Topic 01
Support Tools and Environments

Michael Gerndt

Global Chair

Parallel computing is a key technology for many areas in science and industry. Outstanding examples are the ASCI and Blue Gene programs that target only very few but critical applications. A much broader spectrum of applications can be found on any of the machines of supercomputing centers all over the world.

Although programming parallel computers is a difficult and time consuming task, a lot of highly skilled programmers are engaged in this to make most efficient use of the expensive resources. Tools are crucial to facilitate their task. Not only classical tools, such as performance analysis and debugging tools, are required but also very specialized tools that are useful only on parallel computers.

Critical to the acceptance of those tools by the programmers is their integration into a programming environment and their availability across multiple platforms, thus reducing the learning effort. In addition, tools have to be integrated with the programming language and ideally with the high-level model of the application that the programmers has in mind before actually writing Fortran or C/C++ code.

The articles selected for presentation within this topic demonstrate the wide range of tools necessary to ease programming of parallel computers.

Four papers have been selected in the area of parallel programming environments. The tool described in *Dynamic Performance Tuning Environment* by Anna Morajko, Eduardo César, Tomàs Margalef, Joan Sorribes, and Emilio Luque supports automatic online performance analysis and tuning based on the knowledge of special design patterns.

A technique that underlies a lot of tools for parallel programming is the efficient recording of partial orderings of tasks during parallel execution. Paul Ward and David Taylor present a new technique in the paper on *Self-Organizing Hierarchical Cluster Timestamps*.

The paper on *A Tool for Binding Threads to Processors* by Magnus Broberg, Lars Lundberg, and Håkan Grahn; and the paper on *VizzScheduler - A Framework for the Visualization of Scheduling Algorithms* by Welf Löwe and Alex Liebrich present tools to compute and understand scheduling decisions.

The article by Rajeev Muralidhar and Manish Parashar on *A Distributed Object Infrastructure for Interaction and Steering* describes an application-level tool facilitating the development of interactive parallel applications.

Two papers have been selected for this topic that describe tools on system level. Checkpointing parallel applications is already a difficult and I/O intensive task on parallel machines. The paper on *Checkpointing Facility on a Metasystem* by Yudith Cardinale and Emilio Hernández presents a tool for a much more dynamic environment, a metacomputing system.

R. Sakellariou et al. (Eds.): Euro-Par 2001, LNCS 2150, pp. 34–35, 2001.

Crucial to any parallel program is an efficient implementation of communication abstractions. The paper on *Optimising the MPI Library for the T3E* by Stephen Booth gives a detailed overview of optimization techniques for MPI implementations based on remote memory access on the Cray T3E.

The broad spectrum of tools for parallel computers presented in this topic makes very clear that this area of research still gives a lot of opportunities for bright ideas and many talented students.