Performance Evaluation of the Hitachi SR8000 Using OpenMP Benchmarks

Daisuke Takahashi, Mitsuhisa Sato and Taisuke Boku

Center for Computational Physics,
Institute of Information Sciences and Electronics, University of Tsukuba
1-1-1 Tennodai, Tsukuba-shi, Ibaraki 305-8577, Japan
{daisuke,msato,taisuke}@is.tsukuba.ac.jp

Abstract. This paper reports the performance of a single node of the Hitachi SR8000 when using OpenMP 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 and three benchmarks (LU, SP and BT) of the NAS Parallel 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.

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.

The purpose of this paper is to evaluate the performance of OpenMP benchmark programs on a single node of the Hitachi SR8000. Although a large system of Hitachi SR8000 may contain several nodes which is 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] and the OpenMP version of NAS Parallel Benchmarks (NPB) [9] for performance evaluation.

SPEC OMP2001 is SPEC's first benchmark suite for evaluating performance of OpenMP applications. Aslot et al. [10] 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 [10–12]. Some performance characteristics of the SPEC OMP2001 Benchmarks have been also reported [10, 12].

The NAS Parallel Benchmarks (NPB) [9] were designed to compare the performance of parallel computers and are widely recognized as a standard indicator of parallel computer performance. While the MPI version of NPB is available in public, its OpenMP version PBN (Programming Baseline for NPB) [13] is still restricted only for research. We parallelized NPB2.3-serial version by using OpenMP directives.

We have chose some benchmark programs from the NAS Parallel Benchmarks 2.3-serial version on the SR8000.

The rest of the paper is organized as follows. Section 2 describes the SPEC OMP2001 Benchmarks. Section 3 describes the NAS Parallel Benchmarks. Section 4 describes the Hitachi SR8000 architecture and its unique features. Section 5 gives the performance results. In section 6, we present some concluding remarks.

2 SPEC OMP2001 Benchmarks

The SPEC OMP2001 [8] benchmark suite consists of 11 large application programs that represent the type of software used in scientific technical computing.

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 [10]. Each benchmark requires up to 2 GB of memory at runtime and runs for up to 10 hours on a modern single processor system.

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 fuer 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.

3 The NAS Parallel Benchmarks

The NAS Parallel Benchmarks (NPB) were designed to compare the performance of parallel computers and are widely recognized as a standard indicator of computer performance. The NPB suite consists of five kernels and three simulated CFD applications derived from important classes of aerophysics applications. The five kernels mimic the computational cores of five numerical methods used by CFD applications. The simulated CFD applications reproduce much of the data movement and computation found in full CFD codes. Details of the benchmark specifications can be found in [9], and the MPI implementations of NPB are described in [14].

In this paper, we used three benchmarks (LU, SP and BT) from the serial version of NPB2.3. We parallelized the benchmarks using OpenMP directives.

The LU benchmark solves a finite difference discretization of 3-D compressive Navier-Stokes equations through a block-lower-triangular block-upper-triangular approximate factorization of the original difference scheme. The LU factored form is cast as a relaxation and is solved by SSOR.

SP (Scalar Pentadiagonal) equations and BT (Block Tridiagonal) equations form the basis of the 3-D CFD applications. Both are benchmark programs to solve partial differential equations, with SP based on the finite difference method

