EFFICIENT IMPLEMENTATION ON MULTIPROCESSORS : THE PROBLEM OF SIGNAL PROCESSING APPLICATIONS MODELLING

Laurent Kwiatkowski, Fernand Boéri, Jean-Paul Stromboni
Laboratoire d'Informatique Signaux Systèmes, UNSA - URA 1376 C.N.R.S.
41, Boulevard NAPOLEON III - F 06041 Nice cedex - FRANCE
Tel : +33 21 79 61; fax : +33 21 20 54
e-mail : kwiatkow@alto.unice.fr

ABSTRACT

In signal processing area, applications involve a large amount of computation, suggesting the use of multiprocessors to speed up processing. However, obtaining good performance is not easy because the machine should take advantage of the potential parallelisms of the studied application. That is why several parallel implementation methods using mapping and scheduling algorithms has been developped. One of development shells aims is application partitioning so that every part will be processed by a different processor, like SynDEx [1] or Ptolemy[2]. These shells use some graph models to exhibit both potential parallelisms of the application and the available multiprocessor parallelisms [3], but the task granularity problem is not considered when the application is modelized.

The purpose of this paper is to emphasize the problem of the task granularity when the application is modelized by means of a graph and to study the impact on speedup. As a solution for this problem, this paper presents an original implementation method based on the variation of the granularity and regular application size.

1 INTRODUCTION

In order to research the best parallel implementation, the application is often modelized by a dataflow graph called software graph, where each vertex stands for a task and each edge is a communication. The first part describes the backpropagation learning application. This application has been often used in patterns recognization and classification.

Several models of the same application are possible. They differ by the task granularity defined as the ratio of the mean task execution time to the total application processing time. Exploiting the whole potential parallelism requires a fine grain but leads to combinatorial explosion when the size of the application increases. This combinatorial explosion could be avoided by increasing granularity. Study of parallel implementation methods shows that development shells use a task granularity depending on the size of the application. However, the task granularity problem is not considered and potential parallelisms would then be decreased. The second part of this paper shows by simulation of the particular application, that the parallel performance (speedup) depends on the task granularity. Results of an implementation on MIMD multiDSP machine according to the granularity are presented.

Therefore, that is not very difficult to research the implementation given the best parallel performances if application modelling decrease potential parallelisms. The last part defines the way to increase granularity so that the implementation on the multiprocessor could take advantage of the potential parallelisms that appear when fine-grained is used.

2 APPLICATION CONSIDERED

Simulation of the backpropagation learning algorithm for a fully connected multilayered neural network on a multiprocessors system was considered [4]. Connexionist algorithms contain computation phases with significant potential parallelisms (simultaneous calculations and data transfers). They suggest the use of a multiprocessors implementation. The application we are focusing on is the most popular one and has been widely used in vision, speech [5], sonar, radar, signal processing and robotics applications.

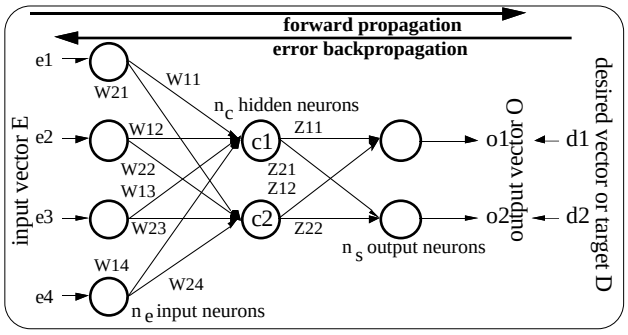

Fig 1 : A fully connected threelayered network [4-2-2]

A three layers network is considered in Fig1. The first layer or input layer contains n_e cells, the second or hidden layer contains n_c neurons and the third or output layer contains n_s neurons. Each neuron in the last two layers is connected to all the neurons in the next layer. Associated with each neuron i is an activation value (c_i for hidden cells and o_i for output cells) and attached to each connection is a synaptic weight (Wji connects input cell i to hidden cell j and Zji connects hidden cell i to output cell j). Supervised learning is a procedure to find a set of weights in a neural network according to training input/output pattern pairs E/D so that given an input pattern, the output pattern produced by the network is sufficiently close to the desired output pattern.

