Exploitation du concept de tolérance aux fautes des réseaux de neurones pour la résolution de problèmes d'optimisation

Daniel CHILLET, Antoine EICHE, Sébastien PILLEMENT, Olivier SENTIEYS

CAIRN - IRISA/INRIA - Université de Rennes 1
ENSSAT, 6 rue de kerampont, BP 80518, 22305 Lannion
Daniel.Chillet@irisa.fr

Résumé – L'une des problématiques importantes qui découle de l'évolution technologique concerne l'apparition de défauts de plus en plus nombreux au sein des systèmes conçus. Pour palier à ces défauts, la définition d'architectures tolérantes aux fautes est une piste qu'il faut explorer. Les caractéristiques intrinsèques des réseaux de neurones font d'eux un bon candidat pour ce type d'architecture. Cet article propose de montrer, sur un cas simple, comment un réseau de neurones peut effectivement supporter des défaillances tout en conservant sa propriété de convergence.

Abstract – One of the main problem of the technology evolution concerns the increase of number of defaults during the system design and fabrication. To prevent these defaults, fault tolerance architecture must be defined. The intrinsic characteristics of the neural networks place them as a good candidate for this type of architecture. This paper shows, for a simple case, how a neural network can convergence even if several faults can occur in the network.

1 Introduction

L'évolution technologique de fabrication des circuits intégrés conduit petit à petit vers des technologies si fines qu'il devient difficile de garantir une homogénéité des transistors sur l'ensemble de la surface de silicium. Cette évolution pousse alors les concepteurs à minorer les performances des systèmes pour tenir compte de cette variabilité en se basant sur le pire cas. Malheureusement, cette variabilité va se traduire progressivement par des cas extrêmes conduisant au non fonctionnement de certains transistors du système, provoquant alors la disparition de certaines fonctionnalités ou un défaut plus général du système. L'une des questions qui se pose alors est de savoir comment contourner ce type de problème et parvenir à assurer un fonctionnement malgré l'apparition de fautes. Il est bien entendu possible d'intégrer des techniques de détection et de corrections d'erreurs, mais il va devenir de plus en plus crucial d'intégrer des techniques de tolérance aux fautes dans ces systèmes. Cette évolution amène aujourd'hui les concepteurs à proposer des architectures innovantes intégrant directement cette caractéristique.

Parmi l'ensemble des solutions possibles, il en est une qui est connue pour disposer implicitement de cette caractéristique, il s'agit des réseaux de neurones. En effet, à l'instar du cerveau humain, la défaillance d'un ou plusieurs neurones dans un système global peut généralement être corrigée par les autres neurones. Sans disposer de techniques de détection de fautes et de correction d'erreurs, les réseaux de neurones disposent en effet de cette capacité à produire des solutions valides même en présence de dysfonctionnements internes. Pour parvenir à rendre tolérant aux fautes les réseaux de neurones, deux techniques peuvent être utilisées : soit des neurones cachés sont ajoutés, soit la méthode d'apprentissage est modifiée pour tenir compte des fautes qui vont survenir [1, 2]. L'étude que nous proposons dans cet article a pour objet de montrer que les réseaux de neurones de type Hopfield sont tout à fait en mesure d'assurer la résolution d'un problème d'optimisation, et cela malgré des défaillances au sein même de la structure du réseau de neurones. Cette caractéristique de tolérance aux fautes nous intéresse particulièrement en vue de l'appliquer à l'un des éléments constituant un *System-on-Chip, SOC*, à savoir la partie contrôle de celui-ci et plus particulièrement l'ordonnanceur de tâches [3, 4]. L'organisation de cet article est la suivante, la section 2 présente les principes de modélisation d'un problème d'optimisation par une structure de type réseaux de neurones de Hopfield. La section 3 présente les fautes qui peuvent survenir dans une structure de type réseaux de neurones et définit alors le modèle de fautes des neurones. La section 4 montre comment, malgré l'apparition de fautes au sein des réseaux de neurones, ceux-ci peuvent toutefois produire des solutions valides. La définition du nombre de fautes supportées, en fonction des fautes qui se produisent, est alors donnée et permet de définir deux contraintes sous lesquelles le réseau continue à converger vers une solution valide. La section 5 présente des résultats de simulation permettant de préciser le comportement d'un réseau de neurones résolvant le problème d'ordonnancement de tâches sur architecture monoprocesseur. Finalement, la section 6 conclue cet article et présente quelques perspectives.