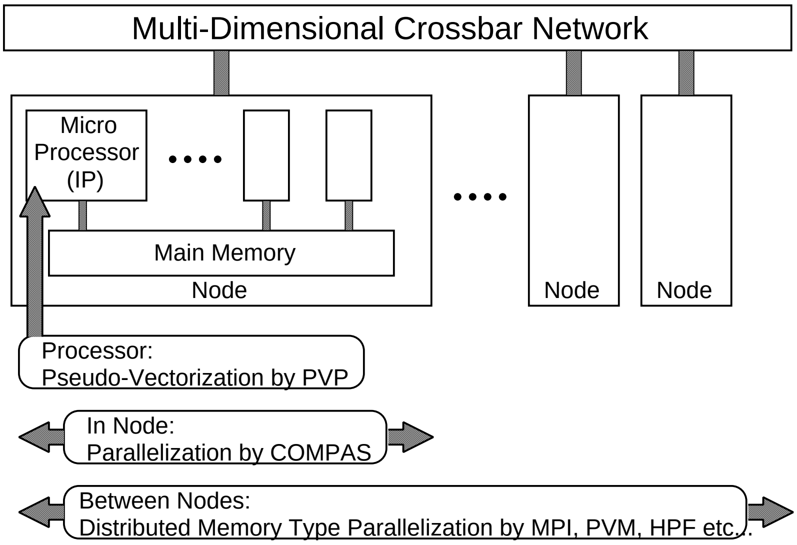

Fig. 1. Architecture of the SR8000 system

(Scalar ADI iteration), and BT based on the finite difference method (5×5 block ADI iteration).

4 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 F1 is a 2.67 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.

The peak performance per processor, or IP, can be attained with two simultaneous multiply/add instructions, resulting in a speed of 1.5 GFLOPS on the SR8000 model F1. The processor has a first-level 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

Table 1. SPEC OMP2001 results on the SR8000

Benchmark	Reference Time	Estimates Runtime			
		1 CPU	2 CPUs	4 CPUs	8 CPUs
wupwise	6000	16336	8312	4193	2158
swim	6000	2400	1226	627	332
mgrid	7300	8458	4636	2645	1257
applu	4000	5414	2801	1418	740
galgel	5100	5309	10045	10453	1042
equake	2600	7924	4768	2863	1854
apsi	3400	6055	3148	1608	830
gafort	8700	26358	13353	6727	3408
fma3d	4600	14372	7550	3946	2129
art	6400	21844	11460	5874	3075
ammp	7000	55664	36678	20465	10950

Table 2. SPEC OMP2001 results

Benchmark	Reference Time	Estimates Runtime	Base Runtime	
		SR8000 (8 CPUs)	Origin 3800 (8 CPUs)	PRIMEPOWER2000 (18 CPUs)
wupwise	6000	2158	1553	1080
swim	6000	332	1655	1211
mgrid	7300	1257	1813	1322
applu	4000	740	1071	435
galgel	5100	1042	905	1170
equake	2600	1854	670	735
apsi	3400	830	1195	439
gafort	8700	3408	2363	1680
fma3d	4600	2129	1590	1210
art	6400	3075	1988	1685
ammp	7000	10950	2957	2174

of a node of the SR8000 model F1 is 12 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 F1 is 1 GB/s (single direction) $\times$ 2.

Hitachi also provides an SR8000 compact C model. It has a peak performance of 12 GFLOPS and it can connect up to four SR8000 systems.

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 [2].

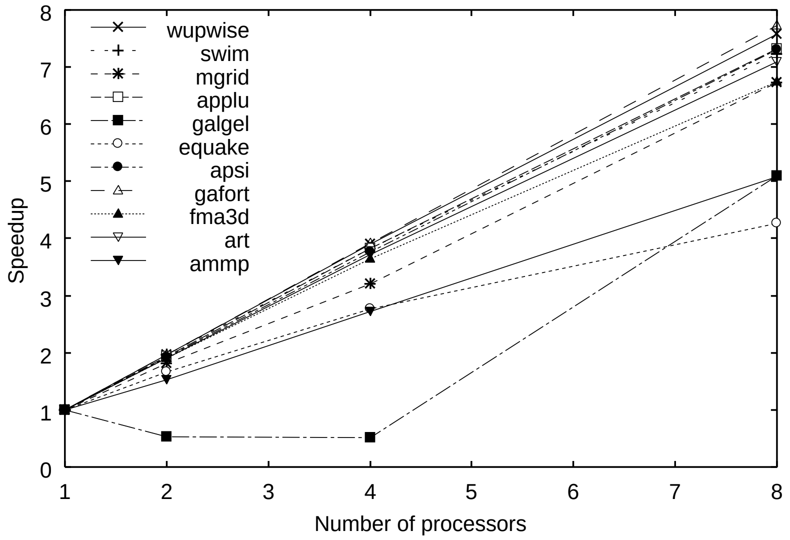

Fig. 2. Speedup of SPEC OMP2001 on the SR8000

5 Performance Results

This section presents the performance results of OpenMP programs on the Hitachi SR8000. All measurements were done on a single node of the SR8000 compact C model (PowerPC + PVP 375 MHz, 8 GB main memory size, 12 GFLOPS peak performance, HI-UX/MPP 03-05).

