Efficient Parallel Implementation of Multilayer Backpropagation Networks on SpiNNaker

Xin Jin
School of Computer Science
University of Manchester
Oxford Road
Manchester, UK
jinxa@cs.man.ac.uk

Mikel Luján
School of Computer Science
University of Manchester
Oxford Road
Manchester, UK
mikel.lujan@manchester.ac.uk

Luis A. Plana
School of Computer Science
University of Manchester
Oxford Road
Manchester, UK
plana@cs.man.ac.uk

Alexander D. Rast
School of Computer Science
University of Manchester
Oxford Road
Manchester, UK
rasta@cs.man.ac.uk

Stephen R. Welbourne
School of Psychological Sciences
University of Manchester
Oxford Road
Manchester, UK
stephen.r.welbourne@manchester.ac.uk

Steve B. Furber
School of Computer Science
University of Manchester
Oxford Road
Manchester, UK
sfurber@cs.man.ac.uk

ABSTRACT

This paper presents an efficient implementation and performance analysis of mapping multilayer perceptron networks with the backpropagation learning rule on SpiNNaker - a massively parallel architecture dedicated for neural network simulation. A new algorithm called pipelined checkerboarding partitioning scheme is proposed for efficient mapping. The new mapping algorithm relies on a checker-board partitioning scheme, but the key advantage comes from introducing a pipelined mode. The six-stage pipelined mode captures the parallelism within each partition of the weight matrix, allowing the overlapping of communication and computation. Not only does the proposed mapping localize communication, but it can also hide a part of or even all the communication for high efficiency.

Categories and Subject Descriptors

C.3 [**Computer Systems Organization**]: Special-Purpose and Application-Based Systems

General Terms

Algorithms, Experimentation

1. INTRODUCTION

As with other parallel applications, the challenge in the parallel simulation of multilayer perceptron (MLP) networks with the backpropagation (BP) learning is to understand how to partition and distribute the computational tasks while minimizing communication requirements.

In this paper, the proposed pipelined checker-boarding partitioning (PCBP) scheme cuts the whole weight matrix into small sub-matrices as in the original checker-boarding partitioning (CBP) scheme, enabling the communication to be localized and reducing the number of communication packets. However, in addition to the traditional group of cores which do the vector-matrix computation, in this new scheme, an extra two groups of cores are employed to compute the partial sums and outputs. The three groups of cores are able to work in parallel and produce a six-stage pipeline, allowing the overlap of computation and communication. We present analysis of the parallel execution of the algorithm on the SpiNNaker architecture. The performance of PCBP scheme is evaluated and is compared with results of CBP scheme.

2. CBP SCHEME

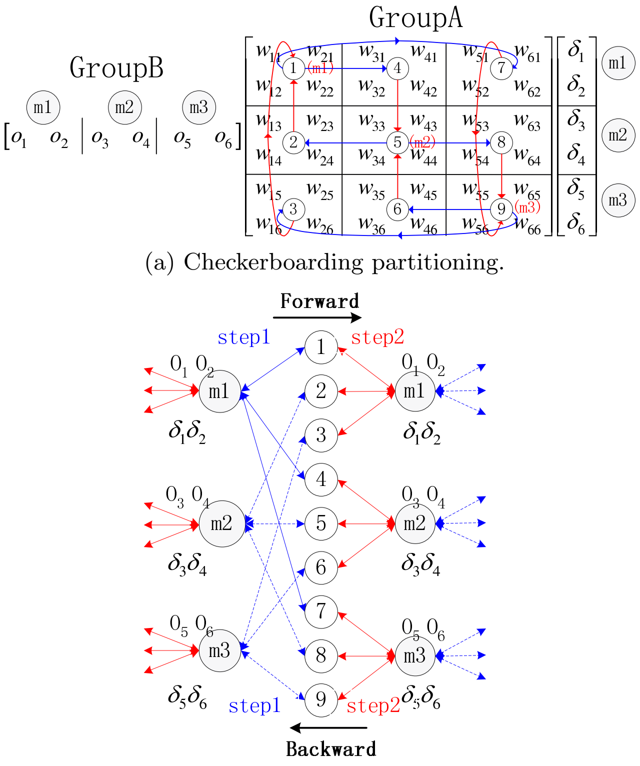
We use the CBP partitioning scheme to split the weight matrix as the starting point of our algorithm. The first step, for simplicity, is to analyse the mapping of neural networks onto a 2D torus topology with one processor per node. Figure 1(a) shows an example of a 6x6 weight matrix mapped onto 9 nodes (or processors) interconnected by a 2D torus. Each processor keeps a 2x2 sub-matrix of weights in its local memory. We assign processors $1-9$ to GroupA responsible for the vector-matrix multiplication. Among those 9 processors in GroupA, we select 3 processors in the main diagonal and name them $m1-m3$. Then we also assign processors $m1-m3$ to GroupB which is responsible for the partial result accumulation and output computation.The corresponding communication pattern is illustrated in Figure 1(b).

