International Journal of Parallel Programming, Vol. 31, No. 3, June 2003 (© 2003)

Performance Evaluation of the Hitachi SR8000 Using SPEC OMP2001 Benchmarks

Daisuke Takahashi,[1] Mitsuhisa Sato,[1] and Taisuke Boku[1]

This paper reports the performance of a single node of the Hitachi SR8000 when using SPEC OMP2001 benchmarks. Each processing node of the SR8000 is a shared-memory parallel computer composed of eight scalar processors with pseudo-vector processing feature. We have run the all of the SPEC OMP2001 benchmarks on the SR8000. According to the results of this performance measurement, we found that the SR8000 has good scalability continuing up to 8 processors except for a few benchmark programs. The performance results demonstrate that the SR8000 achieves high performance especially for memory-intensive applications.

KEY WORDS: OpenMP; SPEC OMP2001; shared-memory parallel computer; Hitachi SR8000.

1. INTRODUCTION

OpenMP[(1)] has emerged as the standard programming model for shared-memory parallel programming. The OpenMP Application Program Interface (API) supports multi-platform shared-memory parallel programming in C/C++ and Fortran on all architectures, including Unix and Windows NT platforms.

Many computer hardware and software vendors have endorsed the OpenMP API and have released commercial compilers that can compile an OpenMP parallel program. Hitachi supports an OpenMP compiler[(2)] that generates native codes for the Hitachi SR8000.

[1] Institute of Information Sciences and Electronics, University of Tsukuba, 1-1-1 Tennodai, Tsukuba-shi, Ibaraki 305-8573, Japan. E-mail: {daisuke,msato,taisuke}@is.tsukuba.ac.jp

0885-7458/03/0600-0185/0 © 2003 Plenum Publishing Corporation

The purpose of this paper is to evaluate the performance of SPEC OMP2001 benchmarks on a single node of the Hitachi SR8000. Although a large system of Hitachi SR8000 may contain several nodes which are connected by their high-speed network, we restrict ourselves to a single node in this paper. A single node of the SR8000 is considered as a shared-memory parallel computer composed of eight scalar processors with pseudo-vector processing feature.[3–7] The automatic parallelizing compiler is provided to perform parallelization for the shared-memory multiprocessors within each node of the SR8000.

We chose the SPEC OMP2001 benchmarks[8] for performance evaluation. SPEC OMP2001 is SPEC's first benchmark suite for evaluating performance of OpenMP applications.

Aslot *et al.*[9] have presented this benchmark suite and described issues encountered in the creation of the OpenMP benchmarks. In addition, some SPEC OMP2001 benchmark results have been presented.[9–12] Some performance characteristics of the SPEC OMP2001 benchmarks have been also reported.[9, 11]

The rest of the paper is organized as follows. Section 2 describes the SPEC OMP2001 benchmarks. Section 3 describes the Hitachi SR8000 architecture and its unique features. Section 4 gives the performance results. In Section 5, we present some concluding remarks.

2. SPEC OMP2001 BENCHMARKS

2.1. SPEC OMPM2001 Benchmarks

The SPEC OMPM2001[8] benchmark suite consists of 11 large application programs that represent the type of software used in scientific technical computing. The SPEC OMPM2001 is developed for medium-scale (4- to 16-way) systems.

All applications, except for *gafort*, are floating-point applications taken directly from the SPEC CPU2000 benchmark suite. Each application is either automatically or manually parallelized by inserting OpenMP directives to mark parallel regions of the code.[9] SPEC OMPM2001 uses medium-sized workloads that require 1.6 GB of memory and take an hour and a half each to run on a 350 MHz, four-processor reference machine.

Wupwise (Wuppertal Wilson Fermion Solver) is a program in the area of lattice gauge theory (quantum chromodynamics). Lattice gauge theory is a discretization of quantum chromodynamics, which is generally accepted to be the fundamental physical theory of strong interactions among quarks as the constituents of matter. Quark propagators are obtained by solving the inhomogeneous lattice-Dirac equation. The *wupwise* solves the

inhomogeneous lattice-Dirac equation via the BiCGStab iterative method, which has become established as the method of choice. Its Fortran source code is 2200 lines long.

