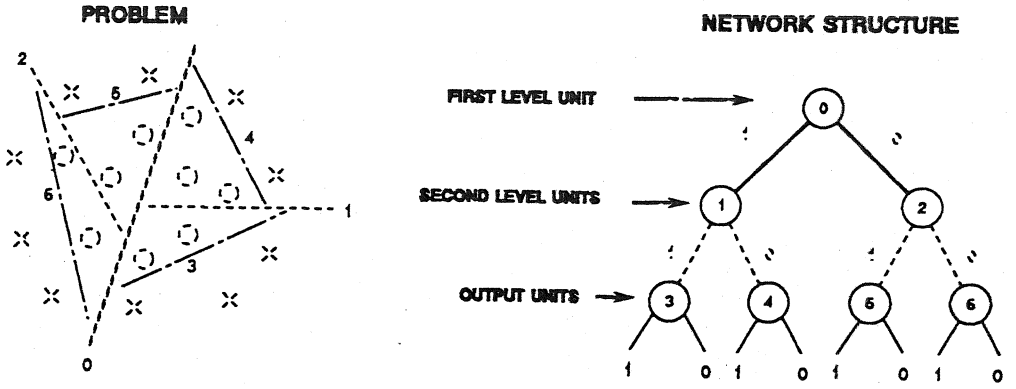

Figure 2: Structure d'un réseau PLS et la division de l'espace associée

On voit facilement qu'en termes d'algorithmes, ce que l'on va appeler un "algorithme à noyaux" va être très similaire à un réseau ROI, si ce n'est que les sorties des fonctions non-linéaires (noyaux dans le premier cas ou de type sigmoïde dans le second) vont être sommées dans le cas des algorithmes à noyaux plutôt que comparées (dans le cas général) pour les algorithmes ROI. Les resources-calcul seront néanmoins très similaires dans les deux cas, ce qui permettra de regrouper ces deux types d'algorithmes au point de vue des implémentations matérielles.

Dans le projet ELENA, les implémentations de ces trois types de réseaux sont étudiées, soit par des méthodes digitales, soit par des méthodes analogiques. La section suivante présente les problèmes de précision dans les algorithmes qui seront cruciaux pour les implémentations matérielles, tandis que les deux dernières sections présentent de manière succinte les travaux qui sont fait d'une part du côté digital en ce qui concerne les algorithmes PLS et d'autre part du côté analogique dans le but d'implémenter des algorithmes ROI et à noyaux.

3 Contraintes matérielles

Les implémentations matérielles de n'importe quel algorithme y imposent un certain nombre de restrictions. Celles-ci peuvent concerner la précision des valeurs utilisées et leur dynamique, ou les problèmes d'allocation des opérations à effectuer aux resources-calcul disponibles. Le second type de restriction étant adressé par le type d'architecture lui-même, nous nous intéresserons plutôt ici au premier type.

L'influence de la précision des opérations sur le comportement d'un algorithme peut difficilement être étudié d'un point de vue théorique. Dans le cadre de ce projet, nous nous sommes limités aux études expérimentales, en essayant lorsque l'occasion le permet de tirer des conclusions générales.

Des simulations ont été effectuées avec différents types d'algorithmes. A titre d'exemple ici, nous nous baserons sur l'algorithme RCE déjà mentionné. Les simulations, effectuées dans l'environnement Packlib, avaient pour but d'illustrer les pourcentages de reconnaissance de l'algorithme sur des bases de données fixées a priori, en fonction de la précision des opérations, mais aussi le nombre de neurones nécessaires (pour atteindre 100% de reconnaissance de la base d'apprentissage dans le cas du RCE), le nombre d'itérations nécessaires,... Les distances Euclidienne et de Manhattan ont également été utilisées afin de comparer leurs performances.

