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

Author
Hitoshi Murai The Earth Simulator Center

High Performance Fortran (HPF) is provided for parallelizing your programs on the Earth Simulator (ES). We developed an optimal implementation of NAS Parallel Benchmarks (NPB) and some other benchmarks with the HPF compiler available on the ES, namely HPF/ES, and evaluated them on the ES. The result shows that the HPF implementation can achieve performance comparable to the MPI (NPB2.4) in the programs of BT, SP, CG and MG, and nearly close in LU, each of which are from NPB. Thus, it can be said that a good HPF program written by fully exploiting the HPF features could be equal to that of MPI in performance. We also studied language specifications and features of HPF required for easier and more efficient parallelization, from the detailed comparison of the implementation and evaluation by HPF with those by MPI. In addition, we parallelized and ran two real-world applications with HPF/ES to evaluate the parallelization capability of HPF/ES.

Keywords: High Performance Fortran, HPF, benchmark evaluation, parallelization, NAS Parallel Benchmarks, NPB

1. Introduction

We believe that a parallelizing means not only easier to use than but also as efficient as MPI is essential for future parallel supercomputing, and that High Performance Fortran (HPF) can play the role.

An HPF compiler HPF/ES is provided on the Earth Simulator (ES). We plan to parallelize large-scale real applications from various fields, such as atmosphere, ocean, plasma, FEM and aerodynamics, with HPF to evaluate them on the ES and investigate the results in detail for more effective usage of HPF. 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 processors to fully exploit the performance of the ES. The required improvements and new features of the HPF compilers will be detected and proposed.

2. Benchmark Evaluation

We evaluate many benchmark programs from the NAS Parallel Benchmarks[1], SPEC OpenMP[2], the HPFBench benchmark suite[3], etc. This work aims to evaluate parallelization capability of HPF/ES and detect both of its advantages and disadvantages toward further development.

2.1. NAS Parallel Benchmarks

NAS Parallel Benchmarks (NPB) is a set of eight programs designed to help evaluate the performance of parallel supercomputers. The benchmarks consist of five kernels and three pseudo-applications[1].

From the eight programs of NPB we parallelize and evaluate the following five programs:

- BT
- SP
- LU
- CG
- MG

In this report, we describe only the implementation of MG for lack of space, before showing the evaluation results. See [4] for the implementation of other benchmarks.

2.1.1. MG implementation

MG is a benchmark program which solves a three-dimensional Poisson equation using the multigrid method. The key issue to parallelize the MG code is the following two[5]:

- hierarchical execution; and
- distribution of the hierarchical data.

The first issue can be solved by recursive procedure call and the second by hierarchical data alignment, each of which is to be described in the following sections.

The advantage of our method is that a procedure handles the hierarchical data of only two levels of the hierarchical execution, instead of possibly infinite levels. Thus, unlike the conventional HPF implementations, our method is free of separate data declarations and procedure calls for each level, wasteful memory allocation, and heavy communications, all of which may cause significant performance degradation.

(a) Hierarchical Execution

Fig. 1 illustrates the dataflow of the hierarchical data in the hierarchical execution of MG.

In the original implementation of NPB, the dataflow is expressed in the sequence of iterative procedure calls (Fig. 2), but it can be parallelized neither efficiently nor in a simple construct by HPF because the original way of argument passing is rather odd and not suited to the HPF specification.

We solved this problem by expressing the dataflow in the form of a recursive procedure call, shown in Fig. 3. Here, the procedure **mg3P** is in charge of the task of only one level and can be called recursively.

(b) Hierarchical Alignment

If the hierarchical data are aligned as shown in Fig. 4, no communication is required in the inter-level data exchange, namely prolongation and restriction. To realize this alignment, each of the hierarchical data is aligned with the template (the most dense data) in the stride determined on the basis of its level.