Swim is a weather prediction program. The model is based on the dynamics of finite-difference models of the shallow-water equations. Its Fortran source code is 400 lines long.

Mgrid is a very simple multigrid solver that computes a three-dimensional potential field. SPEC adapted it from the NAS Parallel Benchmarks, with modifications for portability and a different workload. Its Fortran source code is 500 lines long.

Applu is a solution of five coupled nonlinear PDE's, on a 3-dimensional logically structured grid, using an implicit pseudo-time marching scheme, based on two-factor approximate factorization of the sparse Jacobian matrix. Its Fortran source code is 4000 lines long.

Galgel is a particular case of the GAMM (Gesellschaft für Angewandte Mathematik und Mechanik) benchmark devoted to numerical analysis of oscillatory instability of convection in low-Prandtl-number fluids. Its Fortran source code is 15300 lines long.

Equake simulates the propagation of elastic waves in large, highly heterogeneous valleys, such as California's San Fernando Valley, or the Greater Los Angeles Basin. Computations are performed on an unstructured mesh that locally resolves wavelengths, using a finite element method. Its Fortran source code is 1500 lines long.

Apsi is a program to solve for mesoscale and synoptic variations in potential temperature. Its Fortran source code is 7500 lines long.

Gafort computes the global maximum fitness using a genetic algorithm. Its Fortran source code is 1500 lines long.

Fma3d is a finite element method computer program designed to simulate the inelastic, transient dynamic response of three-dimensional solids and structures subjected to impulsively or suddenly applied loads. Its Fortran source code is over 60000 lines long.

Art (Adaptive Resonance Theory) is neural network, which is used to recognize objects in a thermal image. Its C source code is 1300 lines long.

Ammp (Another Molecular Modeling Program) runs molecular dynamics (i.e., solves the ODE defined by Newton's equations for the motions of the atoms in the system) on a protein-inhibitor complex which is embedded in water. Its C source code is 13500 lines long.

2.2. SPEC OMPL2001 Benchmarks

The SPEC OMPL2001[8] benchmark suite consists of 9 large application programs. In contrast to the SPEC OMPM2001 suite, the target

problem size (working set size and run time) is approximately 4 times to 8 times larger than SPEC OMPM2001. The SPEC OMPL2001 is developed for large-scale (32-way and larger) systems.

Application benchmarks running under SPEC OMPL2001 use up to 6.4 GB of memory and take approximately four hours each to run on a 300 MHz, 16-processor reference machine.

3. ARCHITECTURE OF THE HITACHI SR8000

The Hitachi SR8000 System is a distributed-memory parallel computer which consists of pseudo-vector SMP nodes. An SR8000 system consists of 4 to 512 nodes. Figure 1 shows an overview of the SR8000 system architecture.

The node processor of the SR8000 model G1 is a 2.2 ns PowerPC node with major enhancements made by Hitachi. For example, Hitachi added hardware barrier synchronization mechanism is provided for fast barrier operation. Each node contains eight RISC microprocessors which have the "Pseudo Vector Processing" (PVP) feature.[3–6] This allows data to be fetched from main memory, in a pipelined manner, without stalling the succeeding instructions. The result is that data is fed from memory into the arithmetic units as effectively as in a vector supercomputer. This feature was already available on the CP-PACS,[3] and experiments have shown that this idea works well for vector processing.