A titre d'exemple, la figure 3 montre le pourcentage de classifications correctes de l'algorithme RCE sur une base de 1000 points situés dans un plan, une classe étant définie par l'intérieur d'un cercle, l'autre par une couronne entourant le cercle. La courbe "welclon" correspond à un apprentissage et une reconnaissance avec précision réduite, tandis que la courbe "welcloff" correspond à un apprentissage en pleine précision et une reconnaissance en précision réduite (respectivement pour un apprentissage réalisé sur ou en-dehors de l'architecture implémentée). La figure 4 montre le nombre de neurones nécessaires dans ces deux cas. La distance euclidienne a été utilisée pour ces simulations. De nombreuses autres simulations peuvent être trouvées dans [3].

De manière générale, les conclusions qui peuvent être tirées (dans le cas de l'algorithme RCE) sont les suivantes. Premièrement, le choix du type de distance

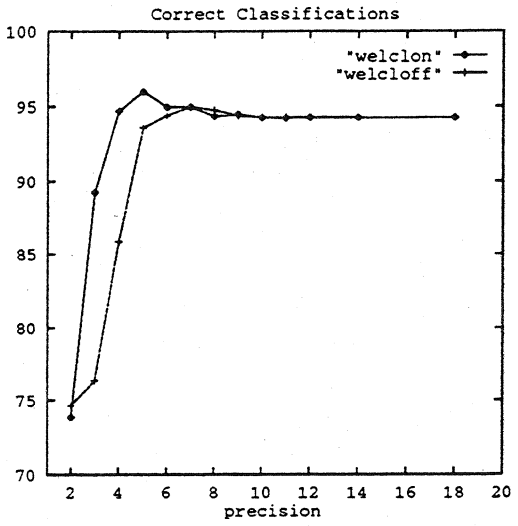

Figure 3: Pourcentage de classifications correctes de l'algorithme RCE sur l'ensemble de test (voir texte)

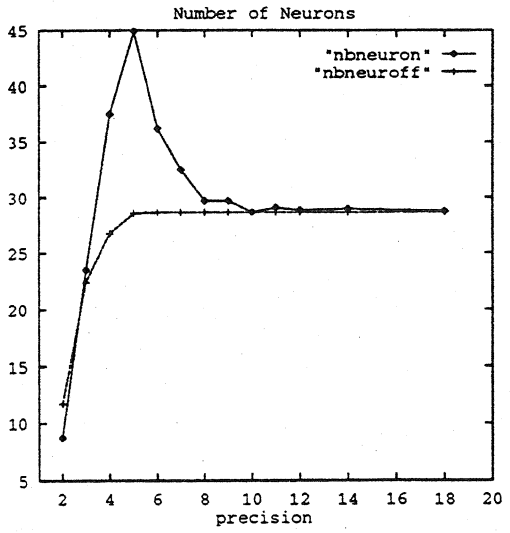

Figure 4: Nombre de neurones nécessaires pour l'algorithme RCE)

utilisé (Euclidienne ou Manhattan) importe peu. En effet, seule la forme des nuages de données dans l'espace peut donner lieu à une préférence pour l'un ou l'autre type de distance. En l'absence d'informations a priori sur la base de données, comme c'est la plupart du temps le cas dans des problèmes réels, il est impossible de prévoir si l'une ou l'autre mesure donnera des meilleures performances. Lors d'une implémentation matérielle, on privilégiera donc le type de distance le plus facilement réalisable.

Deuxièmement, une précision maximale de six à huit bits est généralement suffisante. Ceci s'explique non pas par la nature de l'algorithme lui-même, mais par les caractéristiques des bases de données. La précision de ces bases est en effet rarement supérieure à cette limite, surtout en présence de données industrielles provenant généralement de capteurs physiques dont la précision elle-même est limitée. Comme l'algorithme RCE ne fait pas intervenir de calculs intermédiaires tels qu'inversion de matrices ou adaptation de valeurs par petits incréments, calculs qui nécessiteraient une précision temporairement plus importante, la précision de la base de données elle-même est en général suffisante pour obtenir un fonctionnement correct de l'algorithme.

Enfin, et l'on pouvait s'y attendre, le taux de reconnaissance pour une précision donnée est en général plus important dans le cas d'un apprentissage avec précision réduite que dans l'autre cas; ceci est dû au fait que l'ensemble de l'algorithme ne travaille plus que sur un espace discret, où il est plus simple de déterminer les classes que dans le cas continu.

