

Petascale Simulation Initiative



John M. May
(925) 423-8102
may10@llnl.gov

Future supercomputers will gain performance mainly from increased parallelism, rather than faster processors. To use these machines effectively, application developers must find new ways to exploit massive parallelism. At present, most parallel applications use a single program, multiple data (SPMD) model, in which each processor simultaneously applies the same algorithm to a different portion of the workload. This approach is conceptually simple, but balancing the workload to maximize efficiency can be tricky, especially if the workload distribution changes as the computation proceeds. Moreover, load balancing becomes harder with increasing parallelism.

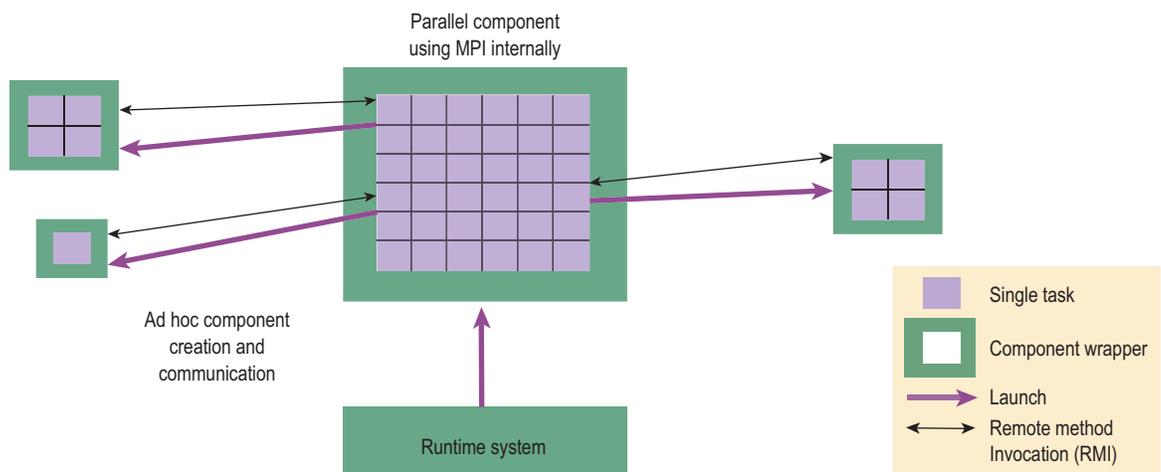
The Petascale Simulation Initiative has developed a new programming model called Cooperative Parallelism that lets applications manage and distribute work more flexibly. It is a multiple program, multiple data (MPMD) model, so different parts of a parallel application can work concurrently on different tasks running different executables. Cooperative Parallelism is designed to complement the SPMD model, so existing parallel applications can be

combined or augmented to form larger federations.

A Cooperative Parallel federation (Fig. 1) begins with the execution of a single software component, which is a sequential or SPMD parallel program. Any process within this component may launch additional components running different executables, and those may launch others, and so on. Components communicate through remote method invocation (RMI), in which a thread of control within one component may invoke a method (*i.e.*, call a function) in another component. Cooperative Parallelism is a loosely-coupled model of computing that permits components to be designed independently of each other while making the interfaces through which they interact more explicit.

We have also investigated a technique called Adaptive Sampling that can greatly improve performance in a broad class of multiscale and multi-physics simulations. This technique supports simulations in which coarse-scale computations are augmented with data from costly fine-scale computations. The application can assign the two kinds of work to separate pools of processors

Figure 1. Cooperative Parallel application, consisting of multiple components that can be launched and terminated as needed. Each component is an executing sequential or parallel program. Components communicate through RMI.



(Fig. 2). That step by itself can improve load balance, since a pool of fine scale “servers” can provide data to any processor in the coarse-scale computation that needs it. However, even more substantial performance gains arise when the simulation caches fine-scale data and uses it to satisfy future requests from the coarse-scale model instead of recalculating it.

Project Goals

We aim to create a programming model that large scientific applications

can adopt incrementally to create more sophisticated simulations and better utilize massively parallel computers. The project also seeks to demonstrate that Cooperative Parallelism can benefit a range of applications of interest to LLNL by allowing them to adopt a federated model to complement standard SPMD parallelism.

Relevance to LLNL Mission

Cooperative Parallelism should benefit large-scale codes as developers seek efficient ways to use massive parallelism.

Its task-parallel model lets applications exploit multiple dimensions of parallelism at once. It also offers a natural route to designing simulations that combine multiple interacting models. Moreover, because cooperative parallelism can coexist with current SPMD parallelism, applications can adopt it incrementally. Cooperative Parallelism will also benefit users of smaller-scale simulations that are executed repeatedly for parameter studies or optimization studies. Our model greatly simplifies the process of orchestrating multiple instances of a simulation running concurrently.

