

# "Lite" Done Right

New parallel library succeeds by going easy on the extras *by Saren Johnston*

**B**etter performance for less hassle – that’s the advantage of a new message-passing library developed at Ames Laboratory. The library, called MP\_Lite, makes it possible to extract optimum performance from both workstation and personal computer clusters, as well as from large massively parallel computers. It supports and enhances the basic capabilities that most software programs require to communicate between computers.

## Fewer and faster

MP\_Lite might be thought of as a “slimmed-down” version of the message-passing interface, or MPI, standard, a widely used model that standardizes the syntax and functionality for message-passing programs, allowing a uniform interface from the application to the underlying communication network. The MPI standard eases the parallel programming task by providing a common syntax for communicating between computers, making codes portable between widely varying supercomputers.

The full MPI standard contains many options that most people don’t use very often. MP\_Lite offers only the core functions of MPI, which are enough for most codes. The emphasis is put on implementing these functions in the most efficient manner, providing all the performance without all the extra options. “After all, a Corvette will go fast and look good even without a CD player and electric seats,” says Dave Turner, an Ames Laboratory computational scientist and the principal investigator for the MP\_Lite project.

## Perfecting performance

MP\_Lite could be scaled up easily, but its objective is not to provide all the capabilities of the full MPI standard. “Our goal with MP\_Lite is to illustrate how to get better performance in a portable and user-friendly manner and to understand exactly where any inefficiencies in the MPI standard may be coming from,” says Turner. He explains that the MP\_Lite library is smaller and much easier to work with than full MPI libraries. “It’s ideal for performing

message-passing research that may eventually be used to improve full MPI implementations and possibly influence the MPI standard,” he says.

Turner notes that it was “mainly frustration” that led him to develop the MP\_Lite library. “Most message-passing packages are large and clunky to work with, and can be difficult to install and optimize. If you run into any errors at all, they give you very cryptic messages that mean nothing unless you actually wrote the library,” he says. “So a lot of the reason I got into this project was not just to improve the efficiency, but also to improve performance – make the message-passing more user-friendly.”

Offering an example, Turner says, “If two processors are communicating, and one waits a minute for a response from the other one – well, a minute is a very long time in this context – the library should put out a warning into a log file. But that’s something that’s not done. Most message-passing systems don’t tell you what’s wrong if a communication buffer overflows or a node is waiting for a message that never gets sent. What if there’s a five-

Dave Turner shows how the MP\_Lite library is organized in this simplified sketch. Running beneath a full MPI implementation, such as Argonne's MPICH, MP\_Lite can pass on its efficiency and performance to the larger library.

minute wait for a message?" he continues. "Something is probably frozen up, so at that point the library should implement an abort and give the user as much information about the current state of the system as possible."

Turner notes that MP\_Lite operates with minimal buffering and warns if there are any potential problems. When possible, MP\_Lite will dump warnings to a log file and eventually time out when a lockup occurs. "There's a lot of these user-friendly aspects that I'd like to see put into other message-passing systems," he says. "That's one area that doesn't get focused on very often."

"Many of the people developing parallel library codes don't necessarily use them, and it's not in the MPI standard that you have to make them user-friendly," Turner continues. "They're getting paid to add

functionality. For them performance and usability have been almost a second consideration, which is unfortunate."

### Putting on the "squeeze"

Turner says MP\_Lite was "born out of a need to squeeze every bit of performance from the interprocessor communication between nodes in personal computer or workstation clusters, where the Fast Ethernet and even Gigabit Ethernet speeds are much slower than communication rates in traditional massively parallel processing systems." This is due to the fact that although massively parallel systems have processors very similar to those of PCs or workstations, they also have more customized communications systems that allow faster communication rates.

"In PC and workstation clusters, we're catching up to where we're getting at least closer to those rates, but we still need to squeeze everything out that we can," says Turner. "So one of the things that I do is look for where we can squeeze it out. Is it at the message-passing level; in the operating system, itself; or in the driver implementations – there's a lot of

evaluation in the whole message-passing, or communication, hierarchy that's involved in the MP\_Lite project."