2 Modélisation d'un problème d'optimisation à base de réseaux de neurones

Comme il a été démontré dans [5], un problème d'optimisation peut être résolu par une structure de type réseau de neurones, et notamment par une structure dite de Hopfield [6]. La particularité de ce type de réseaux de neurones est sa capacité à converger vers des états correspondant à des solutions au problème d'optimisation modélisé [7, 8]. La mise en place d'un réseau de neurones pour résoudre un problème d'optimisation passe par les trois étapes suivantes :

- Il s'agit tout d'abord de représenter le problème à résoudre à l'aide d'une structure de réseau de neurones ;
- Sur la base de cette représentation, il faut définir ce qu'est une solution au problème posé. La définition des solutions au problème permet de calculer les poids de connexions entre les neurones et de définir les énergies à positionner sur leur entrée ;
- Finalement, il s'agit de lancer la convergence du réseau, et de laisser celui-ci se stabiliser sur un état qui représentera une solution.

La convergence est réalisée par une évaluation aléatoire de l'état des neurones. L'état x_i d'un neurone n_i est alors calculé par

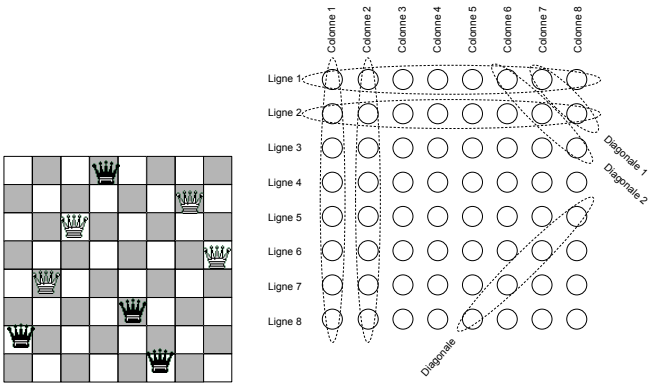
$$x_i \quad = \quad \begin{cases} 1 & \text{si } u_i > 0 \\ 0 & \text{si } u_i \leq 0, \end{cases} \tag{1}$$

avec $u_i = \left(I_i + \sum_{j=1}^{N} x_j \cdot W_{i,j} \right)$ l'état du neurone n_i avant application d'une fonction de seuillage, I_i l'énergie apportée en entrée du neurone n_i et $W_{i,j}$ le poids de connexion entre les neurones n_i et n_j.

Pour parvenir à exprimer la solution d'un problème d'optimisation, la règle dite $k-de-N$ est généralement utilisée [9]. Elle permet de spécifier que parmi un ensemble de N neurones, les solutions recherchées sont celles qui proposent k neurones actifs. Pour illustrer ces propos, nous pouvons donner l'expression du problème classique du positionnement de 8 reines sur un échiquier. Ce problème se présente sous la forme suivante, il s'agit de placer sur un échiquier de taille 8×8, 8 reines de telle sorte qu'elles ne puissent pas se prendre l'une l'autre. Un exemple de placement des 8 reines respectant ces contraintes est présenté sur la figure 1.a. Pour ce problème, la représentation du problème consiste à placer un neurone par case de l'échiquier et à considérer que lorsqu'un neurone est actif cela signifie qu'une reine est placée sur la case correspondante. Puis la définition d'une solution au problème posé consiste à dire qu'il faut :

- pour chaque ligne de neurones, un neurone actif parmi les 8 neurones de la ligne ;
- pour chaque colonne de neurones, un neurone actif parmi les 8 neurones de la colonne ;
- pour chaque diagonale de neurones, zéro ou un neurone actif parmi les neurones de la diagonale.

