Performance Evaluation of Large-scale Parallel Simulation Codes and Designing New Language Features on the HPF (High Performance Fortran) Data-Parallel Programming Environment

Group Representative
Yasuo Okabe Academic Center for Computing and Media Studies, Kyoto University, Professor

Author
Hitoshi Murai The Earth Simulator Center, Engineer

High Performance Fortran (HPF) is considered to be superior to Message Passing Interface (MPI), which is now frequently adopted for the parallelization on distributed-memory parallel computers, in the ease of writing parallel programs. This project aims to evaluate the effectiveness of HPF, clarify its advantage and disadvantage and detect improvements for it. We parallelized a plasma simulation code IMPACT-3D with HPF/ES and obtained performance of 14.9 Tflops in the 512-nodes execution on the ES. This result shows HPF/ES has a high scalability and can be used readily in developing an actual simulation program. We also evaluated many benchmark programs parallelized with HPF/ES.

Results: We parallelized a plasma simulation code IMPACT-3D with HPF/ES and obtained performance of 14.9 Tflops in the 512-nodes execution on the ES.

keywords: High Performance Fortran, HPF, benchmark evaluation, plasma simulation

1. Project Overview

1.1. Motivation

In general, Message Passing Interface (MPI) is frequently adopted for the parallelization on distributed-memory parallel computers. But the programming with MPI is considered to be very difficult and tedious for principal users of the Earth Simulator because they have to specify all of the parallelization steps by themselves. Accordingly, it is strongly invoked to provide an alternative, and we believe that High performance Fortran (HPF) is it. All that users need to do with HPF is to specify the data layout with some simple directives, and the remaining tasks of parallelization are handled automatically by the HPF compiler.

1.2. Goals

We will parallelize large-scale real applications from various fields, such as atmosphere, ocean, plasma, FEM and aerodynamics, with High Performance Fortran (HPF) to evaluate them on the Earth Simulator and investigate the results in detail. We will also study the programming methods of hierarchical parallelization with HPF, because it is important to take advantage of all of the inter-node parallelization, intra-node parallelization and vector processing in one arithmetic processor to fully exploit the performance of the Earth Simulator. The improvements and required new features of the HPF compilers will be detected and proposed.

2. Benchmark Evaluation

We evaluated many benchmark programs from the NAS Parallel Benchmarks [1], the HPFBench benchmark suite [2], etc. This work aims to evaluate parallelization capability of HPF/ES and detect both of its advantages and disadvantages toward further development.

2.1. Evaluation Results

From the NAS Parallel Benchmarks we evaluated the programs of:

- Embarrasingly Parallel (EP)
- Fast Fourier Transform (FT)
- Scalar Pentadiagonal (SP)
- etc.

and from the HPFBench benchmark suite those of:

- Diffusion equation in three dimensions using an explicit finite difference algorithm (diff-3d)
- Seismic processing: generalized moveout (gmo)
- QCD: staggered fermion Conjugate Gradient method (qcd)
- Generic direct N-body solvers with long range forces (n-body)
- etc.

These results are examined in detail and compared with the results by the other HPF compilers or on the other platforms to analyze the characteristics of the HPF language as well as HPF/ES [3]. In addition, we started to evaluate the NAS Parallel Benchmarks 3.0, which includes the HPF version.

2.2. Improvements derived from the evaluation

The evaluation results revealed requirements of improvement of HPF/ES. We reported them to its developers and some have been implemented to the latest version of HPF/ES. Some of the improvements are shown below.