4 Implémentation digitale d'algorithmes PLS

Une caractéristique des algorithmes PLS est, nous l'avons vu plus haut, d'avoir une structure de type "arbre de décision binaire" qui se transforme aisément en une structure de type MLP (Multi-Layer Perceptron) ou une structure directe MLP, selon l'algorithme [6]. L'UPC-DEE avait développé, préalablement au projet ELENA, une architecture systolique appelée DRA (Dynamic Ring Architecture) adaptée aux algorithmes MLP. Dans le cadre du projet ELENA, cette architecture a donc tout naturellement été adaptée aux algorithmes MLP; le principe de cette tranformation est illustré à la figure 5.

L'architecture DRA peut être définie comme suit:

- Chaque unité de traitement est capable d'assurer la fonctionalité de plusieurs neurones dans plusieurs couches du réseau MLP; chaque unité a donc la double propriété de "neurone virtuel" et de "couche virtuelle".
- L'architecture est capable d'émuler le réseau MLP couche par couche par un multiplexage temporel.
- Les ports de communication bidirectionnels entre unités offrent une grande flexibilité dans la structure du réseau à implémenter, puisque chaque couche peut être composée d'un nombre différent de neurones.
- La vitesse de traitement du réseau croît linéairement avec le nombre d'unités de traitement.

Le principe de fonctionnement de l'architecture DRA peut être illustré par l'exemple suivant. Supposons que l'on veuille implémenter le MLP de la figure 6 sur l'architecture