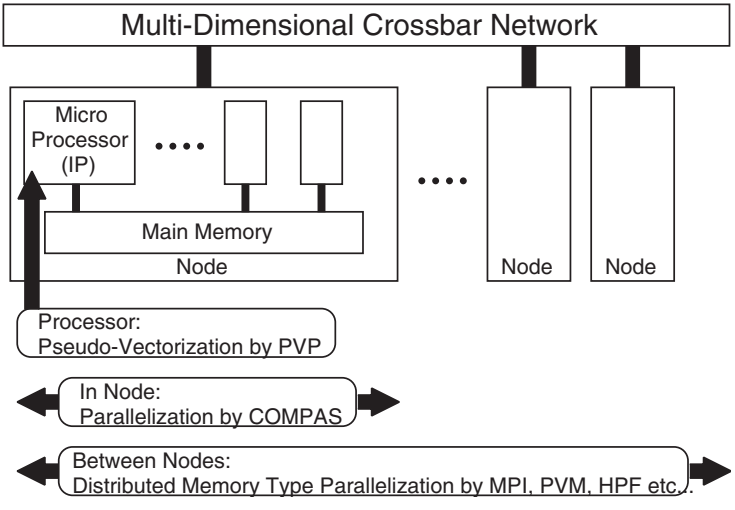

Fig. 1. Architecture of the SR8000 system.

The peak performance per processor, or IP, can be attained with two simultaneous multiply/add instructions, resulting in a speed of 1.8 GFLOPS on the SR8000 model G1. The processor has a primary write-through 128 KB 4-way set associative on-chip data cache with 128-byte cache lines. The instruction cache is 2-way set associative 64 KB on-chip cache.

Eight processors are coupled to form one processing node all addressing a common part of the memory. Hitachi refers to this node configuration as COMPAS, **C**o-**o**perative **M**icro-**P**rosessors in single **A**ddress **S**pace. Peak performance of a node of the SR8000 model G1 is 14.4 GFLOPS and maximum memory capacity is 16 GB.

The nodes on the SR8000 are interconnected through a multidimensional crossbar network. The communication bandwidth available at a node of the SR8000 model G1 is 1.6 GB/s (single direction) $\times 2$.

The automatic parallelizing compiler is provided to perform parallelization for the shared-memory multiprocessors within each node of the SR8000, using the hardware's synchronization mechanism to perform high-speed parallel execution.[2] The performance of automatic parallelization was described in reference.[2]

4. PERFORMANCE RESULTS

This section presents the performance results of SPEC OMP2001 benchmarks on the Hitachi SR8000. All measurements were done on a single node of the SR8000 G1 model (PowerPC + PVP 450 MHz, 16 GB main memory size, 14.4 GFLOPS peak performance, HI-UX/MPP 03-07).

4.1. SPEC OMPM2001 Performance Results

The eight SPEC OMPM2001 benchmark programs written in Fortran (*wupwise*, *swim*, *mgrid*, *applu*, *galgel*, *apsi*, *gafort* and *fma3d*) that we used were compiled with Hitachi's Optimizing Fortran 90 compiler V01-05 under the options "`f90 -Oss -omp -model=G1 -64`". These options instruct the compiler to set the optimization level to maximize the execution speed ("`-Oss`"), to specify whether to enable OpenMP directives ("`-omp`"), to generate an object for the G1 model of the SR8000 ("`-model=G1`") and to use 64-bit addressing mode ("`-64`"), respectively.

For *galgel*, we used portability flag "`-fixed=132`" which specifies the 132 columns that can be written in one line. For *fma3d*, we also used portability flag "`-conti199`" is that up to 199 continuation lines can be written.

The three SPEC OMPM2001 benchmark programs written in C (*equake*, *art* and *ammp*) that we used were compiled with Hitachi's

Optimizing C compiler V01-04-/A under the options "`cc -Os -omp -parallel=4 -model=G1 -64`". These options instruct the compiler to specify various optimizing options including element parallelization and pseudo vectorization so that a user program can execute at the maximum speed ("`-Os`"), to specify whether to enable OpenMP directives ("`-omp`"), to set the parallelization level to level 4 ("`-parallel=4`"), to generate an object for the G1 model of the SR8000 ("`-model=G1`") and to use 64-bit addressing mode ("`-64`"), respectively.

We ran the codes on a single node of the SR8000-G1 using "ref" data sets. Table I shows the result of all SPEC OMPM2001 benchmarks up to 8 processors. The first column of a table indicates the benchmarks used. The second column gives the "Reference Time" in seconds. The next four columns contain the "Estimated Runtime" in seconds. Since the results shown in Table I have not yet been reviewed by SPEC, they are marked as "Estimated Runtime".