The backpropagation learning algorithm [6] uses three steps : forward propagation, backpropagation error and weight update as shown the frame in the following page. In the forward propagation step, the activation value is obtained by summing up weight connections and input products and by using a nonlinear sigmoïd function of the form $sig(x) = (1+e^{-x})^{-1}$. The second step involves a comparison of actival and desired output patterns and propagation of error terms δ_k and δ_j. Weights update is performed in the third step (τ is a learning rate).

presented example : $E = e_1 \ldots e_i$ for $i = 1 .. 4$

1)*forward propagation :* $c_i = sig(\Sigma_j Wij . e_j)$ for $j = 1 .. 2$

$o_k = sig(\Sigma_j Zkj . cj)$ for $k = 1.. 2$

Comparison between O and the desired output D

2)*Gradient error backpropagation :*

correction $_{output}$: $\delta_k = (o_k - d_k) . o'_k$,

correction $_{hidden}$: $\delta_j = [\Sigma_k (Zkj . \delta_k)] . c'_j$

3)*synaptic weights update :*

$Zkj(t) = Zkj(t-1) + \tau . \delta_k . cj$,

$Wji(t) = Wji(t-1) + \tau . \delta_j . ei$

3 THE EFFECT OF TASK GRANULARITY

In many examples, the software graph of the learning algorithm is drawn from the neural network which has only one task including one neuron. Usually, a horizontal partitioning is proposed. In the Fig1 case, it leads to four processors. Each processor keeps in its private memory activation values, error value and input and output weight connections of neurons which were assigned to it. Task granularity is important because it includes all the neuron process. In order to detect a high degree of potential parallelisms, a fine-grained representation is selected as shown in Fig2 (task granularity is the elementary operator level).

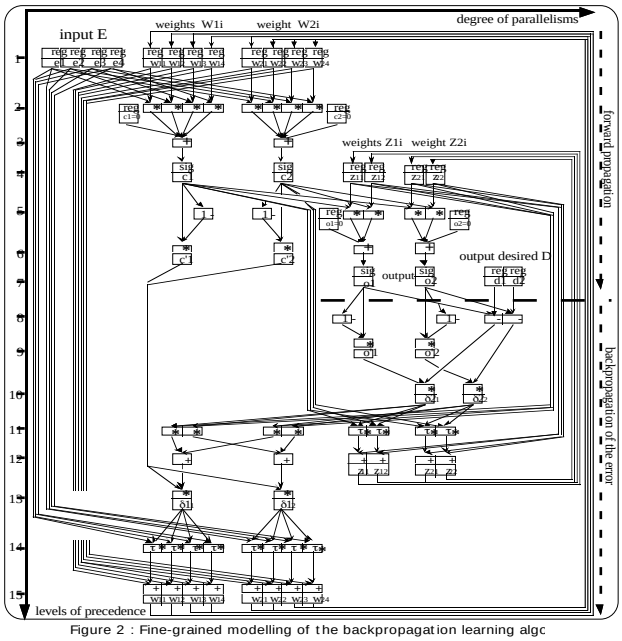

Figure 2 : Fine-grained modelling of the backpropagation learning algo

We show by simulation on MIMD multiDSP 96002, with the neural network application that potential speedup depends on granularity. Processor DSP 96002 [7] is able to compute operation 32 bits floating point. It uses a DMA memory in order to transfer of data restricted the intervention of the DSP core. Registers of the DMA supervisor are accessed by writing / reading and are on memory space dedicated to inputs/outputs. Then, the architecture focused allows an overlapping between data transfers and computation. Ideal performances are obtained with a communication time equal to zero and a no fixed number of processors. The Fig3 shows speedup results with the fully connected threelayered network [4-2-2]. Speedup variations achieved beyond 8 processors are insignificant but efficiency decreases swiftly. A 1,37% speedup drop allows an increase in efficiency from 20% to 51%. This establishment does not allow to determine a number of processors given a trade-off between speedup and efficiency.

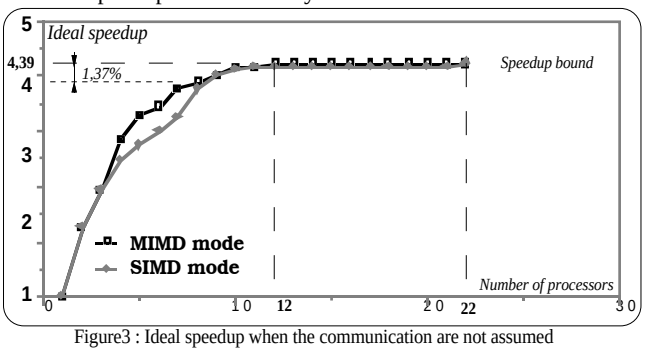

Figure3 : Ideal speedup when the communication are not assumed

Speedup is 4,39 with 12 processors (efficiency is 36,6%). With four processors, *ideal performances* are a *speedup* equal to **3,25** and a *efficiency* equal to **81,3%**. When the communication time is assumed and a horizontal partitioning is used, the simulation of the backpropagation learning algorithm on the fully connected four processors obtains a **1.35** speedup with 4 processors (efficiency is **33,4%**). Implementation of the application fined-grained modelling requires a distribution of tasks to the parallel resources. Simulated annealing [8] insure the best load balance and consequently the maximum possible speedup with the considered machine. Using this heuristic gives a 2.2 maximum speedup with 12 processors (communication times are assumed and efficiency is 18,3%). A trade-off between speedup and efficiency (20% of the maximum speedup) when the speedup is 1,8 and efficiency is 45% with four processors, as shown in Fig4. Results are compared to the horizontal partitioning.