The recursive procedure **mg3P** obtains the level of hierarchical execution, *k*, through the dummy argument, and can align the hierarchical data **r** and **u** in the stride of 2^{lt-k}, as shown in Fig. 5, to eliminate redundant communications. It also allocates the data of the level *k-1*, **u2** and **r2** in Fig. 5, which are recursively passed to the lower-level **mg3p**, and aligns them in the stride of 2^{lt-k+1}.

```
subroutine mg3P(u,v,r, ..., k)

do  k = lt, 2, -1
   j = k-1
   call rprj3(r(ir(k)), ..., r(ir(j)), ..., k)
enddo

call psinv(r(ir(1)),u(ir(1)), ..., 1)

do  k = 2, lt-1
   j = k-1
   call interp(u(ir(j)), ..., u(ir(k)), ..., k)
   call resid(u(ir(k)),r(ir(k)), ..., k)
   call psinv(r(ir(k)),u(ir(k)), ..., k)
enddo

end
```

Fig. 2 Original Implementation of the Hierarchical Execution

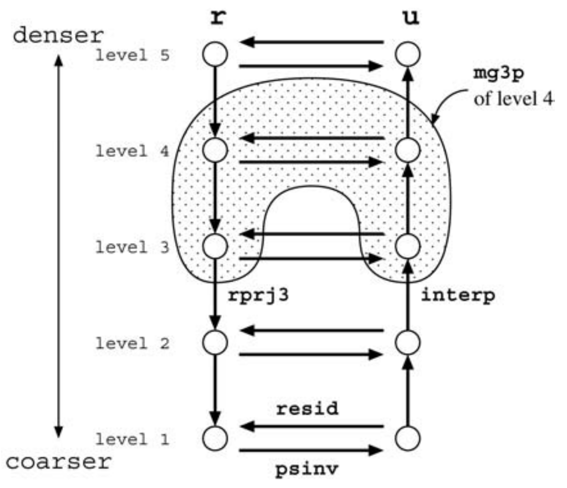

Fig. 1 Dataflow in MG

```
recursive subroutine mg3P(u,v,r, ..., k)

call rprj3(r, ..., r2, ..., k)

if (k > 2) then
   call mg3p(u2,v,r2, ..., k-1)
else
  call psinv(r,u, ..., 1)
end if

call interp(u2, ..., u, ..., k)
call resid(u,r,r, ..., k)
call psinv(r,u, ..., k)

end
```

Fig. 3 Our Implementation of the Hierarchical Execution

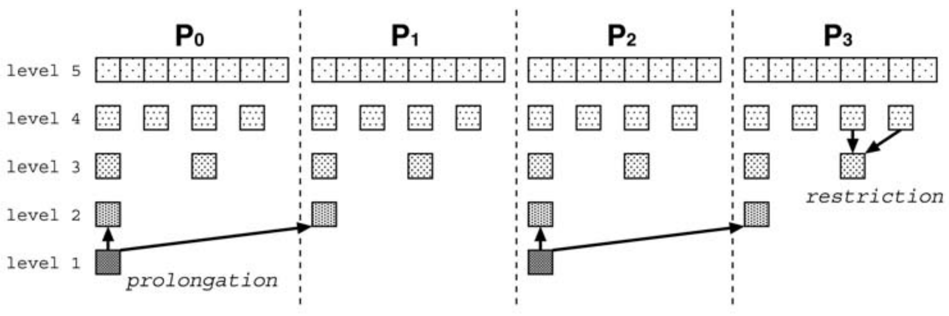

Fig. 4 Desired Alignment of the Hierarchical Data in MG (For simplicity, 1-dimensional arrays are used.)

2.1.2. Evaluation Results

The evaluation results are shown in Fig. 6. The versions of MPI and NPB3.0-HPF are also evaluated for comparison. In addition, some variant implementations are evaluated for LU and CG. The size of each benchmark is Class C.