5.1 SPEC OMP2001 Performance Results

The eight SPEC OMP2001 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 V01-04-/B under the options "`f90 -Oss -omp -model=C -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 compact C model of the SR8000 ("`-model=C`") 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 OMP2001 benchmark programs written in C (*equake*, *art* and *ammp*) that we used were compiled with Hitachi's Optimizing C compiler V01-04 under the options "`cc -Os -omp -parallel=4 -model=C -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, to generate an object for the compact C model of the SR8000 ("`-model=C`") and to use 64-bit addressing mode ("`-64`"), respectively.

We ran the codes on a single node of the SR8000 using "ref" data sets. Table 1 shows the result of all SPEC OMP2001 Benchmarks (Medium size) 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 "Estimates Runtime" in seconds. Since the results shown in Table 1 have not yet been reviewed by SPEC, they are marked as "Estimates Runtime".

Figure 2 shows the speedup of SPEC OMP2001 Benchmarks relative to the one-processor execution time on a single node of the SR8000. The figure shows that the speedup on 8 processors is about 4.3 to 7.7 for the benchmarks.

Mgrid and *fma3d* speedup between 6.5 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.

In *galgel*, the speedup is under 1.0 on 2 and 4 processors, as shown in Figure 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 [11]. 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 OMP2001 Benchmarks.

Table 2 compares the results on the Hitachi SR8000 compact C model (PowerPC + PVP 375 MHz, 8 CPUs), SGI Origin 3800 (MIPS R14000 500 MHz, 8 CPUs) [8] and Fujitsu PRIMEPOWER2000 (SPARC64 GP 563 MHz, 18 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 "Estimates Runtime" in seconds. The next two columns contain the "Base Runtime" in seconds.

We found that the performance of the *swim* on the SR8000 is better than that of the swim on the Origin 3800 and PRIMEPOWER2000. This is mainly because the *swim* is a very memory-intensive program [12]. The SR8000 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 Origin 3800 and PRIMEPOWER2000. In particular, with the *ammp*, the runtime of SR8000 (8 CPUs) is slower than the "Reference Time". *Ammp* includes

Table 3. NPB2.3-serial (Class A) results on the SR8000

Benchmark	1 CPU		2 CPUs		4 CPUs		8 CPUs	
	Time	MFLOPS	Time	MFLOPS	Time	MFLOPS	Time	MFLOPS
LU	595.56	200.31	495.59	240.71	413.37	288.59	276.03	432.19
SP	291.02	292.11	155.15	547.93	85.69	992.03	40.59	2094.51
BT	584.33	287.99	303.13	555.15	168.39	999.37	102.75	1637.84

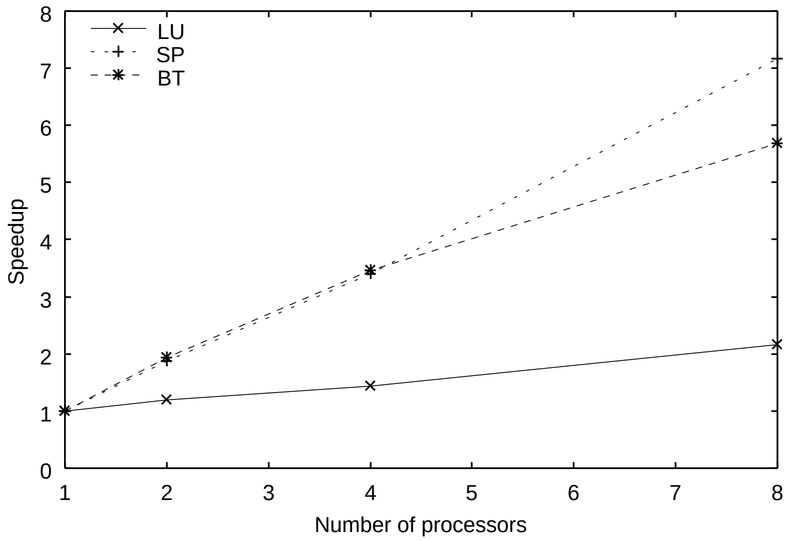

Fig. 3. Speedup of NPB2.3-serial (Class A) on the SR8000

many innermost loops that cannot be pseudo-vectorized due to data dependency. Although the Origin 3800 and PRIMEPOWER2000 have a second-level 8 MB off-chip cache, the processor of the SR8000 has no second-level cache. This is the reason why *ammp* shows poor performance on the SR8000.

