A Neural Online Triggering System Based on Parallel Processing

J.M. Seixas , L.P. Calôba, A.R. Anjos, B. Kastrup, A.C.H. Dantas and R. Linhares
COPPE/EE/UFRJ, C.P. 68504, Rio de Janeiro 21945-970, Brazil
Fax: 55-21-2906626. e-mail: seixas@lacc.ufrj.br

Abstract

The study of a prototype of the second-level triggering system for operation at LHC conditions is addressed by means of a parallel machine implementation. The 16 node transputer based machine uses a fast digital signal processor acting as a coprocessor for optimizing signal processing applications. A C-language development environment is used for running all applications at ultimate speed. The implementation is based on information supplied by four detectors and includes two phases of system operation: feature extraction and global decision. Feature extraction for calorimeters and global decision processing are performed by means of neural networks. Preprocessing and neural network parameters rest in memory and the activation function is implemented using a look up table. Simulated data for the second-level trigger operation are used for performance evaluation.

I. Introduction

The Large Hadron Collider (LHC) is planned to be operational by year 2005. This is a high event rate colliding machine, as the period between collisions is as low as 25 nanoseconds and luminosity will be high. On the other hand, at LHC the interesting events are expected to be extremely rare, so that event selectivity is mandatory. Therefore, the LHC environment represents a two fold challenge: first, in terms of detector technology, as detectors have to perform very sensitive measurements with high performance and at maximum signal speed (to avoid pileup); second, in terms of online trigger algorithms, as the potential candidates of the physics that is being searched for at LHC have to be extracted from the huge background noise produced at high-rate by the colliding machine.

To address online event validation, a multi-level triggering system is being designed [1]. In the first level, the event rate is reduced to 100 kHz and regions of valuable physics information (known as Regions of Interest, ROIs) are identified in each subdetector. The second-level (LVL2) system acts on events that pass the conditions of the previous level and restricts its attention to ROI information. This approach allows for reducing the requirements in the bandwidth of the LVL2 system. ROI information is stored in readout buffers (ROBs).

One possible approach for the LVL2 system design is to split the data processing into two phases [2]. In the first, variables with physics significance (or features) are extracted from preprocessed data related to a given ROI. The ROI information is assembled by collecting the respective ROBs in which the information has been stored. Next, the global decision phase operates on features related to all ROIs and subdetectors for a given event. The global decision phase correlates the information generated by the different detectors in order to achieve a final (and more refined) decision on keeping or discarding the particular event.

Both feature extraction (for calorimeters) and global decision phases have been suggested to be performed by neural processing [3, 4]. Neural processing can be considered an interesting approach for a number of reasons. It is a fast processing technique that is valuable in computing correlations in a multidimensional space [5]. Pattern recognition ability of neural networks can be fully exploited when the information of the various subdetectors are combined in the global decision phase [6]. Finally, neural networks can be implemented in fast commercial processors at a reasonable cost. In fact, the neural processing for the second-level trigger was successfully implemented on a fast (40 MHz) digital signal processor (ADSP-21060) [4, 7].

Data used in this paper are obtained from Monte Carlo simulations for the four detectors (calorimeter, tracking and muon systems) involved in the LVL2 system [8, 9]. Only events that passed the conditions of a simulated first-level trigger algorithm were used in the analysis. The LVL2 prototype implementation was achieved by using a transputer based parallel machine as a developing platform. The global decision phase and feature extraction for calorimeters were addressed by neural processing. For the other three detectors a simulation of current classical algorithms for feature extraction was used. For this, C codes simulated module interfaces and performed main functionalities, not necessarily handling all constraints.

The next section describes the neural processing techniques and section 3 highlights the main features of the parallel machine. Prototype implementation and performance measurements are addressed by section 4. Conclusions are found in section 5.

II. Neural Processing

The ROI description for calorimeters was based on a 11 x 11 matrix of total deposited energy in the 121 cells (e.m. and hadronic added) of a fine-grained calorimeter system. For this, neural feature extraction was performed by searching preprocessing methods on ROI data capable to reduce the high dimensionability of data input space, and realize efficient electron/jet separation. In this way, both processing speed and discrimination efficiency could be accomplished.

As the outermost cells sample low energy levels but perform efficient discrimination, grouping cells by means of a weighting combination help to boost the contribution of such cells in the ROI based discrimination. The determination of the weighting factor of each grouped sum is integrated to the training phase of the network, so that optimum normalization factors for the grouped sums can be obtained altogether with the network's weighting vectors that minimize the desired cost function (mean squared error) [7].