Figure 2 shows the speedup of SPEC OMPM2001 benchmarks relative to the one-processor execution time on a single node of the SR8000-G1. The figure shows that the speedup on 8 processors is about 2.1 to 7.8 for the benchmarks.

Mgrid, *galgel*, *fma3d* and *ammp* speedup between 5.0 and 7.0 on 8 processors, whereas *wupwise*, *swim*, *applu*, *apsi*, *gafort* and *art* speedup by a factor over 7.0 on 8 processors. *Wupwise* uses OpenMP versions of the LAPACK routines (`dznrm2.f zaxpy.f zcopy.f zdotc.f zscal.f`). Thus, *wupwise* achieved almost perfect scaling on the SR8000-G1.

Table I. SPEC OMPM2001 results on the SR8000-G1

Benchmark	Reference time	Estimated runtime			
		1 CPU	2 CPUs	4 CPUs	8 CPUs
310.wupwise_m	6000	13604	6908	3483	1788
312.swim_m	6000	1991	1007	513	267
314.mgrid_m	7300	6999	3813	2132	1006
316.applu_m	4000	4480	2321	1174	609
318.galgel_m	5100	4415	7847	8014	865
320.equake_m	2600	6621	4901	3778	3187
324.apsi_m	3400	5071	2625	1336	687
326.gafort_m	8700	21837	11047	5536	2810
328.fma3d_m	4600	12042	6269	3264	1763
330.art_m	6400	21792	11277	5768	2996
332.ammp_m	7000	44971	29763	16777	8993

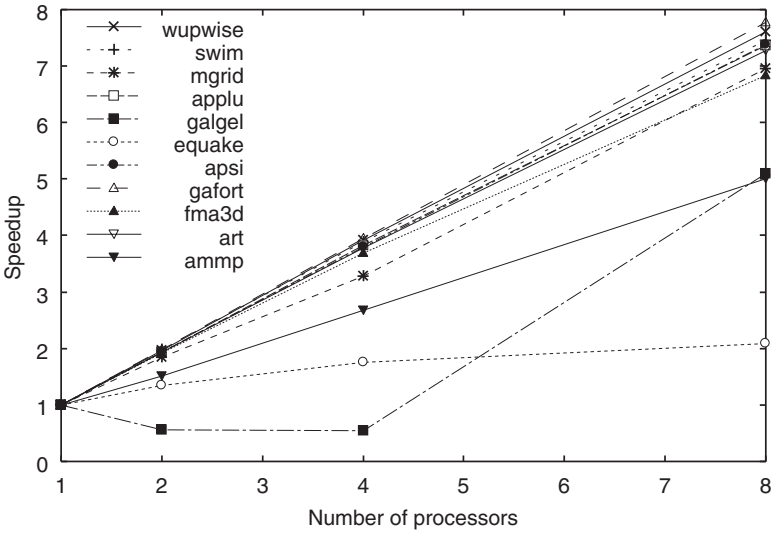

Fig. 2. Speedup of SPEC OMPM2001 on the SR8000-G1.

In *galgel*, the speedup is under 1.0 on 2 and 4 processors, as shown in Fig. 2. Its speedup loss comes mainly from fork-join overhead. The key of the performance in *galgel* is a `PARALLEL DO` directive. *Galgel* includes about 90 `PARALLEL DO` blocks and all of them enclose only a few assignment statements without nested-DO loops.[10] Moreover, the fork-join overhead on 2 and 4 processor executions is relatively high compared to on 8 processors execution due to implementation of Hitachi's OpenMP compiler. This is the reason why *galgel* shows poor speedup on 2 and 4 processors.

Similar to *galgel*, *equake* includes many `parallel for` directive which contains trivial computation. Thus, *equake* shows the lowest speedup among all of the SPEC OMPM2001 benchmarks.