Ces règles sont partiellement représentées sur la figure 1.b.



a) b)

FIGURE 1 – Problème du positionnement de 8 reines sur un échiquier de taille 8×8. a) Solution possible ; b) Modélisation à base de réseau de neurones de Hopfield.

Cette règle $k-de-N$ est écrite comme une fonction d'énergie du réseau de neurones et après transformation elle permet de définir à la fois les entrées et les connexions des neurones. La fonction d'énergie est définie par

$$E = \left(\sum_{i=0}^{N} x_i - k \right)^2 . \tag{2}$$

Cette fonction d'énergie se traduit par les poids des entrées et des connexions entre neurones selon

$$\begin{aligned} T_{ij} &= -2 \cdot \overline{\delta_{i,j}} \qquad \forall i, \forall j \\ I_i &= 2k - 1 \qquad \forall\, i \end{aligned} \tag{3}$$

3 Modèle de fautes au sein d'un réseau de neurones de Hopfield

Pour illustrer les capacités de tolérances aux fautes des réseaux de neurones, nous étudions dans un premier temps la robustesse de la règle $k - de - N$ appliquée à un ensemble de neurones subissant des défaillances. Soit un ensemble de N neurones sur lequel une règle $k - de - N$ est appliquée. Sur une tel réseau, le nombre de neurones est égal à N, le nombre d'entrées du réseau est égal N et le nombre de connexions entre neurones est égal à $N \times (N-1)$. Au sein de ce réseau, les différentes fautes qui peuvent apparaître peuvent être de plusieurs natures :

- faute sur la fonctionnalité du neurone lui même, qui est figée à une valeur donnée. On parlera alors de faute de « collage » correspondant soit à un état actif permanent soit à un état inactif permanent du neurone ;
- faute sur une entrée d'un neurone, cette entrée pouvant amener soit trop d'énergie conduisant le neurone à toujours rester dans l'état actif, soit amenant trop peu d'énergie ne permettant jamais au neurone de devenir actif. Si on considère que le neurone dont l'entrée est défaillante est au moins évalué une fois, alors ces deux cas se ramènent

à une faite de « collage » , à savoir un état actif ou inactif permanent en fonction de la défaillance survenant sur l'entrée du neurone ;
– faute sur une connexion entre deux neurones, cette connexion pouvant amener soit trop d'énergie négative conduisant le neurone évalué à devenir inactif, soit pas assez d'énergie négative conduisant à considérer que l'on est connecté à un neurone toujours inactif. Si l'on se place du point de vue du neurone évalué à un instant donné, une défaillance sur l'une de ces connexions revient à considérer qu'il est connecté à une neurone figé dans l'état actif ou inactif en fonction de la défaillance.

En conclusion, quelles que soit les défaillances qui peuvent survenir dans le réseau, celles-ci peuvent se ramener à l'étude du comportement de la convergence lorsque des neurones sont « collés » dans l'état actif ou dans l'état inactif. C'est donc ce modèle de fautes qui est retenu pour cette étude.

4 Maintien de la convergence d'un réseau de Hopfield malgré les défaillances

Soit un ensemble ε_N composé de N neurones sur lequel une règle $k-de-N$ est appliquée, et soit un sous ensemble $\varepsilon'_{N_{D_1}}$ composé de N_{D_1} neurones défaillants et collés à 1, avec $\varepsilon'_{N_{D_1}} \subset \varepsilon_N$. Alors la fonction d'énergie relative à la règle $k-de-N$ (eq 2) appliquée à l'ensemble ε_N devient

$$\begin{aligned} E &= \left(\sum_{xi\in\varepsilon_N} x_i - k\right)^2 \qquad (4)\\ &= \left(\left(\sum_{xi\in\varepsilon_N-\varepsilon'_{N_{D_1}}} x_i + \sum_{xi\in\varepsilon'_{N_{D_1}}} x_i\right) - k\right)^2. \end{aligned}$$