5.2 NPB Performance Results

We measured the performance of a single node of the SR8000 by the NAS Parallel Benchmarks (NPB) [9].

The OpenMP versions of the NAS Parallel Benchmarks are derived from the serial versions of NAS Parallel Benchmarks 2.3 (NPB2.3-serial). We parallelized the benchmarks by inserting only the OMP directives. The code was compiled with Hitachi's Optimizing Fortran 90 compiler V01-04-/B. The compiler options used were "`f90 -Oss -omp -model=C -i,EU,L -64`". We ran the codes on a single node of the SR8000 using Class A and Class B data sets.

Table 4. NPB2.3-serial (Class B) results on the SR8000

Benchmark	1 CPU		2 CPUs		4 CPUs		8 CPUs	
	Time	MFLOPS	Time	MFLOPS	Time	MFLOPS	Time	MFLOPS
LU	2455.73	203.13	1851.08	269.48	1476.70	337.80	1150.83	433.45
SP	1196.93	296.60	619.81	572.77	316.29	1122.42	163.29	2174.09
BT	2579.87	272.18	1348.13	520.86	748.57	938.03	452.72	1551.03

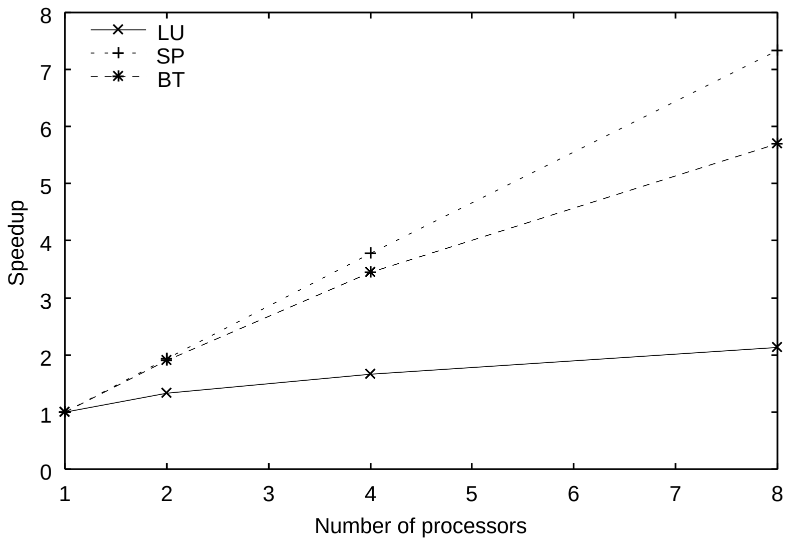

Fig. 4. Speedup of NPB2.3-serial (Class B) on the SR8000

The results for LU, SP and BT are shown in the following. Tables 3 and 4 show the results of the performance of the NAS Parallel Benchmarks 2.3-serial version. The first column of tables indicate the benchmarks used. The next eight columns contain the elapsed time in seconds and the performance in MFLOPS. The performance of SP (Class B) is over 2100 MFLOPS, namely about 18% of the peak performance of a single node of the Hitachi SR8000 compact C model, as shown in Table 4.

Figures 3 and 4 show the speedup of the NAS Parallel Benchmarks relative to the one-processor execution time on a single node of the Hitachi SR8000. The figures show that the speedup on 8 processors is about 5.7 to 7.3 for two of the three benchmarks, LU being the exception. The reason why LU is slow to speed up is that it contains a wavefront-style loop; every loop in the loop nest has dependence across loop iterations, and the loop in this case is not parallelized. Although this loop cannot be parallelized easily in OpenMP, the automatic parallelization of Hitachi's compiler can parallelize such a wave front style loop [2].