Figure 1 shows such a possible ROI mapping. Here, cells which belong to a sector have their energies added up to form group sums. The resulting grouped information is fed into the input nodes of a neural classifier. This classifier achieves 97% electron efficiency for less than 7.3% of jets being misclassified as electrons. This fully-connected network comprised five nodes in the hidden layer.

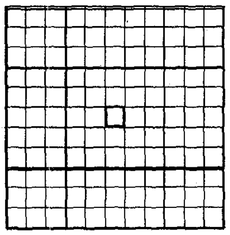

Fig. 1 A grouped sum structure. Thicker lines define the border of each sector.

For the global decision phase, data are analysed for each ROI of a given event. In average 5 ROIs are produced per event. The feature space that is being used comprises 12 features from the 4 detectors that contribute to the decision to be taken by the LVL2 system. To correlate such diverse information, a fully-connected network with 6 nodes in the hidden layer is used. The normalization of the input vector is performed by fixed factors stored in data memory, which are obtained by computing the mean value of each vector component in the training set and adding 2 σ.

Four output nodes provide particle probabilities for electrons, muons, pions and jets. These probabilities are computed for each ROI and are used in a following step that concerns physics channel identification. The achieved discrimination efficiencies to ROI data from four physics processes ($4e^-$, $2e^- + 2j$, $2e^- + 2\mu$, 4μ) were 97% (e^-), 98% (jets), 99% (muons), which were in good agreement with previous results [10]. In practice, very few pions were available in the simulated data set, so that the performance for such particle is not quoted.

Both feature extraction and global processing networks were trained by using backpropagation [11] and the hyperbolic tangent was the gain function of each neuron.

III. The Parallel Processing Environment

The TN-310 system [12] is a multiple instruction multiple data parallel computer with a distributed memory architecture. The system (see Figure 2) houses 16 nodes based on T9000 transputers, which communicate with each other by means of a network of C104 chips. Each node has access to the communication network through four high speed (100 Mbits/s) serial links (DS-links), which allow it to access data held anywhere in the system. Each node makes use of a 25 MHz digital signal processor (DSP) as a coprocessor for signal processing applications. This DSP (ADSP-21020) is, in fact, the core processor of the ADSP-21060, which facilitates the adaptation of neural processing codes already developed.

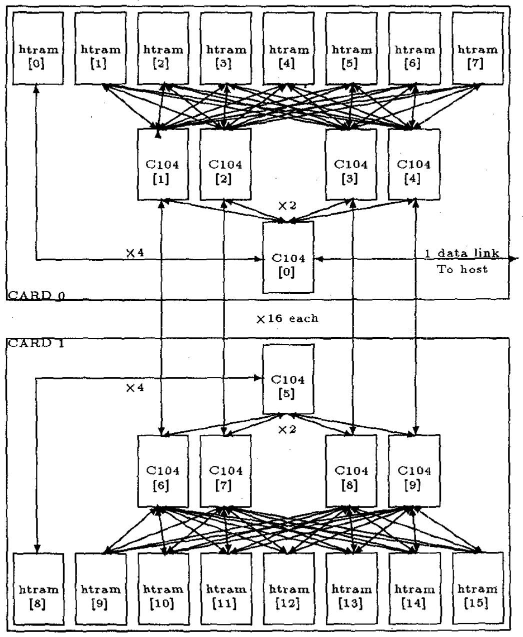

Fig. 2 The basic architecture of TN 310 system.

The T9000 is a 32-bit microprocessor that exhibits multiprocessing capabilities. Its architecture supports the creation and scheduling of any number of concurrent processes. Communication between processes on different processors takes place over virtual channels. A virtual channel processor (VCP) manages the use of physical links (DS-links) by multiplexing concurrent (virtual) channels. Thus, a message to be transmitted is split into a number of packets (according to the size of the message), and packets from different messages may be interleaved over each physical link. As a consequence, processes communicate simultaneously via each physical link.

Every packet communication is a blocking operation, which means that the transmitter VCP has to wait for an acknowledgement of the receiver unit (another T9000 in this case) until it can proceed in packet distribution. This acknowledgement is sent by the receiver unit as the header of packet arrives and this makes possible parallel usage of the four DS-Links, but introduces a possibility of saturation. This blocking feature, as a hardware characteristic, is transmitted over the software layer and applications developed for this machine must consider carefully such restriction in order to reach the ultimate speed.