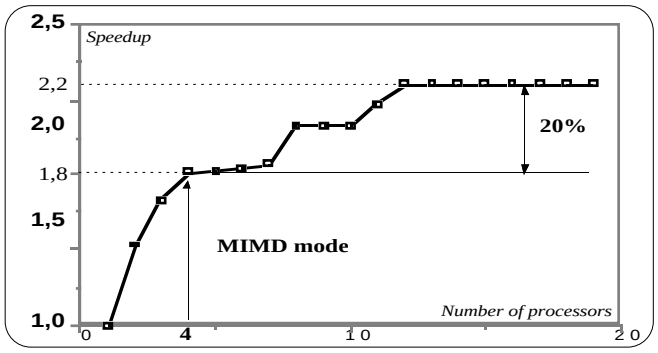

Figure4 : Research of an efficiency/speedup trade-off

Then, application modelling is important to obtain maximum speedup performance, i.e. to exploit all the potential parallelisms. Development shells should take into account the task granularity aspect for description and should not choose a granularity so that the potential parallelisms are decreased. Performance variations could be important : 25% in our case.

4 THE COMBINATORIAL EXPLOSION

Fine-grained modelling is limited to the combinatorial explosion when the application size is important. For example, the backpropagation learning application graph associated to a [16-6-2] multilayered network is composed of 24 vertices and 108 edges. Fine-grained modelling requires 518 vertices and 960 edges. For these regular applications, a method defines the way to increase the task granularity so that the research of an efficient implementation on the multiprocessor architecture could take advantage of the potential parallelisms that appear when a fine-grained decomposition is used. Task clustering are executed in order to increase the granularity but parallel performances are preserved. From

the implementation of the application fined-grained modelling, the Gantt processing graph is studied.

The Gantt graph determines when tasks are processed by the same processor and the communication requirement. Then this graph supplies a simple method to research clusters which are not decrease parallel performances. Fig 5 shows the medium grained modelling of the application after clustering.

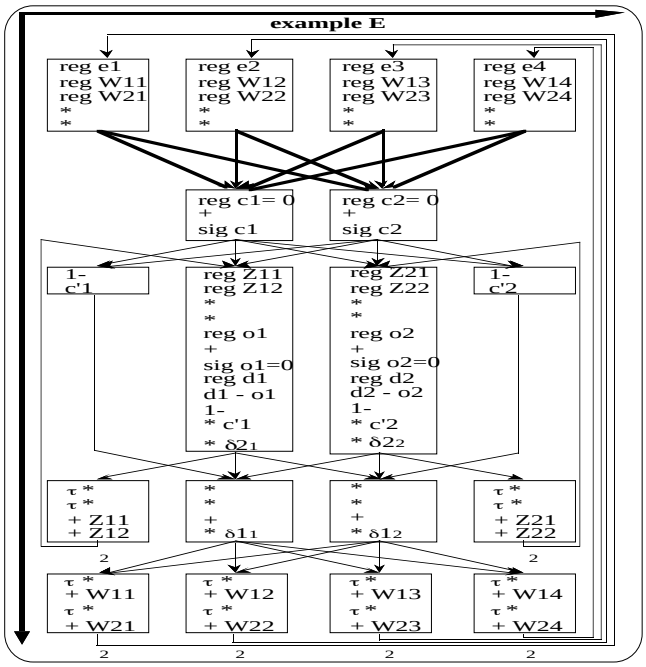

Figure5 : RPGE [4-2-2] modelling after the granularity increasing

With this medium-grained modelling, the use of simulated annealing in order to research the best implementation does not modify the mapping and scheduling of tasks on the processors. Speedup and efficiency remain unchanging. In order to make sure this model is equivalent to efficient implementation obtained with the use of fine-grained whatever the network size, this method is applied to the [16-6-2] fully connected multilayered neural network. The fined-grained modelling requires 518 vertices and 960 edges. Implementation on MIMD multiDSP gives a trade-off speedup/efficiency with 16 processors : **6,96**/**43,5**%. In a second step, clusters are achieved according to rules obtained with the study of the previous application Gantt graph as shown the Fig6.