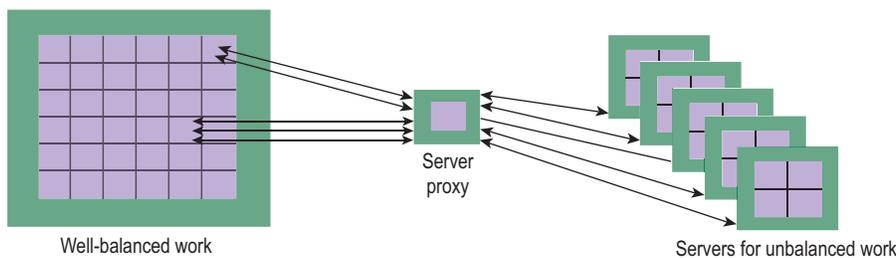


Figure 2. Adaptive sampling. Applications can be factored into well-balanced and imbalanced portions, with each portion assigned to a different pool of nodes. When processes in the well-balanced portion need additional data, they can send a request using RMI through a server proxy, which assigns the work to an available server component.

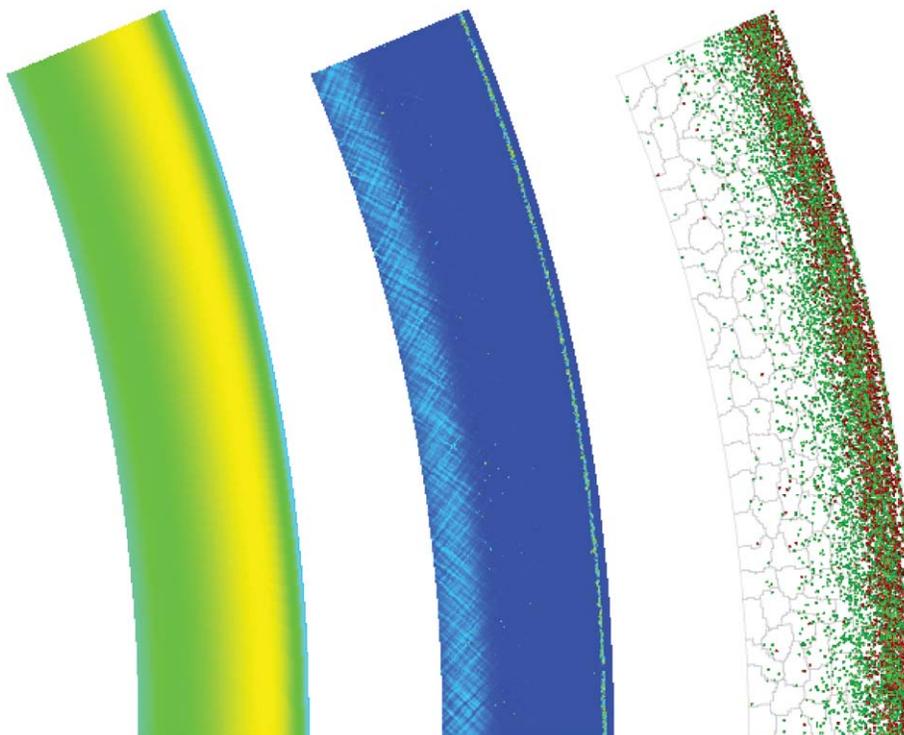


Figure 3. A multiscale material model of a part of an expanding cylinder, showing pressure on the left, plastic strain rate in the middle, and the number of fine-scale evaluations required on the right. Green indicates one evaluation; red indicates two or more evaluations. The locations requiring these extra calculations track loading changes induced by passage of waves through the material. Data are shown for a time increment over which a relatively large number of evaluations are performed. Parallel domain decomposition is also indicated by light gray boundaries in the plot on the left.

FY2007 Accomplishments and Results

The Petascale Simulation Initiative concluded its three-year term in April 2007. We designed the Cooperative Parallel model, implemented an application-independent approach to adaptive sampling, developed a working prototype of the Cooperative Parallelism runtime system and demonstrated a multiscale materials modeling simulation (Fig. 3) using adaptive sampling on more than 1,000 processors. The project also demonstrated its utility as a means for simplifying parameter studies through collaboration with LLNL’s PSUADE project.

Related References

1. Ashby, S., and J. May, “Multiphysics Simulations and Petascale Computing,” *Petascale Computing: Algorithms and Applications*, D. Bader, Ed., Chapman & Hall/CRC Press, 2007.
2. Barton, N., J. Knap, A. Arsenlis, R. Becker, R. Hornung, and D. Jefferson, “Embedded Polycrystal Plasticity and *In Situ* Adaptive Tabulation,” *Int. J. Plasticity*, in press, 2007.
3. Kumpf, G., J. Leek, and T. Epperly, “Babel Remote Method Invocation,” *IEEE Int. Parallel and Distributed Processing Symp.*, March 2007.

FY2008 Proposed Work

Further work, including closer collaborations with major LLNL simulation codes, is anticipated in FY2008.