Figure 2 shows that five C104 are used per board, each one connected to a corresponding DS-link of the T9000. It should be noted that the communication between tasks may be established by means of a different number of C104 switches, depending on the node allocation of the tasks. Table 1 shows such feature, as the number of packets to be transmitted is varied (in blocks of 32 bytes per packet) and the time needed for sending such packets to the addressable node is measured as the packet is routed through one up to four switches. It can be seen from the table that the time per packet is reduced when the number of packets increase (up to saturation) and each additional switch present in the communication path represents $\sim 1.7\ \mu s$ overhead. This means that communication overheads can be minimized by taking care on the selection of which task is going to run in which node. This will be further explored in next section.

The measurements displayed in Table 1 (from the TN310 system) closely agree with other measurements of the communication performance of T9000 based processing nodes through the C104 switching network, and which were performed on systems with different hardware arrangements of these devices [13, 14]. As here, the C104 delay was measured to be close to 2 μs and the bandwidth for a single virtual link was found to saturate at $\sim$ 4.5 MBytes per second, when 10 or more packets are transmitted.

The ADSP-21020 is a 32/40-bit floating point processor. The basic architecture of this device includes three independent computational units which are connected in parallel: ALU, multiplier with fixed-point accumulator and shifter. Every instruction on this processor is executed in a single cycle (40 ns), so that multiplication with accumulation and search for the next operands can be performed in one cycle.

The system can be accessed through a host machine, in our case a PC running MS-DOS and Windows. In terms of software development, the TN-310 system offers three layers of increasing flexibility but also of decreasing processing speed. In order to achieve the ultimate speed of the machine, the lowest level of software development was selected, named *C Toolset*. This layer allows for hardware configuration (in a specific language) and coding of tasks in C language with some availability for implementing typical concurrent system functions. These two levels of coding

Table 1
Communication performance (time in μs) in terms of number of packets to be transmitted and number of C104 switches involved.

# packets	1 C104	2 C104	3 C104	4 C104
1	10.2	12.1	13.8	15.5
2	16.9	20.6	24.0	27.6
3	23.5	28.9	34.2	39.5
4	30.2	37.5	44.6	51.8
5	37.2	46.0	54.7	63.6
7	50.5	62.9	75.3	87.6
8	57.2	71.4	85.4	99.5
10	70.8	88.2	106	123
11	77.3	96.7	116	136
13	90.6	114	136	159
15	104	130	157	183
16	111	139	167	195

(hardware and software) are compiled together in order to produce a single "bootable" file that runs the application.

IV. Prototype Implementation

In which follows, data to be transmitted to processors represent realistic assumptions in the amount of data that a ROI may move for each detector [13, 15]. A header is always included for identifying the event and ROI information. In average 5 ROIs per event are assumed and the ROIs are always seen in their specific form by each of the four different detectors which participate in the trigger decision.

All resources of the TN-310 system were firstly allocated to realize the global decision phase of the LVL2 prototype. Figure 3 shows the topology used, where a master node has access to the outside world and supervises the continuous distribution of data along the available processors (slaves). This node also measures the time required for such application over a significant sample of events. Neural network parameters rested in memory and the activation function (hyperbolic tangent) was stored in a look up table (with 0.01 resolution) that saturates at 7.0. Comparison with a single processor application resulted in a speed up of 7.4 for the global decision phase, representing a rate of 27 μs per event.

The prototype of the second-level trigger system was implemented by using an extension of the subfarm concept [15]. Figure 4 shows the topology assumed, in which local and global subfarms are connected. As a first assumption, the supervisor node of this implementation simulates also the distribution of ROIs by means of the local network and the preprocessing required for building ROIs with zero subtraction. This layer of processing is being developed to be introduced soon in the overall architecture being tested.

As the machine provides C104 connections of DS-links, it is assumed that the local subfarm is composed of a

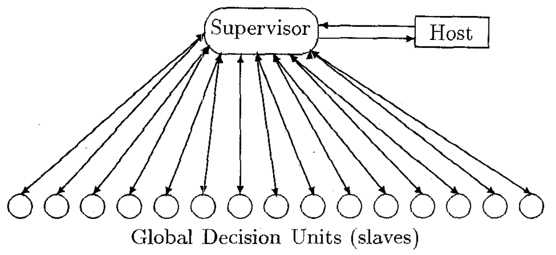

Fig. 3 The topology for implementing the global decision phase in the TN-310 system

number of slaves which can also communicate with the global subfarm. As the slaves send a small amount of data in the form of features (one packet), this assumption does not represent a significant impact on the overall performance, as communication is always performed via DS-links. On the other hand, as we process global data in the global decision phase, which needs ROI information from different detectors, we also assume that there are slaves processing ROIs from the four different detectors in the local subfarm. Therefore, the global subfarm receives the features supplied by the slaves of the local subfarm and performs global processing by using a number of slaves running neural processing. When a given ROI of a given event becomes available as seen by the four detectors, the particle probability is computed by the neural network. Further processing is then required when a full event is available (with all ROIs that belong to it).