Les neurones $x_i \in \varepsilon'_{N_{D_1}}$ étant collés à l'état actif (c'est-à-dire dans l'état 1), alors $\sum_{xi\in\varepsilon'_{N_{D_1}}} x_i = N_{D_1}$. Sachant que seuls les neurones contenus dans l'ensemble $\varepsilon_N - \varepsilon'_{N_{D_1}}$ peuvent changer d'état, pour que cette fonction d'énergie puisse être minimisée et ramenée à zéro, il faut que le terme $\sum_{xi\in\varepsilon'_{N_{D_1}}} x_i - k$ soit négatif, c'est-à-dire que $N_{D_1} - k \leq 0$. Cette contrainte s'écrit donc

$$C_1 : \qquad N_{D_1} \leq k \qquad (5)$$

et indique qu'il ne faut pas que le sous ensemble de neurones défaillants $\varepsilon'_{N_{D_1}}$ contienne plus de k neurones. Le même raisonnement peut être mené pour des défaillances de collage dans l'état inactif des neurones. Soit un sous ensemble $\varepsilon'_{N_{D_0}}$ composé de N_{D_0} neurones défaillants et collés à 0, avec $\varepsilon'_{N_{D_0}} \subset \varepsilon_N$. Alors la fonction d'énergie relative à la règle $k-de-N$ appliquée à l'ensemble ε_N s'écrit alors :

$$\begin{aligned} E &= \left(\sum_{xi\in\varepsilon_N} x_i - k\right)^2 \qquad (6)\\ &= \left(\left(\sum_{xi\in(\varepsilon_N-\varepsilon'_{N_{D_0}})} x_i + \sum_{xi\in\varepsilon'_{N_{D_0}}} x_i\right) - k\right)^2. \end{aligned}$$

Les neurones $x_i \in \varepsilon'_{N_{D_0}}$ étant collés à l'état inactif (c'est-à-dire dans l'état 0), alors $\sum_{xi\in\varepsilon'_{N_{D_0}}} x_i = 0$. Sachant que seuls les neurones contenu dans l'ensemble $\varepsilon_N - \varepsilon'_{N_{D_0}}$ peuvent changer d'état, pour que cette fonction d'énergie puisse être minimisée et ramenée à zéro, il faut que le terme $\sum_{xi\in(\varepsilon_N-\varepsilon'_{N_{D_0}})} x_i - k$ puisse être annulé, pour assurer cela, il faut que l'ensemble $\varepsilon_N - \varepsilon'_{N_{D_0}}$ contienne au moins k éléments. Cela se traduit par $Card(\varepsilon_N) - Card(\varepsilon'_{N_{D_0}}) \geq k$, soit $N - N_{D_0} \geq k$. La contrainte s'écrit donc

$$C_2 : \qquad N_{D_0} \leq N - k \qquad (7)$$

Toute combinaison de défaillances pour laquelle les contraintes C_1 et C_2 sont respectées permet d'assurer la convergence du réseau vers une solution possible au problème de l'activation de k neurones parmi N.

5 Résultats

La règle $k-de-N$ est aussi celle qui est utilisée dans le cas qui nous intéresse plus particulièrement dans cet article et qui consiste à proposer un ordonnanceur de tâches tolérant aux fautes. L'ordonnancement de tâches est effectivement un problème d'optimisation pour lequel les réseaux de neurones de Hopfield sont l'une des solutions possibles d'implémentation. Dans ce cas de figure, la représentation du problème d'ordonnancement est simplement réalisée en plaçant un neurone par tâche et par cycle d'ordonnancement, tel que représenté sur la figure 2 pour un exemple d'ordonnancement de 4 tâches à exécuter sur 15 cycles d'ordonnancement. Sur cette modélisation

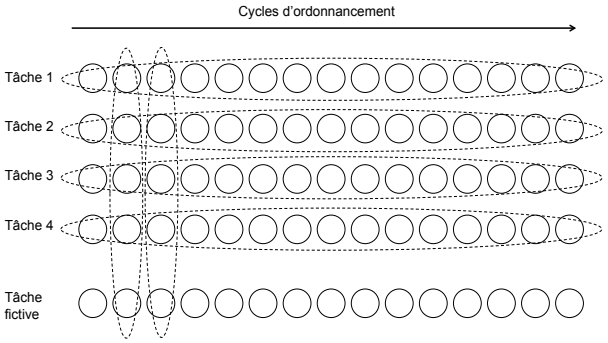


FIGURE 2 – Représentation du problème d'ordonnancement de tâches sur une architecture monoprocesseur.