These results show that our HPF implementation achieves performance comparable or superior to the MPI in BT, SP, CG and MG, and nearly close in LU.

```
!HPF$ template t(N)
!HPF$ distribute (block) :: t

      double precision u(n), r(n)
!HPF$ align (i) with *t(i*(2**(lt-k))-m) :: u, r

      double precision, allocatable :: u2(:), r2(:)
!HPF$ align (i) with t(i*(2**(lt-k+1))-m2) :: u2, r2
```

Fig. 5 Specification of the Hierarchical Alignment

We run the same programs also on a PC-cluster system with *HPF/ES for PC cluster*, a ported version of HPF/ES. The results show some properties rather different from those on the ES. Although we suppose that this difference mainly comes from whether vector processing is available or not, or to what extent the MPI libraries are optimized in each environment, further studies and analyses are necessary to explain the exact reason of the difference.

2.2. Other Benchmarks

We also implement the benchmarks from SPEC OpenMP and HPFBench with HPF/ES or other HPF compilers, and do a cross-platform evaluation of them on the ES or other parallel computers.

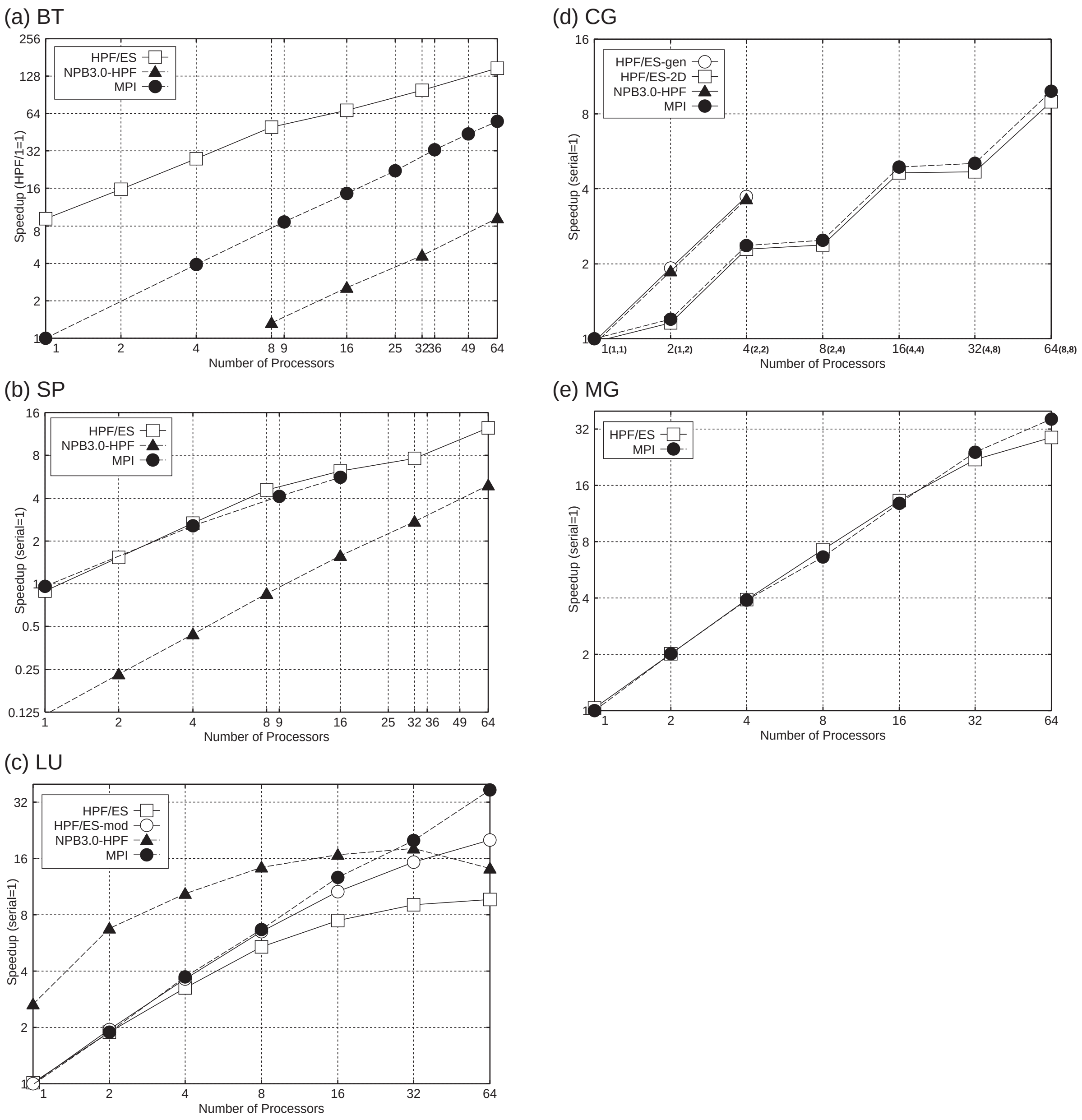

Fig. 6 Evaluation Results of NPB

2.3 Features in demand

Our study described above reveals that the following three features are required to achieve higher performance especially in LU from NPB.

(a) pipelined execution of the DOACROSS-style loops
(b) overlapped execution on the shadow region (EXT_HOME)
(c) partial REFLECT

It is one of the largest drawbacks of the current HPF/ES that it cannot handle efficiently *unparallelizable* loops, including the DOACROSS-style loops. Therefore, the feature (a) is considered to be very important to make HPF/ES more applicable. The feature (b) is also in demand because it is one of the HPF/JA extensions. The feature (c), which was proposed in [6], is effective in improving the performance of the REFLECT communication in some cases.

On the other hand, although more flexible data distributions other than one-dimensional BLOCK distribution, such as two-dimensional distribution, *multi-partition* proposed in [6], etc., are considered to be useful, it is shown by our evaluation that some of them do not work so efficiently or are not be supported by HPF/ES. They are also to be improved or supported in the future development.