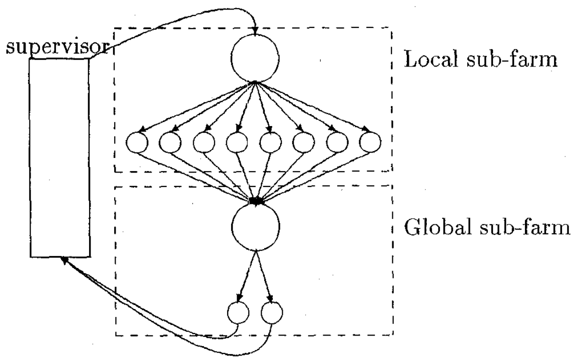

Fig. 4 Local and Global Subfarms

This extension of the subfarm concept was implemented using all resources of the TN-310 system. Figure 5 shows the configuration, in which the filled circles represent the number of switches that participate in the communication path of each node. Optimization of communication paths was applied. The feature extraction (FEX) layer handles data from calorimeters (calo), a transition radiation tracker (trt), a silicon tracker (sct) and the muon system. More processing nodes were allocated to trt and sct, as their algorithms are more time consuming. The so called local networks in fact collect global data from the FEXs located in the local subfarm. The first node (Local0) receives data from FEXs and labels features from the different detectors and for different ROIs and events. As soon as a ROI (or an event) is fully available for global processing, data are transmitted to the second node of this structure (Local1), which distributes data to the slaves (global decision units - gdus) of the global subfarm.

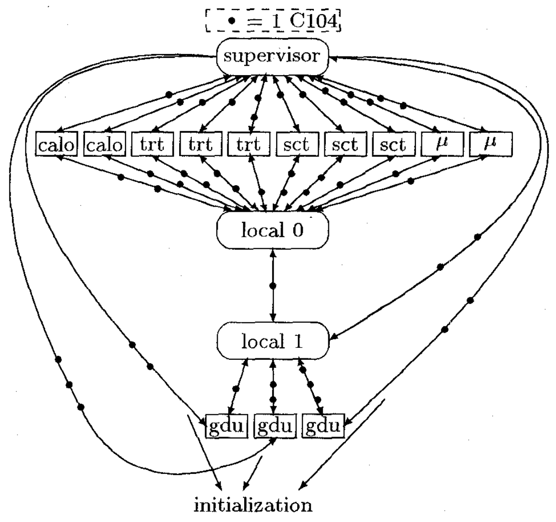

Fig. 5 Implementation diagram showing the "distance" between tasks

A speed up of 4.7 was achieved when the described approach is compared with a single processor implementation. The maximum speed per event was measured to be 380 μs (a frequency of $\sim$ 2.6 kHz for such slice). This result was obtained by using a round-robin distribution of data to the slaves of both subfarms. Evaluation tests on a circular data distribution, which was implemented by a primitive of the *C Toolset* environment, were also performed. However, this implementation raised enormously the processing time in the form of overheads to identify which processor was ready to receive data.

A final implementation was then developed, as the data displayed in Table 1 are considered in more details. Considering the time required for packet transmission through the C104 switching network, it is noted that processing time is not much smaller than communication time, when data size for each detector is considered. For example, FEXs that process calorimeter data need more than 100 μs for receiving ROI data and require only 10 μs for neural processing. Similar ratios are found for the other three detectors. Therefore, it should be investigated whether there are some idle nodes in the architecture being studied. This is to be performed by analysing the overall time required for data distribution along slaves in the local subfarm. Of course, if saturation in data distribution is achieved, it is useless to add slaves in the subfarm, as speed up can not be increased in such circumstances. The same applies to the layer of global processing.

Figure 6 shows such optimized architecture, in which the parallelism of actions is restricted to two processors for each task. This topology still sustains the same input frequency in the slice but makes use of minimal resources in both processing layers.