du problème, une solution d'ordonnancement de tâches possible répond aux règles suivantes :

– chaque ligne de neurones (lignes de neurones correspondant à une tâche) doit avoir exactement C_i neurones actifs (avec C_i la charge de la tâche sur le processeur) ;
– chaque colonne de neurones (colonne de neurones correspondant à un cycle d'ordonnancement) doit avoir au plus un neurone actif.

Pour simplifier la représentation, une tâche fictive est généralement ajoutée, elle permet de simplifier l'application des règles sur les colonnes en indiquant que chaque colonne doit avoir exactement un neurone actif. L'activité d'un neurone de la tâche fictive indiquant alors que le processeur n'a pas de tâche à exécuter pour le cycle considéré. Dans la figure 3, nous montrons le nombre de défaillances de type collage à 0 ou collage à 1 qu'un réseau composé de 100 neurones est capable de supporter. Ces valeurs de défaillances sont dépendantes de k, c'est-à-dire du nombre de neurones actifs souhaités pour la convergence. La zone grisée correspond à la zone pour laquelle le nombre de défaillances ne remet pas en cause la convergence du réseau.

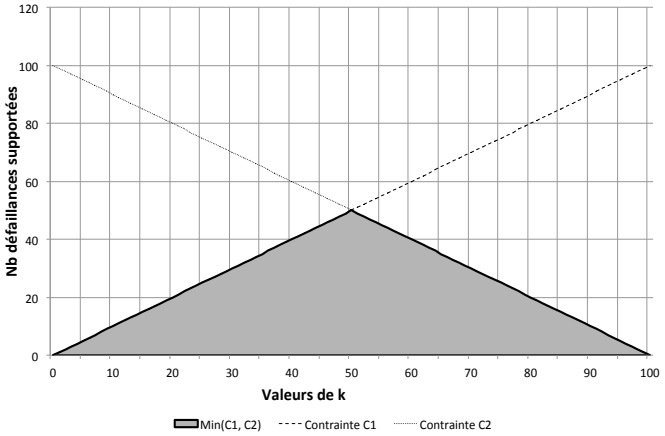


FIGURE 3 – Nombre de défaillances supportées par une règle $k - de - N$ pour $N = 100$ et k variant de 0 à 100.

Finalement, la figure 4 illustre l'impact des défaillances sur le temps de convergence. Le réseau utilisé pour ces résultats est constitué de 100 neurones, une règle $50 - de - 100$ lui est appliquée, et le nombre de défaillances varie de 0 à 50 défaillances de collage à 1, puis de collage à 0, puis de collage à la fois à 1 et à 0. La figure montre que lorsque le nombre de fautes reste relativement faible (quelques neurones défaillants), alors le réseau parvient sans difficulté à converger vers une solution valide. On notera aussi que le nombre de cycles pour atteindre la convergence est peu impacté, notamment lorsque les défauts sont à la fois des collages à 1 et des collages à 0, ce qui est plus probable que le cas de collage uniquement à 0 ou uniquement à 1.

6 Conclusion

Les travaux présentés dans cet article illustrent les capacités de tolérance aux fautes des réseaux de neurones de Hopfield. Ces réseaux sont capables de produire des solutions à

