Can Parallel Programming Be Made Easy for Scientists?

Peter Kacsuk

Distinguished/Leading Professor MTA SZTAKI Research Institute
H-1132 Budapest Victor Hugo 18-22. Hungary
Phone: +36-(1)-329-7864 FAX: +36-(1)-329-7864
kacsuk@sztaki.hu

Abstract

The general opinion is that parallel programming is much harder than sequential programming. It is true if the progammer would like to reach over 90

Our P-GRADE environment was designed to meet these natural requirements of scientists. It is a completely graphical environment that supports the whole life-cycle of parallel program development. The programming language, called GRAPNEL, is a graphical extension of C, C++ or FORTRAN where graphics is used to express activities related to parallelism (like process creation, communication, etc.) and at the same time graphics hides the low level details of message passing library calls like PVM and MPI calls. Program constructs independent of parallelism can be inherited from sequential C, C++ or FORTRAN code. Moreover complete sequential C, C++ or FORTRAN libraries can be used in the GRAPNEL program and in this way parallelizing sequential code becomes extremely easy. Usage of predefined process topology templates enables the user to quickly generate very large parallel programs, too.

A user-friendly dragg-and-drop style graphical editor (GRED) helps the programmer to generate any necessary graphical constructs of GRPNEL. The DIWIDE distributed debugger provides systematic and automatic discovery of deadlock situations that are the most common problems of message passing parallel programs. DIWIDE also supports replay technique and hence the cyclic debugging techniques like breakpoint, step-by-step execution can be applied even in a non-deterministic parallel programming system. Performance analysis is supported by the GRM monitor and the PROVE execution visualization tool. The instrumentation is completely automatic, filters can be easily added or removed for the GRM monitor. The execution visualization can be done both off-line and on-line providing various synchronized trace-event views as well as statistics windows on processor utilization and communications. The connection between the source code and the trace-events can be easily identified by the source code click-back and click-forward facilities. GRM and PROVE are able to support the observation of real-size, long-running parallel programs, too. In many cases performance bottlenecks are due to wrong mapping of processes to processors. An easy-to-use mapping tool supports the user to quickly rearrange the processes on the processors of the parallel system.

The talk will highlight those features of P-GRADE that makes parallel programming really easy for non-hacker programmers, including scientists

V.N. Alexandrov et al. (Eds.): ICCS 2001, LNCS 2073, p. 7, 2001.