BMC Neuroscience

Poster presentation

Neuronal spike exchange on a Blue Green/P supercomputer: MPI_Allgather vs DCMF_Multicast

Michael L Hines*[1], Sameer Kumar[2], Henry Markram[3] and Felix Schürmann[3]

Address: [1]Department of Computer Science, Yale University, New Haven, CT, USA, [2]IBM, Yorktown Heights, NY, USA and [3]Brain Mind Institute, EPFL, Lausanne, Switzerland

Email: Michael L Hines* - Michael.hines@yale.edu

* Corresponding author

from Eighteenth Annual Computational Neuroscience Meeting: CNS*2009
Berlin, Germany. 18–23 July 2009

Published: 13 July 2009

BMC Neuroscience 2009, **10**(Suppl 1):P48 doi:10.1186/1471-2202-10-S1-P48

This abstract is available from: http://www.biomedcentral.com/1471-2202/10/S1/P48

For spiking neural network simulations on parallel machines, interprocessor spike communication can be a significant portion of the total simulation time. The simplest spike exchange method is to alternately integrate the cell equations for the minimum spike delay interval between spike initiation and spike delivery, and, at the end of the interval, send all spikes generated in the interval to all processors using the standard Message Passsing Interface MPI_Allgather operation. Given that a source cell is typically connected to thousands of target cells in large network models, the MPI_Allgather method is very effective on supercomputer clusters with thousands of processors. However, as the number of available processors becomes much larger than the cell-to-cell fanout, more complication point-to-point exchange methods should become faster than MPI_Allgather.

The Blue Gene/P supercomputer provides direct memory access (DMA) interprocessor communication across the 3-D torus network connecting adjacent quad-core nodes. The DMA controller accepts a list of destination processors and transfer takes place in the background without affecting normal CPU computation speed. The Deep Computing Messaging Framework (DCMF) provides an API to use this feature which is initiated by the function DCMF_Multicast.

With DCMF_Multicast, a spike message consists of a double precision spike initiation time and a long integer global source cell identifier. DCMF_Multicast is called when a cell fires and computation continues. In order to allow a substantial time interval between the time a multicase is initiated and the time that the spikes must be received before computation can continue on the target machines, the minimum delay interval is divided into alternating even and odd intervals. This a multicast initiated in an even interval does not have to complete until the end of the following odd interval. At the end of an interval, a spike conservation loop using MPI_Allreduce is used to guarantee that the number of spikes received at the end of the current interval is equal to the number of spikes sent in the previous interval. In this way, as long as half-delay interval computation time is longer than the multicast transfer time, transfer time should not contribute to the overall runtime and the only communication portions that increase the runtime consist of the initiation of the multicast, the handling of received spikes, and the MPI_Allreduce latency.

We compared the performance of MPI_Allgather and DCMF_Multicast spike exchange methods using 256 K artificial cell text models with each cell randomly spiking every 10–20 ms. One model had each cell randomly connected to 1 K target cells and the other model used 10 K connections per cell. To minimize MPI_Allgather communication time, the MPI_Allgather buffer size was set large enough so that overflow spikes requiring a subsequent MPI_Allgather collecting was rarely or never necessary. Furthermore, because there are fewer than 256 integration steps per minimum delay interval and fewer than 256 neurons per processor, the MPI_Allgather method can usefully compress each (spiketime, gid) pair from 12-2 bytes, reducing the constant payload to a minimum. Of course, spike compression provides no benefit with the DCMF_Multicast method since a single (spiketime, gid) pair fits into the 16 byte msginfo field of a minimum size 32 byte header packet and requires no receive callback to assemble the message data. With the 1 K connectivity text model, the turnover point where the DCMF_Multicast method has less runtime that MPI_Allgather is 16 K cores. This unexpectedly large size is primarily due to the computation time noise introduced by the small random number of DCMF_Multicast calls on a core during the half – interval. That is, load balance is slightly worse and so the synchronization time (for measurement purposes we use MPI_Barrier) becomes a large portion of spike communication overhead. Implementation modifications to the DCMF library to reduce the call time of DCMF_Multicast are ongoing.

Acknowledgements

NINDS grant NS11613l; The Blue Brain Project. The DEISA Extreme Computing Initiative for granting access to a BG/P. Philip Heidelberger and Gabor Dozsa of IBM, Yorktown gave us invaluable advice on the use of DCMF_Multicast.