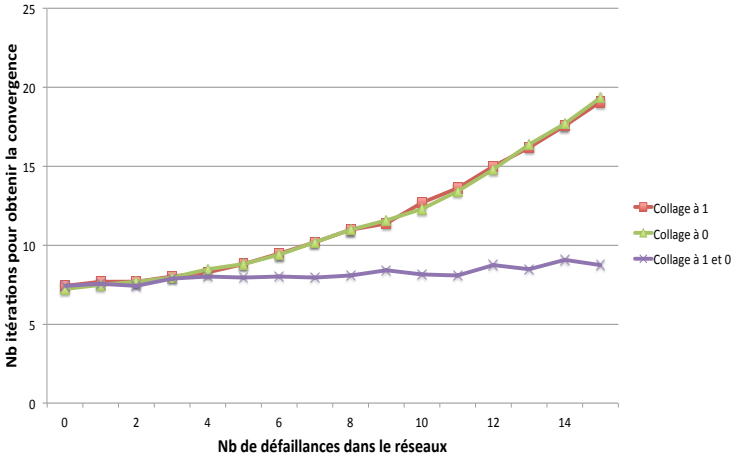


FIGURE 4 – Impact sur le temps de convergence en cas de défaillances au sein du réseau de neurones.

un problème d'optimisation pour peu que l'on sache modéliser le problème et définir l'ensemble des solutions. Dans cette étude nous avons montré que la défaillance d'une partie de la structure du réseau de neurones ne remet pas en cause la production de solutions valides. Nous avons défini le modèle de fautes qui peut survenir dans une structure de Hopfield, et nous avons illustré la capacité de tolérance aux fautes au travers d'une structure de réseau de neurones simple sur laquelle nous appliquons une règle de convergence particulière. Nous avons alors montré sous quelles conditions de défaillance ce type de réseaux peut continuer à produire des résultats valides. Un exemple de convergence de réseau de neurones en présence de fautes a été présenté et nous avons montré que la convergence est peu sensible au nombre de défaillances au sein du réseau, lorsque ce nombre de fautes reste raisonnable.

Références

[1] Z.-H. Zhou and S.-F. Chen, "Evolving fault-tolerant neural networks," *Neural Computing & Applications*, vol. 11, pp. 156–160, 2003, 10.1007/s00521-003-0353-4. [Online]. Available : http ://dx.doi.org/10.1007/s00521-003-0353-4

[2] N. Kamiura, T. Isokawa, and N. Matsui, "On improvement in fault tolerance of hopfield neural networks," in *Test Symposium, 2004. 13th Asian*, november 2004, pp. 406 – 411.

[3] D. Chillet, S. Pillement, and O. Sentieys, *Algorithm-Architecture Matching for Signal and Image Processing, Springer*, ser. Lecture Notes in Electrical Engineering. Springer Netherlands, 2010, vol. 73, ch. RANN : A Reconfigurable Artificial Neural Network Model for Task Scheduling on Reconfigurable System-on-Chip, pp. 117–144.

[4] D. Chillet, A. Eiche, S. Pillement, and O. Sentieys, "Real-time scheduling on heterogeneous system-on-chip architectures using an optimized artificial neural network," *Journal of Systems Architecture*, vol. In Press, Accepted Manuscript, pp. –, 2011. [Online]. Available : http ://www.sciencedirect.com/science/article/B6V1F-524FSKG-1/2/f8cce034de55596f5712443d39108ffb

[5] G. Tagliarini, J. F. Christ, and W. E. Page, "Optimization using neural networks," *IEEE Transactions on Computer*, vol. 40, no. 12, pp. 1347–58, Dec. 1991.

[6] J. J. Hopfield and D. W. Tank, "Neural computation of decisions in optimization problems," *Biological Cybernetics*, vol. 52, pp. 141–152, 1985.

[7] M. Cohen and S. Grossberg, "Absolute stability of global pattern formation and parallel memory storage by competitive neural networks," *IEEE transactions on Systems, Man, and Cybernetics*, vol. 13, no 5, pp. 815–826, 1983.

[8] S. Grossberg, *Studies of Mind and Brain : Neural Principles of Learning, Perception, Development, Cognition and Motor Control*, B. S. in the Philosophy of Science, Ed. D. Reidel Publ. Co., 1988.

[9] C. Cardeira, M. Silva, and Z. Mammeri, "Handling precedence constraints with neural network based real-time scheduling algorithms," in *Proc. of the 9th Euromicro Workshop on Real Time Systems*, Toldeo, Spain, Jun. 1997, pp. 207–214.