3. Applications

Two real-world applications are parallelized with HPF/ES to evaluate the parallelization capability of HPF/ES:

• global atmospheric simulation

This is based on the Global Atmospheric Model (GAM) originally developed by the Australian Bureau of Meteorology. We plan to parallelize the code with HPF/ES and compare it with the MPI implementation to evaluate the expressiveness and performance of HPF/ES. This work is now progressing and to be continued to FY2004.

• plasma simulation

Although the code Impact-3d has been moved to another ES research project and is now used practically for fusion science, we continue to run it to evaluate the communication performance or parallel I/O of HPF/ES on a large-scale application.

4. Conclusion and Future Works

The optimal implementation of NPB with HPF/ES is studied to evaluate the parallelization capability of HPF/ES. The result shows that a good HPF program written by fully exploiting the HPF features could be equal to that of MPI in performance. In addition, two real-world applications are parallelized with HPF/ES and evaluated on the ES.

There are following future works planned:

• evaluation of real-world applications (contd.);
• cross-platform evaluation of benchmarks; and
• evaluation of proposed language features.

References

1) D.E. Bailey, et al., "The NAS Parallel Benchmarks," Technical Report RNR-94-007, NASA Ames Research Center, 1994.
2) Standard Performance Evaluation Corporation, "SPEC OMP," http://www.spec.org/omp/.
3) 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.
4) H. Murai and Y. Okabe, "Implementation and Evaluation of NAS Parallel Benchmarks with HPF on the Earth Simulator," In proc. of SACSIS2004, Sapporo, Japan, May 2004.
5) B.L. Chamberlain, S.J. Deits, and L. Snyder, "A Comparative Study of the NAS MG Benchmark across Parallel Languages and Architectures," In proc. of SC2000, Dallas, USA, Nov. 2000.
6) D. Chavarria-Miranda and J. Mellor-Crummey, "An Evaluation of Data-Parallel Compiler Support for Line-Sweep Applications," Journal of Instruction Level Parallelism, vol. 5, 2003.