- schedule reuse for the REFLECT communications
  HPF/ES can make some communications share once-generated schedule to reduce the cost of generating schedules, if possible. A REFLECT directive has not been the target of this optimization. The optimization facility is now extended, and a schedule once generated for a REFLECT can be reused by the other REFLECTs for arrays of the same distribution.
- modification of the SHIFT communications
  The SHIFT communication is the primal communication primitive of HPF/ES and generated for the nearest-neighborhood access of a distributed array.
  Although that on the last dimension of an array has been implemented as effectively as possible, there is a room of optimization for that on the another dimension, because the region of the array to be transferred is not continuous and therefore each of the segment is transferred one by one in a straightforward way. This modification internally fuses such segments into one temporal buffer and transfers it to achieve higher efficiency.
- message aggregation
  This is an optimization technique for successive communications which handle their respective arrays but share the same schedule. Buffers of such successive communications are packed into one and sent/received by only one invocation of the runtime, which can reduce the overhead significantly especially for such expensive communications as transposition and gather/scatter.
  The feature is now under implementation and to be provided in the future release.
- etc.

3. Three-dimensional Fluid Code

3.1. Code Characteristic

In inertial confinement fusion with ablatively accelerated targets, the Rayleigh-Taylor instability, which is one of phenomena in fluid dynamics, can be induced and destroy spherical symmetry of the imploding target. Perturbations at the interface exponentially grow in linear stage of the Rayleigh-Taylor instability. After saturation of the linear growth, the instability shifts to the nonlinear free-falling phase and forms bubble-spike structures. It is one of major subjects in this research field to investigate this instability since target deformation, which is associated with the Rayleigh-Taylor instability, reduces total nuclear reaction yield to significantly lower than predicted value. Three-dimensional fluid code, IMPACT-3D (IMPlosion Analysis Code with TVD scheme) is fully Eulerian and this is necessitated from the requirement to accurately model rotational flows that are generated by the Rayleigh-Taylor. A Cartesian coordinate system is employed in IMPACT-3D to model convergent asymmetric flows precisely. Other coordinate systems contain a singularity at the origin that would cause inaccuracies. The TVD scheme can capture discontinuities such as shock wave fronts and contact surfaces within a few meshes even after many time steps, and it has a second order accuracy both in space and time without introducing non-physical oscillations at the discontinuity.

3.2. Automatic Communication

IMPACT-3D performs three-dimensional compressible and inviscid Eulerian fluid computation with the explicit 5-point stencil scheme for spatial differentiation and the fractional time step for time integration. Therefore, it is easy to parallelize this code with an ordinary domain decomposition method [4]. The first dimension of three-dimensional Fortran array is used for vectorization and the third dimension for HPF parallelization. Thus we distributed all three-dimensional arrays with (*,*,BLOCK), and shadow regions are allocated for distributed dimension. All parallelizable loops except one in the code were automatically parallelized without inserting INDEPENDENT directives. The only one exception was a loop that includes reduction operations. For this loop, an INDEPENDENT directive followed by a REDUCTION clause was inserted. Required communication is shift- and reduction-type. HPF/ES succeeded in generating efficient communications for them and eliminating all the redundant data transfers [5]. It performed message coalescing optimization for all the possible opportunities. In other words, more than one shift-type communications of the same pattern for a loop nest are packed into one message to make the message length larger. Communications generated by HPF/ES are composed of two phases, communication schedule construction and message transfer. The schedule is a set of information required for transferring messages in the second phase, such as buffers to be transferred, communication pattern, and pairs of the sender and receiver. The structure of the schedule is designed to be independent of a target array. The schedule constructed in a communication can, therefore, be shared with others if the arrays to be transferred have the same shape and distribution, and are accessed in the same index pattern. HPF/ES performs the optimization of communication schedule reuse. It generates just one communication schedule for the off-processor array accesses of the same pattern inside of a subroutine boundary. Many techniques to optimize shift-type communications was developed [6,7], but our optimization method is superior to them in the point that schedule construction and message transfer can be optimized separately. We made the performance evaluation using 2048x2048x4096 mesh, and achieved 12.5 TFLOPS, 39% of the peak performance on 512 nodes (4096 processors). We are very encouraged to get this outstanding performance on a real-world scientific application parallelized with HPF. For this achievement, 45 lines of HPF directives are inserted to 1334 lines (without comment lines) of the original Fortran program.

