

“MxN” Parallel Data Redistribution Research in the Common Component Architecture (CCA)

D. Bernholdt (ORNL), F. Bertrand (Indiana), R. Bramley (Indiana), K. Damevski (Utah), J. Kohl (ORNL), S. Parker (Utah), A. Sussman (Maryland)

Summary

The “MxN problem” relates to sharing and exchanging decomposed data fields between two parallel software components, each running on a different number of processes or with a disparate topology. Cooperating parallel programs need this new fundamental capability for model coupling and parallel computation/visualization pipelines, to negotiate the efficient exchange of parallel data structures. Further, such parallel programs must functionally communicate with each other using Parallel Remote Method Invocation (PRMI).

A major research focus for the Common Component Architecture Forum is defining semantics and efficient coupling schemes between large-scale parallel scientific software components. Part of this effort is captured in solving the “**MxN Problem**,” where parallel components running on differing numbers of processors, possibly remote, must share and exchange elements of decomposed data arrays. The MxN problem is increasingly important, as scientists begin to couple together large single-domain codes to create higher-fidelity integrated multidisciplinary simulations. The number of processes used by each parallel code can often be uniquely restricted, either by algorithm (e.g., to powers of two), by resource usage (e.g. batch scheduling systems), or simply by the relative amount of work each component needs to perform on its given portion of an array. Therefore, efficient mappings and “communication schedules” must be generated to relate the elements of one parallel array to those of another, to effectively and efficiently share the data. This concept is illustrated in Figure 1.

Within most modern distributed memory programs, the most difficult aspect of MxN is the bookkeeping associated with identifying process ownership of data, and arranging the communication schedule (the timing and sequencing of messages) to keep the data correctly updated. Most automated data distribution systems and libraries handle this bookkeeping for the user, yet there must still

be some generalized means for describing the semantics of each given data distribution. In component-based frameworks, such data description services must be provided to enable redistribution of data between one component with M processes and one with N processes, with possibly widely varying data layout for each parallel component. The CCA Forum is addressing these challenges in two ways: by developing a uniform interface to describe the data distributions held by parallel components, and by creating components and framework services to handle the synchronization and transfer data elements for MxN redistribution.

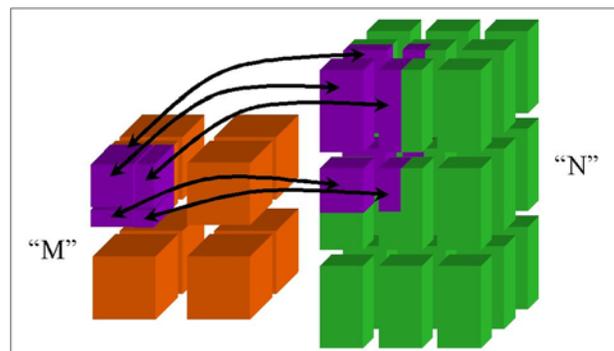


Figure 1: “MxN” Data Mapping Problem

Because the majority of data in scientific simulations appears as scalar values or arrays of primitive types, CCA is developing a **distributed array descriptor** interface that succinctly describes the ways data arrays can be distributed. Most standard decomposition types are supported, including: block, cyclical, generalized block, trees and explicitly indexed patches.

Currently, MxN services are being provided for two classes of CCA frameworks. ORNL has developed a stand-alone MxN component for parallel direct-connect frameworks, i.e. where all of the interacting components are part of a single parallel program. This MxN component has an instance associated with each parallel process, providing an attachment point for all of the parallel components in the framework (Figure 2). This allows users to explicitly specify MxN transfers and data redistribution policies via a powerful API. The cohort of MxN component instances uses an “out of band” communication system, like PVM or MPI, to efficiently invoke the actual data transfers. Eventually, the encapsulating frameworks will be able to automatically negotiate an optimized MxN communication schedule, and will invoke redistribution as a service when needed between coupled parallel components.

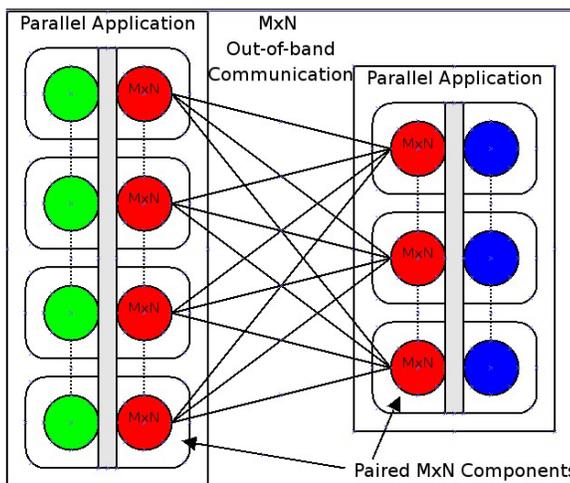


Figure 2: MxN Components in a Parallel Direct-Connect Framework

Unfortunately, in distributed frameworks an MxN component cannot be collocated with application components, because each may exist on a different machine or be started with a separate run-time system. In this case, **Parallel Remote Method Invocation** (PRMI) must be applied to communicate among the distributed components. The University of Utah has extended SCIRun2 to define two types of PRMI: *independent* calls, where one process on each side is matched up to form a send/receive

RMI pair; or *collective*, where all of the processes on each side participate. Collective SCIRun2 calls can handle disparities in the number of processes on each side by creating ghost arguments or return values, depending on which side has “extra” processes, respectively. Indiana University has created the Distributed CCA Architecture (DCA) framework, which uses MPI communicator groups to determine such distributed process participation. A user describes data layouts using MPI data types, displacements and counts, and the framework automatically provides the necessary remote parallel data redistribution operations.

Although each of these approaches are fundamentally different, the CCA can hide the underlying distinctions between parallel direct-connect and distributed frameworks by creating a common MxN interface. This interface allows users to portably define the transfer of data elements between disparate distributions, regardless of the specific framework being used.

While quite challenging, this MxN problem only scratches the surface of the more general problem of coupling parallel codes to create multidisciplinary simulations. A variety of spatial and temporal interpolation technologies, with flux conservation and even unit conversions, are required to couple real production codes, e.g. for domains like climate modeling, process simulation or fusion energy simulations. The CCA seeks to address these capabilities in its future research agenda.

For further information on this subject contact:

James A. Kohl
Oak Ridge National Laboratory
E-mail: kohlja@ornl.gov
<http://www.csm.ornl.gov/cca/mxn/>

Randall Bramley
Indiana University
E-mail: bramley@indiana.edu
<http://www.cs.indiana.edu/~febertra/mxn/>

Steven G. Parker
University of Utah
E-mail: sparker@cs.utah.edu
<http://www.sci.utah.edu/>