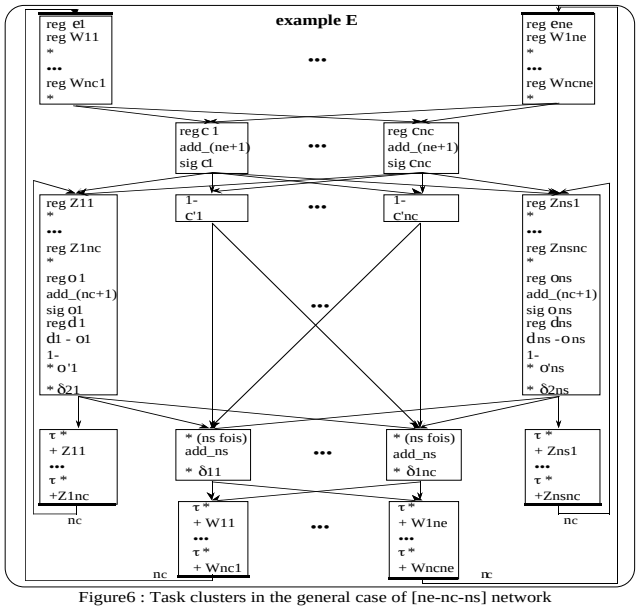

Figure6 : Task clusters in the general case of [ne-nc-ns] network

Implementation of this graph (54 vertices and 260 edges i.e. vertices reduction of 90% and edges reduction of 74%) gives similar speedup/efficiency performances : **6,89**/**42,5**%. Then, This method is applied to the high size neural network application [5] NetTalk, using the backpropagation learning algorithm on a [203-80-26] multilayered network. Fined-grained modelling is difficult because the graph is composed of 76331 vertices and 151674 edges. We increase the granularity without decrease the potential parallelisms : the medium-grained graph is composed of 698 vertices and 41215 edges. Performances variations between a fine-grained modelling and the medium-grained software graph resulting from our method are below 3%. Implementation on MIMD multiDSP gives a trade-off speedup/efficiency equal to **115**/**56,66**%.

5 CONCLUSION

This short paper has presented a problem that is not considered in development shells : the task granularity when application modelling is used to research the implementation on multiprocessor given the best performances. But, simulation of a particular forms recognition application shows according to models that performance variations could be important. Then models choosen by shells decrease potential parallelisms and the research of the best performances is silly. So an alternative is presented in order to take into account all the potential parallelisms; a fined-grained modelling with elementary operator level is considered. First in order to avoid combinatorial explosion, a small application size is selected. Its Gantt graph is studied and cluster rules are defined. This method defines the way to increase the granularity without performances are changed. Second, this medium-grained modelling is applied when the application size is important. The simulation results are better than usual results for this type of regular application. Now a study is conducted in order to detect regularities automatically and to lead the implementation problem to a feasible size and to avoid combinatorial explosion.

References

[1] Y. Sorel, C. Lavarenne et al.,*"Implantation d'algorithmes de traitement d'images sur une architecture multiDSP avec l'environnement d'aide SynDEx"*, Quatorzième colloque GRETSI, Juan-les-Pins, 1993, pages 1019-1022.

[2] Ptolemy 0.5 user's manual, University of California at Berkeley - College of Engineering, Department of Electrical and Computer Sciences Berkeley California - 94.720, 1994.
[3] JP. Stromboni, L. Kwiatkowski, *"Conception d'un modèle pour l'analyse du parallélisme"*, International Workshop On Principles Of PArallel Computing, OPOPAC, Lacanau, FRANCE, Edition Hermes, april 1993, pages 85-99.
[4] R. Lippmann, *"An introduction to computing with neural nets"*, IEEE ASSP MAGAZINE, april 87, pp 4-22.
[5] T-J. Sejnowski, C-R Rosenberg, *"Parallel networks that learn to pronounce english text"*, Complex Systems Publications, Vol 1, N°1, 1987, pages 145-168.
[6] D-E. Rumelhart, G-E. Hinton, R-J. Williams, *"Learning internal representations by error propagation"*, Parallel Distributed Processing : explorations in the microstructure of cognition 1, MIT Press, Cambridge, MA, 1987, pp 318-362.
[7] Motorola,*"DSP 96002, IEEE floating-point dual port processor"*. User-s manuel, 1989.
[8] S. Kirkpatrick, C-D. Gelatt, M-P. Vecchi, *"Optimiza-tion by Simulated Annealing"*, Science, Vol 228, N° 4598, 1983, pages 671-680.

• • •