In addition to enhancing performance, another goal Turner has for MP\_Lite is to tie it directly to a full MPI library. To do so, he's been working with the Department of Energy's Argonne National Laboratory and running their MPICH library on top of MP\_Lite. "By doing this, we can pass the good performance of MP\_Lite on to the full MPI implementation," he says. "So we combine the best of both, keeping the efficiency of my library and the greater functionality of Argonne's."

Turner says he named the library MP\_Lite for several reasons. The small size of the library's code makes it easy to install anywhere – it compiles in under a minute. There's much less code, so it's more streamlined than MPI. It can also be easily modified to include more of the full MPI functions. And MP\_Lite has its own syntax, which is simpler and can be used in place of the MPI syntax. Of course, there's always the fun of the clever response Turner is able to make to people who say to him, "I use this MPI function; why isn't it in your library?" He simply replies, "Well, it's 'lite.'"

MP\_Lite is on the World Wide Web at [http://www.scl.ameslab.gov/Projects/MP\\_Lite/](http://www.scl.ameslab.gov/Projects/MP_Lite/) and can be downloaded free of charge. "I tell people straight out that it's an experimental product under development," says Turner. "It's been a good way of finding out about bugs in the code."

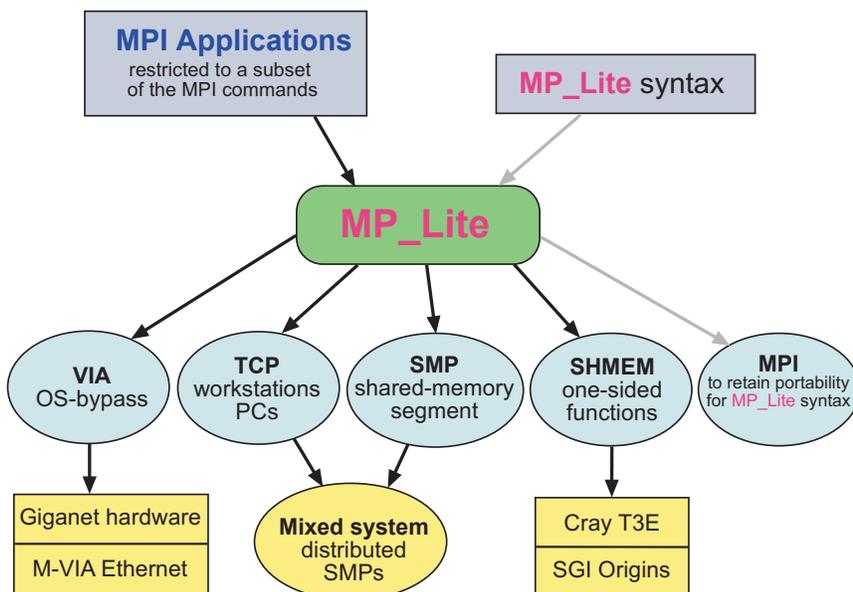
Turner admits that the work on MP\_Lite suits him well. "I like the puzzle aspect of it," he says. "I like tuning codes and getting them to run on a scalable computer, and trying to squeeze more performance out of what's there." ♦

#### For more information:

David Turner, (515) 294-1307  
deturner@iastate.edu

#### Research funded by:

DOE Office of Advanced Scientific Computing Research,  
Mathematical, Information and Computational Sciences Division



*MP\_Lite delivers nearly the full range of performance as the underlying communication layer, and does so in a portable and user-friendly manner. It can be run on top of transmission control protocol, or TCP, on clusters of workstations; the high-performance, native SHMEM library on Cray T3E and SGI Origin systems; and on any system where MPI is installed to retain complete portability. Performance is achieved by keeping everything simple and clean.*