Table II compares the SPEC OMPM2001 results on the Hitachi SR8000 G1 model (PowerPC + PVP 450 MHz, 64 KB L1 instruction cache, 128 KB L1 data cache, 8 CPUs), hp server rp7410 (PA-8700 + 875 MHz, 0.75 MB L1 instruction cache, 1.5 MB L1 data cache, 8 CPUs)[8] and SGI 2800 (MIPS R12000 400 MHz, 32 KB L1 instruction cache, 32 KB L1 data cache, 8 MB L2 cache, 32 CPUs).[8] The first column of a table indicates the benchmarks used. The second column gives the "Reference Time" in seconds. The third column gives the "Estimated Runtime" in seconds. The next two columns contain the "Base Runtime" in seconds.

Table II. SPEC OMPM2001 Results

Benchmark	Reference time	Estimated runtime SR8000-G1 (8 CPUs)	Base runtime hp server rp7410 (8 CPUs)	Base runtime SGI 2800 (32 CPUs)
310.wupwise_m	6000	1788	1279	582
312.swim_m	6000	267	1306	804
314.mgrid_m	7300	1006	1826	795
316.applu_m	4000	609	830	420
318.galgel_m	5100	865	1659	1009
320.equake_m	2600	3187	619	416
324.apsi_m	3400	687	797	515
326.gafort_m	8700	2810	1885	1499
328.fma3d_m	4600	1763	1135	752
330.art_m	6400	2996	707	956
332.ammp_m	7000	8993	2408	1615

We found that the performance of the *swim* on the SR8000-G1 is better than that of the *swim* on the hp server rp7410 and SGI 2800. This is mainly because the *swim* is a very memory-intensive program.[11] The SR8000-G1 can hide memory latency effectively with software managed controlling of data movement by pseudo-vector processing feature.

On the other hand, the performances of the some benchmark programs (*wupwise*, *equake*, *gafort*, *fma3d*, *art* and *ammp*) are lower than those of both the hp server rp7410 and SGI 2800. In particular, with the *equake* and *ammp*, the runtime of the SR8000-G1 (8 CPUs) is slower than the "Reference Time". *Ammp* includes many innermost loops that cannot be pseudo-vectorized due to data dependency.

Although HP's PA-8700+processor has 1.5 MB primary data cache (and no secondary cache) and the R12000 processor in SGI 2800 has an 8 MB off-chip unified secondary cache, the processor of the SR8000-G1 has only 128 KB primary data cache (and no secondary cache). This is the reason why *ammp* shows poor performance on the SR8000-G1.

4.2. SPEC OMPL2001 Performance Results

The seven SPEC OMPL2001 benchmark programs written in Fortran (*wupwise*, *swim*, *mgrid*, *applu*, *apsi*, *gafort* and *fma3d*) that we used were compiled with Hitachi's Optimizing Fortran 90 compiler V01-05 under the options "`f90 -Oss -omp -model=G1 -64`".

The two SPEC OMPL2001 benchmark programs written in C (*equake* and *art*) that we used were compiled with Hitachi's Optimizing C compiler

Table III. SPEC OMPL2001 Results on the SR8000-G1

Benchmark	Reference time	Estimated runtime			
		1 CPU	2 CPUs	4 CPUs	8 CPUs
311.wupwise_l	9200	72644	37430	18829	9591
313.swim_l	12500	16513	8172	4162	2162
315.mgrid_l	13500	36080	18716	9827	5157
317.applu_l	13500	61991	31941	16289	8313
321.equake_l	13000	137342	143276	82708	68931
325.apsi_l	10500	35618	18861	9498	4806
327.gafort_l	11000	84788	42972	21651	10950
329.fma3d_l	23500	162142	85001	44431	23850
331.art_l	25000	[a]	148774	74102	37046

[a] Means that we were not able to execute because CPU time limit exceeded.

V01-04-/A under the options "`cc -Os -omp -parallel=4 -model=G1 -64`".

We ran the codes on a single node of the SR8000-G1 using "ref" data sets. Table III shows the result of all SPEC OMPL2001 benchmarks up to 8 processors. The first column of a table indicates the benchmarks used. The second column gives the "Reference Time" in seconds. The next four columns contain the "Estimated Runtime" in seconds. Since the results shown in Table III have not yet been reviewed by SPEC, they are marked as "Estimated Runtime".