3. MAPPING ONTO SPINNAKER

3.1 SpiNNaker

SpiNNaker is a system based on the torus-connected CMPs topology. It is a massively-parallel architecture, comprising multiple identical SpiNNaker chips connected in a 2D torus mesh topology. Each SpiNNaker chip is a multi-core system containing 20 ARM968 processors and a router. Each

CF'10, May 17–19, 2010, Bertinoro, Italy.
ACM 978-1-4503-0044-5/10/05.



(a) Checkerboarding partitioning.

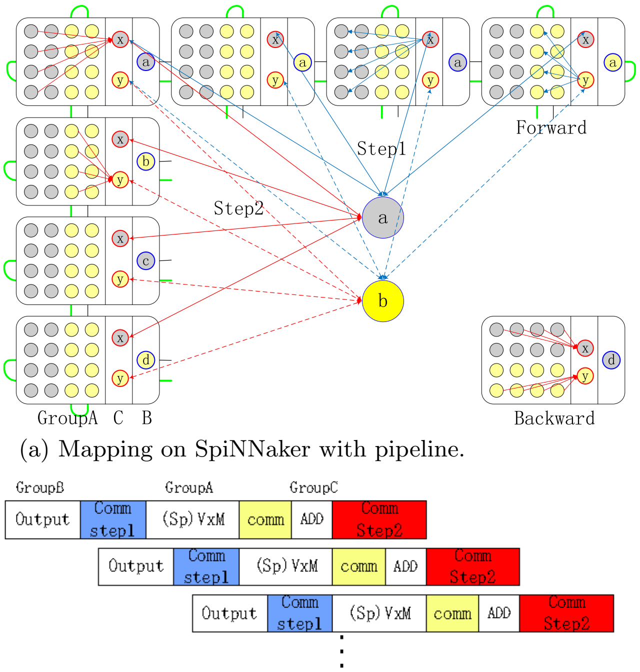
(b) Communication pattern.

Figure 1: CBP partitioning and communication.

processor has a 64KB local private memory. All processors on one chip share an on-chip system RAM and an off-chip SDRAM for information exchange and extended data storage. The on-chip router supports multicast packets.

3.2 Pipelined CBP

The mapping of MLP networks on SpiNNaker using PCBP scheme is illustrated in Figure 2(a). Each rectangle (with rounded corners) represents one SpiNNaker chip. Each circle in a rectangle denotes a processing core. We use 19 processors out of 20 in each chip. Among them 16 (4 by 4) processors are allocated to GroupA, 1 (a/b/c/d) processor is allocated to GroupB and the other 2 (x and y) are allocated to GroupC. In step1, GroupB processors produce outputs and send to GroupC processors. GroupC processors get single node broadcast packets from GroupB processors, and then forward to GroupA processors. In step2, GroupA processors do the vector matrix computation and send reults to GroupC processors with same color. Each GroupC processor receives packets from GroupA processors in two columns (2 by 4 processors) in turn and accumulate partial results, then forward the results to GroupB processors. Notice that what ever the number of chips are there in one column, we only require four GroupB processors in total, each responsible for one column, since there are only four columns of GroupA processors in total. GroupC processors need to send packets to two GroupB processors in the same color in turn (for example processor x sends to processor a and c). The backward phase works exactly the same as the forward phase, but swaps the order of columns and rows. In each chip, the three groups of processors are working in parallel and produce a six-stage pipeline shown in Figure 2(b).



(a) Mapping on SpiNNaker with pipeline.

(b) The six-stage pipeline.

Figure 2: Mapping and the pipeline

4. PERFORMANCE

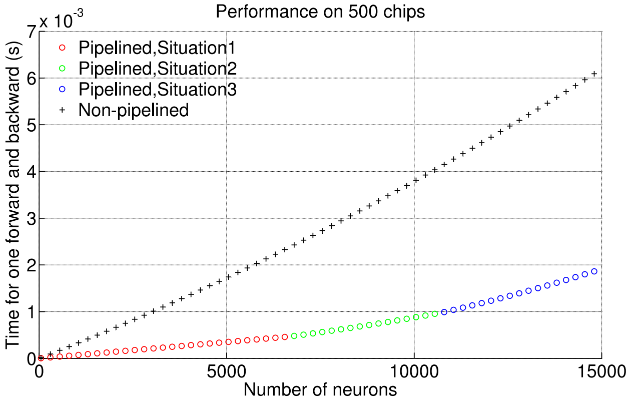


Figure 3: A performance comparison.

We show a performance comparison between the non-pipelined and the pipelined model in Figure 3. Note that the performance curve of the pipelined model is segmented into three situations according to the differences between communication and computation.

5. CONCLUSIONS

This paper shows how to efficient implement MLP networks with the BP rule on SpiNNaker with a new efficient pipelined checker-boarding partitioning scheme.