Valgo, ISSN 1243-4825
Numéro 94-1 : les processeurs neuronaux - juin 1994

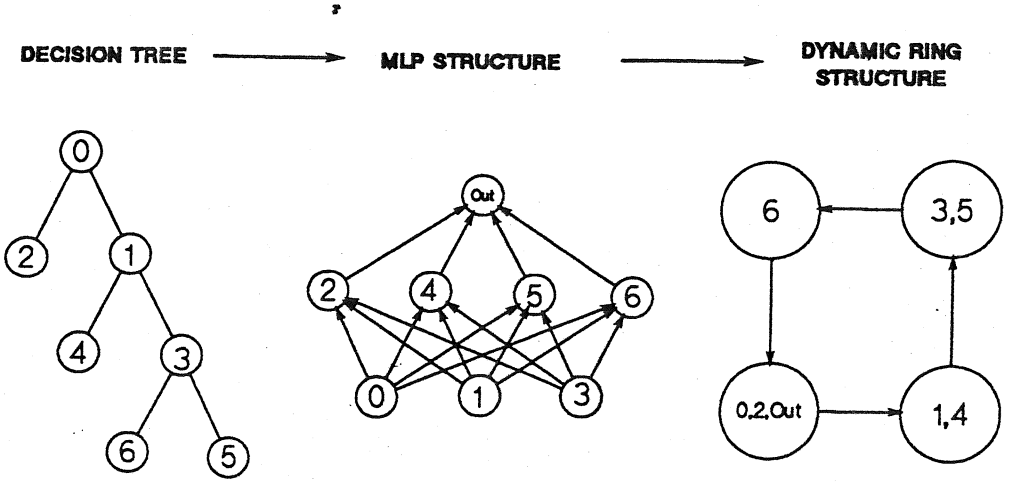

Figure 5: Transformation d'un algorithme PLS sur une architecture DRA

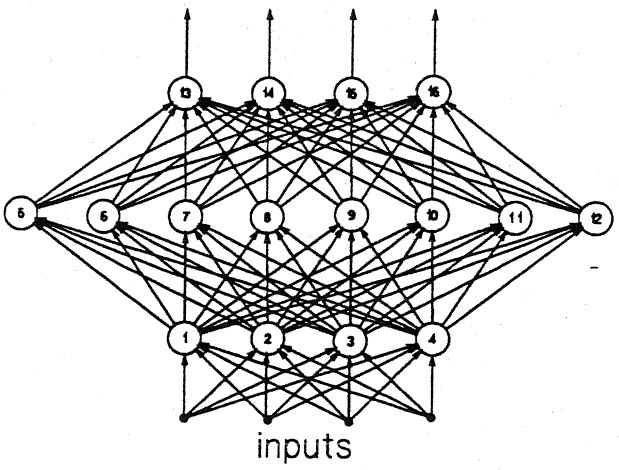

Figure 6: Perceptron multi-couches

DRA composée de huit processeurs illustrée à la figure 7. Chaque couche du MLP sera successivement calculée par le DRA, comme illustré à la figure 8; les processeurs en pointillés ne sont pas utilisés, tandis que ceux marqués par un damier ne sont utilisés que pour la transmission d'informations. La différence entre les couches 1 et 3 s'explique par le principe systolique utilisé dans l'architecture. A l'intérieur de chaque couche n, le calcul de l'activation (entrée de la fonction non-linéaire d'un neurone) ne sera complet que lorsque l'ensemble des produits partiels $w_{ij}x_j$ auront été calculés et sommés (w_{ij} est le poids de la connexion reliant le neurone j de la couche $n-1$ au neurone i de la couche n, et x_j est la sortie du neurone j de la couche $n-1$). Le principe est d'alors d'assurer un cadencement tel des opérations que les valeurs utilisées pour le calcul de chaque activation soient présentées successivement au processeur adéquat, ce qui explique l'utilisation illustrée à la figure 8. La propriété de "neurone virtuel", par laquelle plusieurs neurones de la même couche peuvent être implémentés sur le même processeur, est décrite plus en détails dans [1].

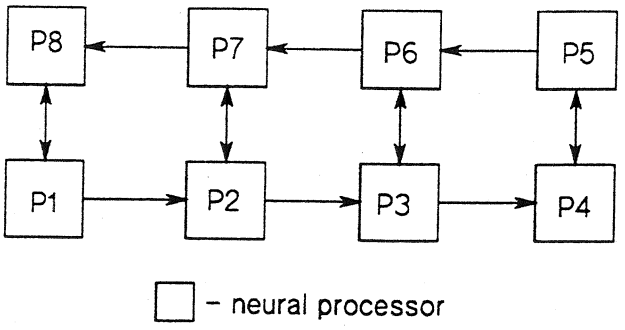

Figure 7: Architecture DRA à huit processeurs

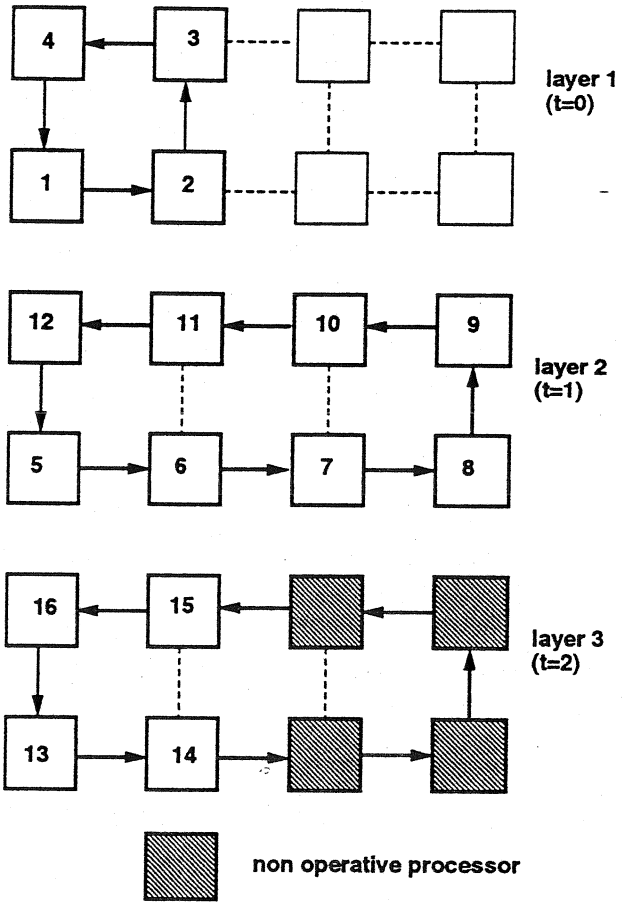

Figure 8: Emulation du réseau avec l'architecture DRA

Valgo, ISSN 1243-4825
Numéro 94-1 : les processeurs neuronaux - juin 1994

5 Implémentation analogique d'algorithmes ROI et à noyaux

L'architecture VLSI analogique, destinée à implémenter tant des algorithmes ROI que des algorithmes à noyaux, proposée dans le cadre de ce projet, est schématiquement représentée à la figure 9.

