Matrix Multiplication on Three Heterogeneous Processors

Brett A. Becker and Alexey Lastovetsky
School of Computer Science and Informatics
University College Dublin (UCD), Belfield, Dublin 4, Ireland
brett.becker@ucd.ie and alexey.lastovetsky@ucd.ie

Extended Abstract

We present a new algorithm specifically designed to perform matrix multiplication on three heterogeneous processors. This algorithm is an extension of the 'square-corner' algorithm designed for two-processor architectures [2]. For three processors, this algorithm partitions data in a way which on a fully-connected network minimizes the total volume of communication (TVC) between the processors resulting in lower execution times for a defined range of processor power ratios, when compared to existing partitionings. On a non-wraparound linear array where the fastest processor is the middle node, this algorithm always results in a lower TVC. Minimizing the TVC is a natural goal as matrix multiplication involves substantial communication volumes, and the links connecting the processors are possible bottlenecks. The goal of this paper is to determine if the square-corner algorithm of [2] can be successfully extended to three processors.

To our knowledge no research has been conducted to optimize matrix multiplication for the specific architecture of three connected heterogeneous processors. The most related work is [1], which used this architecture to test a more general algorithm designed for any number of processors including three. The authors also derive a lower bound which cannot be met by their partitioning for three processors. We mathematically compare our algorithm to that of [1] and show that the square-corner algorithm does approach this lower bound, and for certain power ratios can reduce the TVC.

Our motivation stems from the fact that there are many general algorithms which work well for several, dozens, or even hundreds of nodes, but all result in straight-forward partitionings when applied to a three processor architecture. These algorithms are 'straight-line' algorithms – they use straight lines to partition the matrix resulting in rectangular partitions. An example of such a partitioning with the associated communications is shown in Figure 1a. The square-corner algorithm does not use straight lines to create partitions. In [1] this algorithm was shown to minimize the TVC and therefore the total execution time whenever the power ratio between the two processors is greater than 3:1. MPI experiments comparing the square-corner and straight-line algorithms showed an average reduction of 45% in the TVC and an average reduction of 14% in the execution time with a link bandwidth of 80Mb/s, when multiplying two dense matrices on processors with ratios ranging from 1:3 to 1:25. Similar results were achieved for other link bandwidths.

In the case of three processors, the square-corner algorithm allocates square partitions "cut" from diagonally-opposite corners of the matrix to the slowest two processors. The fastest processor receives the balance of the matrix. This partitioning and the associated communications are shown in Figure 1b. On a fully-connected network this partitioning lowers the TVC only in cases where the sum of the speeds of the slower two processors is less than approximately 20% of the speed of the fastest processor. The actual point at which the TVC is lower depends on the particular ratio used. MPI experiments multiplying dense matrices show that for these ratios the execution time can be reduced by up to 10% compared to the straight-line algorithm. However for these ratios these times are slower than that of running the whole multiplication sequentially on the fastest processor.

On a non-wraparound linear array where the middle node is the fastest processor, the square-corner algorithm achieves much better results. This is because the two slower nodes do not need to communicate to complete the operation, unlike the straight-line partitioning which requires the two slower processors to communicate through the faster node, therefore increasing the TVC. A mathematical analysis shows that for this topology, the square-corner algorithm results in a lower TVC regardless of power ratio. MPI experiments show that for power ratios of 5:3:2 and a bandwidth of 190Mb/s, the communication time is 28% less and the execution time is 10% less compared to the straight-line algorithm. Similar results are achieved for other ratios and bandwidths. These execution times are faster than the sequential execution times of running the entire multiplication on the fastest processor.

Another advantage to this algorithm is that a large sub-partition of the matrix is immediately calculable – no communication is required for the product of this sub-partition to be calculated. This area is within the dashed square of processor 1's *C* matrix in Figure 1b. Architectures with dedicated communication sub-systems could exploit this property which on a fully-connected network would both broaden the range of power ratios where this algorithm is faster, and increase the margin by which it is. Future work will include exploiting this property as well as experimenting with heterogeneous link bandwidths.

We point out that this algorithm has useful applicability as the top-level partitioning between three connected clusters – a natural architecture for high performance computing. On such architectures, each cluster (containing any number of nodes internally) can be of considerable computational power, and well worth using in parallel. At a high level such collections of clusters can be viewed as a collection of individual processors, and after an initial top-level partition utilizing the square-corner algorithm, each cluster can deal with its data partition using an algorithm that will most efficiently exploit its particular architecture. The square-corner partitioning strategy may also reduce the TVC of other linear algebra kernels, and problems which have similar communication patterns.

References

[1] Olivier Beaumont, Vincent Boudet, Fabrice Rastello and Yves Robert, "Matrix-Matrix Multiplication on Heterogeneous Platforms", *IEEE Transactions on Parallel and Distributed Systems*, 2001, Vol.12, No.10, pp.1033-1051

[2] Brett Becker and Alexey Lastovetsky, "Matrix Multiplication on Two Interconnected Processors", Accepted to *HeteroPar'06,* to appear in *Proceedings of Cluster2006* (*IEEE International Conference on Cluster Computing*), *Barcelona, September 2006.*

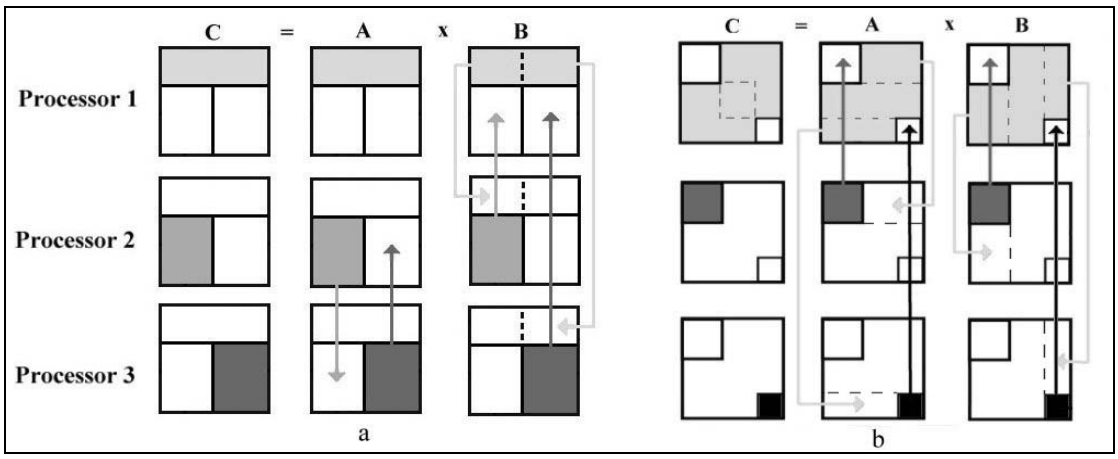

Figure 1. ***a*** **shows the straight-line partitioning and associated communications.** ***b*** **shows the square-corner partitioning and associated communications.**