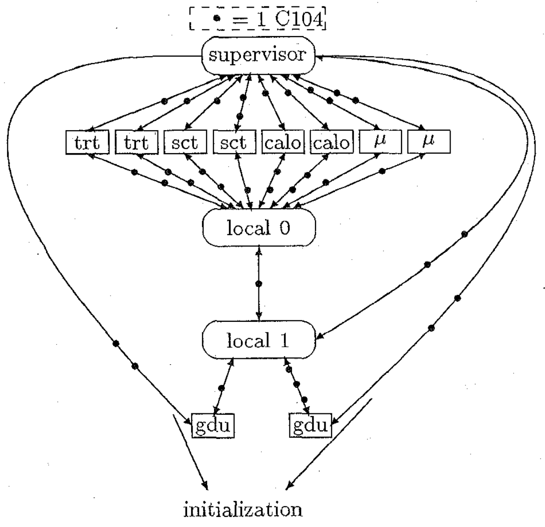

Fig. 6 Minimal configuration allowing maximum speed.

V. Conclusions

An implementation of a global decision unit and a prototype system for the second level of trigger of the ATLAS experiment were made using a distributed processing system. The mentioned system (Telmat TN310) was built using transputer based processing nodes connected with C104 switches. The global decision unit was implemented using neural networks and the concept of subfarm was applied in order to reduce ROI processing times. We were able to achieve 27 μs per ROI and this result lead us to a speed up of 7.4, when comparing such application with a single-node implementation.

A prototype of a "slice" of the second level trigger was also built. This slice, using the configuration showed at Figure 5 together with the subfarm concept, was able to achieve an event processing time of $380\mu s$ (speed up of 4.7). As a direct result of communication patterns inside the system (table 1), the number of processing nodes was reduced to 13 (Figure 6) since this configuration still could sustain the same input frequency of the first one used.

Acknowledgements

We are thankful to the support provided by CNPq, FUJB (Brazil) and the European Commission. We also thank the RD11 and ATLAS/Trigger-DAQ collaborations at CERN for providing the simulated data sets and contributing with fruitful discussions of some of the ideas developed in this work.

VI. References

[1] The Atlas Collaboration, "Technical Proposal for a General-Purpose pp Experiment at the Large Hadron Collider at CERN," CERN/LHCC/94-43, 1994.

[2] R. Cranfield et al, "Demonstration of a Four Buffer Slice of a Prototype Second-level Trigger for Atlas," Proc. of *Second Int. Workshop on Electronics for LHC Experiments*, Balatonfüred, Hungary, pp 180-183, 1996.

[3] J. M. Seixas et al, "Neural Second-Level Trigger System Based on Calorimetry," *Computer Physics Communications*, 95, pp 143-157, 1996.

[4] J. M. Seixas et al, "Implementing a Neural Second-Level Trigger System on Digital Signal Processor Technology: The Global Decision Problem," Proc. of *First Int. Workshop on Electronics for LHC Experiments*, Lisbon, Portugal, pp 324-327, 1995.

[5] C. Kiesling and S. Udluft, "Triggering on Secondary Vertices: First Study Towards a Network Algorithm for LHC Experiments," Proc. of *Second Int. Workshop on Electronics for LHC Experiments*, Balatonfüred, Hungary, pp 199-203, 1996.

[6] R.K. Bock et al. "What Can Artificial Neural Networks Do for the Global Second Level Trigger," CERN/ATLAS-DAQ/11, 1994.

[7] J.M. Seixas et al. "Neural Feature Extraction for Calorimeters Based on Optimum Weighting Procedures," *Nuclear Instruments and Methods* A389, pp 146-147, 1997.

[8] G. Klyuchnikov et al. "A Second-level Trigger Based on Calorimetry Only," CERN-ATLAS/DAQ-007, 1992.

[9] R.K. Bock et al. "Test Data for the Global Second-Level Trigger," CERN/EAST 93-01, 1993.

[10] L. Lundheim et al, "A Programmable Active Memory Implementation of a Neural Network for Second Level Triggering in ATLAS," *Int. Jour. Mod. Phys.* C6 pp 561-566, 1995.

[11] J. Hertz, A. Krogh and R. G. Palmer, "Introduction to the Theory of Neural Computation," Addison-Wesley, 1991.

[12] Telmat Multinode, "TN 310 System Manual and Training Set," France, 1995.

[13] R.W. Dobinson et al, "Triggering and Event Building Results Using the C104 Packet Routing Chip," *Nuclear Instruments and Methods* A376, pp 59-66, 1996.

[14] R. Heeley et al, "The Application of the T9000 Transputer to the CPLEAR Experiment at CERN," *Nuclear Instruments and Methods* A368, pp 666-674, 1996.

[15] B. Kastrup et al. "Study of the FEX Subfarm Concept with C40s and Digital Memory Channel Cluster," CERN/ATLAS-DAQ/68, 1997.