Performance Analysis for SVM-Fortran with OPAL

Gerndt, M. (Corresponding author) ; Krumme, A. ; Özmen, S.

1995

CSREA [Athens, Ga.]
ISBN: 0-9648666-0-9

International Conference on Parallel and Distributed Processing Techniques and Applications, PDPTA '95, Athens, Georgia, USA, 3 Nov 1995 - 4 Nov 1995

Please use a persistent id in citations: http://hdl.handle.net/2128/11587

Abstract: Programming distributed memory parallel computers with message passing is often considered to be a difficult task. To overcome the drawbacks of this programming style, several efforts have been made in the field of new parallel programming languages. One example is the shared virtual memory programming model. SVM-Fortran is an extension of Fortran77 for programming shared virtual memory systems. It provides special notations for work distribution to optimize data locality and load balance. There exists a number of different tools for performance debugging in message passing systems, but none of these would fulfill the special requirements of the SVM programming model. Moreover, we observed that the user requirements for performance debugging strongly recommend tools which are adapted to the programmers view of the program. Therefore, special emphasis is set on source code related methods of OPAL and the underlying trace generation with the performance monitor SAM.

Contributing Institute(s):

1. Zentralinstitut für Angewandte Mathematik (ZAM)
2. Jülich Supercomputing Center (JSC)

Research Program(s):

1. 899 - ohne Topic (POF3-899) (POF3-899)
Coarray Fortran is today supported by Cray with extended features and by Intel with compatibility with Fortran standards [4]. Open-source compilers are also in different development stages, such as the GCC compiler (OpenCoarrays [5]) and OpenUH [6]. Over the years many benchmark studies have been performed [7–12] to investigate the performance of Coarray Fortran in comparison to MPI. This behavior is dependent upon support for remote direct access, by the underlying hardware architecture. The overall performance comparison of MPI and Coarray Fortran is murky with contradicting results, such as, for example, by [7, 8]. The contradiction in the better performance could be attributed to the different communication requirements of various scientific codes.