Figure 3 shows the speedup of SPEC OMPL2001 benchmarks (except for *art*) relative to the one-processor execution time on a single node of the SR8000-G1. The figure shows that the speedup on 8 processors is about 2.0 to 7.7 for the benchmarks.

Mgrid and *fma3d* speedup between 6.8 and 7.0 on 8 processors, whereas *wupwise*, *swim*, *applu*, *apsi* and *gafort* speedup by a factor over 7.0 on 8 processors.

Similar to the SPEC OMPM2001 results, *equake* shows the lowest speedup among all of the SPEC OMPL2001 benchmarks.

Table IV compares the SPEC OMPL2001 results on the Hitachi SR8000 G1 model (PowerPC + PVP 450 MHz, 64 KB L1 instruction cache, 128 KB L1 data cache, 8 CPUs), hp server rp7410 (PA-8700 + 875 MHz, 0.75 MB L1 instruction cache, 1.5 MB L1 data cache, 8 CPUs)[8] and SGI 3800 (MIPS R12000 400 MHz, 32 KB L1 instruction cache, 32 KB L1 data cache, 8 MB L2 cache, 32 CPUs).[8] The first column of a table indicates the benchmarks used. The second column gives the "Reference Time" in

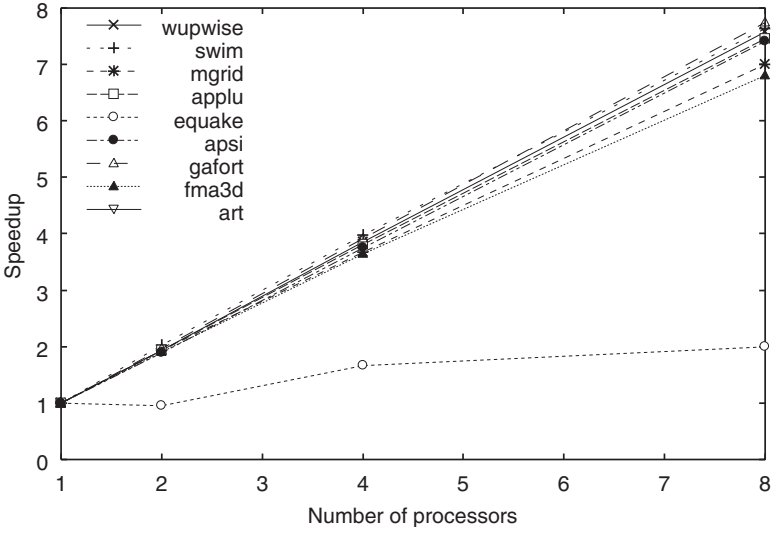

Fig. 3. Speedup of SPEC OMPL2001 on the SR8000-G1.

seconds. The third column gives the "Estimated Runtime" in seconds. The next two columns contain the "Base Runtime" in seconds.

As Fig. 3 indicates, the tendency of the speedup is almost the same as that shown in the results of Fig. 2. The scalability of *equake* has a dip between 1 and 4 processors, and this is under investigation.

Table IV. SPEC OMPL2001 Results

		Estimated runtime	Base runtime	
Benchmark	Reference time	SR8000-G1 (8 CPUs)	hp server rp7410 (8 CPUs)	SGI 2800 (32 CPUs)
311.wupwise_l	9200	9591	6165	2294
313.swim_l	12500	2162	10790	3659
315.mgrid_l	13500	5157	9739	2620
317.applu_l	13500	8313	11482	3071
321.equake_l	13000	68931	11700	3775
325.apsi_l	10500	4806	11638	3522
327.gafort_l	11000	10950	8115	3086
329.fma3d_l	23500	23850	15365	6606
331.art_l	25000	37046	7230	7480

5. CONCLUDING REMARKS

In this paper, we evaluated the SPEC OMP2001 benchmarks on a single node of the Hitachi SR8000.

For the SPEC OMPM2001 benchmarks evaluated, we observed about 2.1 to 7.8 times speedup on 8 processors. On the other hand, for the SPEC OMPL2001 benchmarks evaluated, we observed about 2.0 to 7.7 times speedup on 8 processors.