Il s'agit d'une architecture purement parallèle, qui regroupe les resources-calcul nécessaires à ce type d'algorithme. Les blocs de base devant être réalisés pour cette architecture sont les suivants:

- stockage de P centres de noyaux, chacun avec N coordonnées analogiques (l'architecture est donc supposée contenir N noyaux ou région d'influences dans un espace à P dimensions),
- stockage de C classes associées aux noyaux,
- possibilité soit d'enregistrer de nouvelles valeurs dans les points mémoires analogiques, soit d'adapter la valeur y contenue dans la direction du vecteur d'entrée,
- calcul des distances entre chaque noyau mémorisé et le vecteur d'entrée,
- choix de la plus petite de ces distances ("winner-take-all",...),
- fonctions (noyaux) non-linéaires sur ces P distances,
- addition des sorties des noyaux classe par classe (pour les estimations de densité de probabilités),
- multiplication de ces résultats par des constantes (probabilités a priori et fonctions de coût), et
- calcul de la classe gagnante.

Les différentes parties peuvent se retrouver facilement dans la figure 9. Jusqu'à présent, seules certaines parties de cette architecture ont été étudiées, le but étant d'accroître le plus possible la densité d'intégration pour augmenter les possibilités de parallélisme.

Le circuit décrit ci-dessus devra être considéré comme la partie opérative de l'architecture, indépendante de l'algorithme utilisé. La partie de contrôle, elle, sera dépendante de l'algorithme et sera réalisée de façon digitale. Elle sera utilisée pour cadencer les différentes opérations en utilisant éventuellement des résultats intermédiaires, par exemple la sortie des fonctions noyaux pour les algorithme ROI plutôt que la sortie du second "winner-take-all", uniquement utilisée dans le cas d'estimation de densités de probabilités (des sorties sont prévues à différents niveaux de l'architecture).

Une des parties les plus intéressantes de l'architecture est bien entendu le stockage de valeurs analogiques dans des points mémoires. Il est bien connu que le stockage de valeurs sur des capacités souffre des courants de fuite du transistor qui doit nécessairement y être raccordé pour permettre l'enregistrement d'une valeur. Ces courants de fuite n'étant pas négligeables, la charge stockée sur une capacité de petite taille aura vite fait de varier et donc de perturber les calculs. Un principe original de