6 Concluding Remarks

In this paper, we evaluated the SPEC OMP2001 Benchmarks and three benchmarks (LU, SP and BT) of the NAS Parallel Benchmarks on a single node of the Hitachi SR8000.

For the SPEC OMP2001 Benchmarks evaluated, we observed about 4.3 to 7.7 times speedup on 8 processors. On the other hand, for the NAS Parallel Benchmarks evaluated, we observed about 5.7 to 7.3 times speedup, except for LU, 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 second-level 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. Nishitani, Y., Negishi, K., Ohta, H., Nunohiro, E.: Implementation and evaluation of OpenMP for Hitachi SR8000. In: Proc. Third International Symposium on High Performance Computing (ISHPC 2000). Volume 1940 of Lecture Notes in Computer Science., Springer-Verlag (2000) 391–402
3. Nakazawa, K., Nakamura, H., Boku, T., Nakata, I., Yamashita, Y.: CP-PACS: A massively parallel processor at the University of Tsukuba. Parallel Computing **25** (1999) 1635–1661
4. Tamaki, Y., Sukegawa, N., Ito, M., Tanaka, Y., Fukagawa, M., Sumimoto, T., Ioki, N.: Node architecture and performance evaluation of the Hitachi Technical Server SR8000. In: Proc. 12th International Conference on Parallel and Distributed Computing Systems. (1999) 487–493
5. Shimada, K., Kawashimo, T., Hanawa, M., Yamagata, R., Kamada, E.: A superscalar RISC processor with 160 FPRs for large scale scientific processing. In: Proc. International Conference on Computer Design (ICCD'99). (1999) 279–280
6. Nishiyama, H., Motokawa, K., Kyushima, I., Kikuchi, S.: Pseudo-vectorizing compiler for the SR8000. In: Proc. 6th International Euro-Par Conference (Euro-Par 2000). Volume 1900 of Lecture Notes in Computer Science., Springer-Verlag (2000) 1023–1028
7. Brehm, M., Bader, R., Heller, H., Ebner, R.: Pseudovectorization, smp, and message passing on the Hitachi SR8000-F1. In: Proc. 6th International Euro-Par Conference (Euro-Par 2000). Volume 1900 of Lecture Notes in Computer Science., Springer-Verlag (2000) 1351–1361

8. Standard Performance Evaluation Corporation (SPEC): SPEComp 2001. (http://www.spec.org/hpg/omp)
9. Bailey, D., Barton, J., Lasinski, T., Simon, H.: The NAS parallel benchmarks. NAS Technical Report RNR-91-002, NASA Ames Research Center, Moffett Field, CA (1991)
10. Aslot, V., Domeika, M., Eigenmann, R., Gaertner, G., Jones, W.B., Parady, B.: SPEComp: A new benchmark suite for measuring parallel computer performance. In: Proc. International Workshop on OpenMP Applications and Tools (WOMPAT 2001). Volume 2104 of Lecture Notes in Computer Science., Springer-Verlag (2001) 1–10
11. Iwashita, H., Yamanaka, E., Sueyasu, N., van Waveren, M., Miura, K.: The SPEC OMP2001 benchmark on the Fujitsu PRIMEPOWER system. In: Proc. Third European Workshop on OpenMP (EWOMP 2001). (2001)
12. Aslot, V. Eigenmann, R.: Performance characteristics of the SPEC OMP2001 benchmarks. In: Proc. Third European Workshop on OpenMP (EWOMP 2001). (2001)
13. Jin, H., Frumkin, M., Yan, J.: The OpenMP implementation of NAS parallel benchmarks and its performance. NAS Technical Report NAS-99-011, NASA Ames Research Center, Moffett Field, CA (1999)
14. Bailey, D., Harris, T., Saphir, W., Van der Wijngaart, R., Woo, A., Yarrow, M.: The NAS parallel benchmarks 2.0. NAS Technical Report NAS-95-020, NASA Ames Research Center, Moffett Field, CA (1995)