A few benchmark programs include many innermost loops that cannot be pseudo-vectorized due to data dependency. Moreover, the processor of the SR8000 has no secondary cache. This is the reason why these benchmark programs show poor performance on the SR8000.

According to the results of our performance measurement, we found that the SR8000 has good scalability continuing up to 8 processors except for a few benchmark programs.

The SR8000 can hide memory latency effectively with software managed controlling of data movement by pseudo-vector processing feature. The performance results demonstrate that the SR8000 achieves high performance especially for memory-intensive applications.

REFERENCES

1. OpenMP, Simple, Portable, Scalable SMP Programming, http://www.openmp.org.
2. Y. Nishitani, K. Negishi, H. Ohta, and E. Nunohiro, Implementation and Evaluation of OpenMP for Hitachi SR8000, in *Proc. Third International Symposium on High Performance Computing (ISHPC 2000)*, Vol. 1940, Lecture Notes in Computer Science, Springer-Verlag, (2000), pp. 391–402.
3. K. Nakazawa, H. Nakamura, T. Boku, I. Nakata, and Y. Yamashita, CP-PACS: A Massively Parallel Processor at the University of Tsukuba, *Parallel Computing* **25**:1635–1661 (1999).
4. Y. Tamaki, N. Sukegawa, M. Ito, Y. Tanaka, M. Fukagawa, T. Sumimoto, and N. Ioki, Node Architecture and Performance Evaluation of the Hitachi Super Technical Server SR8000, In *Proc. 12th International Conference on Parallel and Distributed Computing Systems* (1999), pp. 487–493.
5. K. Shimada, T. Kawashimo, M. Hanawa, R. Yamagata, and E. Kamada, A Superscalar RISC Processor with 160 FPRs for Large Scale Scientific Processing, in *Proc. International Conference on Computer Design (ICCD'99)* (1999), pp. 279–280.
6. H. Nishiyama, K. Motokawa, I. Kyushima, and S. Kikuchi, Pseudo-Vectorizing Compiler for the SR8000, in *Proc. 6th International Euro-Par Conference (Euro-Par 2000)*, Vol. 1900, Lecture Notes in Computer Science, Springer-Verlag (2000), pp. 1023–1028.
7. M. Brehm, R. Bader, H. Heller, and R. Ebner, Pseudovectorization, SMP, and Message Passing on the Hitachi SR8000-F1, in *Proc. 6th International Euro-Par Conference (Euro-Par 2000)*, Vol. 1900, Lecture Notes in Computer Science, Springer-Verlag (2000), pp. 1351–1361.
8. Standard Performance Evaluation Corporation (SPEC), SPEComp 2001, http://www.spec.org/hpg/omp.

9. V. Aslot, M. Domeika, R. Eigenmann, G. Gaertner, W. B. Jones, and B. Parady, SPEComp: A New Benchmark Suite for Measuring Parallel Computer Performance, in *Proc. International Workshop on OpenMP Applications and Tools (WOMPAT 2001)*, Vol. 2104, Lecture Notes in Computer Science, Springer-Verlag (2001), pp. 1–10.
10. H. Iwashita, E. Yamanaka, N. Sueyasu, M. van Waveren, and K. Miura, The SPEC OMP2001 Benchmark on the Fujitsu PRIMEPOWER System, in *Proc. Third European Workshop on OpenMP (EWOMP 2001)* (2001).
11. V. Aslot and R. Eigenmann, Performance Characteristics of the SPEC OMP2001 Benchmarks, in *Proc. Third European Workshop on OpenMP (EWOMP 2001)* (2001).
12. H. Saito, G. Gaertner, W. B. Jones, R. Eigenmann, H. Iwashita, R. Lieberman, M. van Waveren, and B. Whitney, Large System Performance of SPEC OMP2001 Benchmarks, in *Proc. 4th International Symposium on High Performance Computing (ISHPC 2002)*, Vol. 2327, Lecture Notes in Computer Science, Springer-Verlag (2002), pp. 370–379.