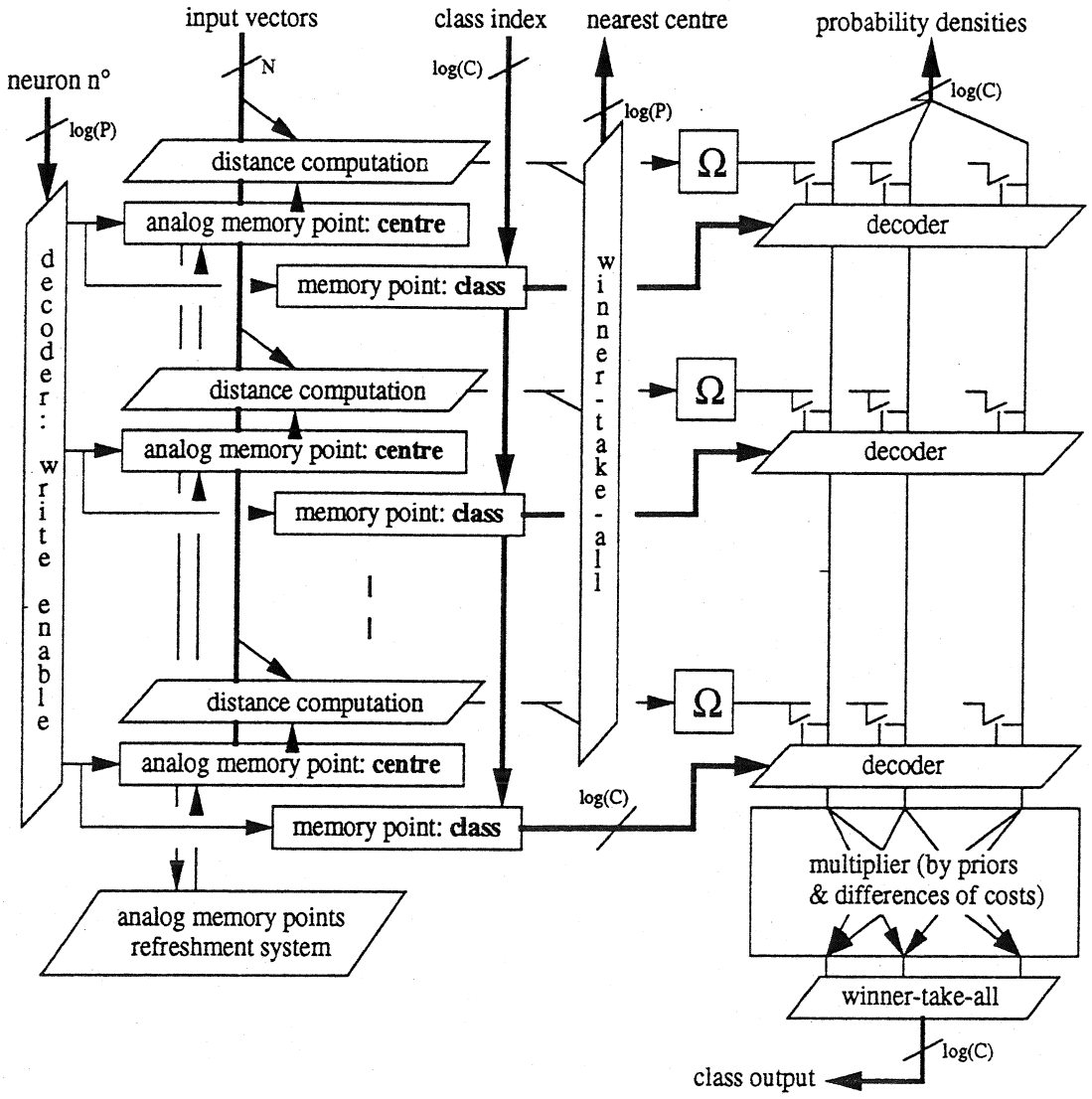

Figure 9: Architecture analogique pour réseaux ROI et à noyaux

Valgo, ISSN 1243-4825
Numéro 94-1 : les processeurs neuronaux - juin 1994

stockage analogique de valeurs a donc été mis au point dans ce sens. Il s'agit de se fixer à priori une précision que l'on veut atteindre pour la mémoire analogique. Une fois cette précision fixée, on peut estimer le temps qu'une valeur restera stockée à cette précision près sur une capacité, tenant compte d'une valeur maximum de courants de fuite admissibles. Chaque valeur analogique mémorisée dans le circuit est alors successivement lue et envoyée dans un convertisseur analogique-digital unique incorporé sur le chip. Supposons maintenant que le sens des courants de fuite est tel qu'il fait toujours diminuer la valeur des charges stockées sur les capacités. Il suffira alors, lorsque la valeur mémorisée dans une capacité est lue et convertie en digital, d'enregistrer à nouveau sur la même capacité une charge correspondant à la valeur immédiatement supérieure dans l'échelle discrète que l'on s'est fixée. De cette façon, si on prend soin de rafraîchir chaque capacité avec une fréquence telle que les courants de fuite entre deux rafraîchissements ne sont pas assez suffisants pour faire diminuer la charge d'un échelon dans l'échelle, on est assuré que celle-ci restera toujours bornée par deux échelons successifs, et donc gardera la précision souhaitée. Une description de la mémoire analogique réalisée à l'UCL peut être trouvée dans [4]; on peut néanmoins signaler qu'une précision de 7 bits a été réalisée dans une technologie CMOS bulk standard, avec une durée entre deux rafraîchissements de la même cellule de 18.9 ms.