3.3. Manual Optimization with HPF/JA

We made a further performance tuning by using the REFLECT and LOCAL directives defined in the HPF/JA language specification. As HPF/ES could perform advanced communication optimizations, there were no differences in the generated communications themselves whether or not the directives are inserted. However, we were able to reduce the number of communication schedule generations by using the directives. When the compiler makes the automatic computation of communication schedule, it has to be generated at least once for a subroutine invocation, because the communication pattern is computed using the runtime information of array access pattern. On the other hand, when the directives are manually specified, the communication pattern can be determined statically, because the array elements to be transferred can be computed just by the shape, mapping, and shadow of the array, which are specified in declarative statements. Thus, the manual optimization can reduce frequency of the schedule construction to only once for each array throughout the whole execution. By inserting 12 lines of the HPF/JA directive, we can finally get 14.9 TFLOPS of the total sustained performance, which is corresponding to 45% of the peak performance. We would like to emphasize that this excellent achievement has been done with easy HPF directives, not with complicated MPI programming.

References

1. D. E. Bailey, et al., The NAS Parallel Benchmarks, Technical Report RNR-94-007, NASA Ames Research Center (1994).
2. Y. Charlie Hu, Guohua Jin, S. Lennart Johnsson, Dimitris Kehagias, and Nadia Shalaby, HPFBench: a high performance Fortran benchmark suite, ACM Transactions on Mathematical Software, vol. 26, no. 1, pp. 99-149 (2000).
3. K. Asaoka, H. Murai, A. Hirano, Y. Okabe, and M. Kanazawa: Evaluation of HPF Compilers on Multiple Platforms, IPSJ SIG Notes, 2003-HPC-93(HOKKE-2003), pp. 49-54 (2003).
4. Sakagami, H. and Ogawa, Y.: Performance Evaluation for 3D Fluid Code and 2D Particle Code Using HPF, Proc. of 3rd HPF User Group meeting, Redondo Beach, USA, August 1-2 (1999).
5. Sakagami, H. and Mizuno, T.: Compatibility Comparison and Performance Evaluation for Japanese HPF Compilers using Scientific Applications, Concurrency and Computation: Practice and Experience, Vol. 14, pp. 679-689 (2002).
6. Chakrabarti, S., Gupta, M., and Choi, J.-D.: Global communication analysis and optimization, Proc. of ACM SIGPLAN Conference on Programming Language Design and Implementation, Philadelphia, PA, USA, May 21-24 (1996).
7. Roth, G., Mellor-Crummey, J., Kennedy, K. and Brickner, R. G.: Compiling Stencils in High Performance Fortran, Proc. of SC'97: High Performance Networking and Computing, San Jose, CA, USA, November 15-21 (1997).

並列処理言語HPF(High Performance Fortran)を用いた大規模並列実行の性能検証および新規機能の検討

利用責任者
岡部　寿男　　京都大学 学術情報メディアセンター 研究開発部 ネットワーク開発部門 教授
著者
村井　　均　　地球シミュレータセンター 技術員

現在、並列計算機上のプログラミング手段としてはMessage Passing Interface(MPI)が広く用いられているが、プログラミングの容易さという点でHigh Performance Fortran(HPF)が優れている。本プロジェクトは、HPFの有効性を検証するとともにその長所と短所を明らかにし、HPFの言語と処理系に対する改善項目を見出すことを目的とする。

我々は、HPF/ESを用いてプラズマシミュレーションコードIMPACT-3Dを並列化し、地球シミュレータの512ノードを使って14.9 Tflopsという性能を達成した。この結果から、HPF/ESで並列化されたプログラムが良好なスケーラビリティを示し、HPFが実シミュレーションコードの開発に有用であることがわかる。また、我々はHPF/ESを用いて種々のベンチマークプログラムの評価も行った。

キーワード：High Performance Fortran、HPF、ベンチマーク評価、プラズマシミュレーション