A l'heure actuelle, des essais sont effectués pour essayer d'améliorer les caractéristiques des circuits analogiques dans deux directions: d'une part une réduction des courants de fuite dans les transistors afin d'améliorer la mémorisation de valeurs analogiques, et d'autre part l'amélioration des propriétés d'appariement entre composants sur un même circuit. Deux transistors réalisés sur un même circuit et soumis aux mêmes conditions sont en effet susceptibles d'avoir des comportements différents (variation de quelques pourcents dans les courants), et cet effet est bien entendu indésirable si l'on veut utiliser la loi d'Ohm des courants pour sommer les valeurs provenant des différents noyaux. La technologie CMOS SOI (Silicon-On-Insulator), étudiée dans le cadre de ce projet, semble très prometteuse quant à ces deux caractéristiques; des mesures préliminaires ont montré un appariement de transistors nettement meilleur que dans une technologie CMOS conventionnelle (bulk), et les courants de fuite naturellement réduits par ce type de technologie permettent d'entrevoir la possibilité de réaliser des points mémoires analogiques de précision supérieure à ceux réalisés en CMOS bulk.

6 Conclusion

Le projet ESPRIT ELENA fait partie de la catégorie de projets "Basic Research" de la Communauté Européenne. Il tend à développer l'état de l'art dans les domaines des réseaux de neurones évolutifs et de la classification. Afin de tirer pleinement parti des avantages des réseaux de neurones artificiels en termes de parallélisme et de vitesse de traitement, une partie importante du projet concerne le développement d'architectures intégrées implémentant les algorithmes développés.

Diverses solutions sont étudiées dans le projet ELENA. Suivant le type d'algorithme à implémenter, des solutions digitales ou analogiques sont présentées. Les premières font appel à des technologies éprouvées, les secondes nécessitent pour certains points le développement de nouvelles cellules (points mémoires analogiques), ou utilisent de

nouvelles technologies (Silicon-On-Insulator). Le présent article donne un aperçu des différentes approches étudiées dans le cadre du projet. Il va de soi qu'au cours du développement de ces architectures, leurs performances seront comparées à celles des solutions plus classiques (logiciel sur station de travail, transputers,...) afin de cerner les types d'application dans lesquelles chacune des solutions sera la plus appropriée.

7 Equipes de travail

Nous tenons à remercier ici tous ceux qui participent au projet ESPRIT ELENA, en particulier son coordonateur, le Prof. Christian Jutten, de l'INPG, les chercheurs des équipes de l'UCL (P. Thissen et J.-L. Voz) et de l'UPC (F. Castillo, J. Madrenas et M. Moreno) mais aussi tous ceux qui participent aux autres tâches entreprises dans ce projet et qui contribuent donc à sa réussite.

References

[1] F. Castillo and J. Cabestany. A VLSI neural net architecture - a proposal. In *Proceedings of Neuro-Nimes 1990*, pages 515–523, Paris, France, November 1990. EC2.

[2] F. Frean. The upstart algorithm: A method for constructing and training feedforward neural networks. In *Neural Computation 2*, 1989.

[3] C. Jutten, M. Verleysen, et al. Deliverable r1-c1-p - axis c: Hardware implementations. Technical report, Elena-NervesII ESPRIT-BRA project, 1993.

[4] D. Macq, M. Verleysen, P. Jespers, and J.D. Legat. Analog implementation of a Kohonen map with on-chip learning. *IEEE Transactions on Neural Networks*, 4(3):456–461, May 1993.

[5] M. Mezard and J.P. Nadal. Learning in feedforward layered networks: The tiling algorithm. In *J. of Physics A22*, 1989.

[6] J.M. Moreno, F. Castillo, and J. Cabestany. Hardware implementation of piecewise linear separation incremental algorithms. In *Proceedings of Neuro-Nimes 1993*, pages 199–208, Paris, France, October 1993. EC2.

[7] D.L. Reilly, L.N. Cooper, and C. Elbaum. A neural model for category learning. *Biological Cybernetics*, 45(1):35–41, 1982.