OpenMP Comparisons and Experiences
Terry Nelson
Computer Sciences Corp.
NASA Ames Research Center
CUG 2002
May 21, 2002
OpenMP Comparisons
Table of Contents

• Site Background
• C90 --> SV1ex
• SV1ex --> Origins
• MPI and OpenMP
• Conversion to OpenMP Issues
• Summary
• References
OpenMP Comparisons
Site Background

• C90 (vn) - 16 CPUs, 1 GW
decommissioned January 31, 2002

• SV1ex (bright) - 32CPUs, 4GW
put into service February 1, 2002

Around 30 Origin Systems, from 8 to 1024 CPUs

Mirror, mirror on the wall…
Program memintensive
! A memory intensive program to check timings between vn and bright
  parameter (niter = 10000, nmax = 1000000)
dimension a(nmax), b(nmax)
t0 = second()
call random_number(a)
t1 = second()
do i = 1, niter
  call sub(nmax, a, b)
endo
t2 = second()
print *, ’Time for random_number = ‘, t1 - t0
print *, ’Time for memory copy = ‘, t2 - t1
stop
d
OpenMP Comparisons
C90 --> SV1

subroutine sub (nmax,a,b)
dimension a(nmax), b(nmax)
b = a
return
end

• For a memory - intensive code, the CPU time on SV1ex can be 4 times “slower” than on C90.
• For a computation - intensive code, the CPU time on SV1ex can be 2.5 times “faster” than on C90.
OpenMP Comparisons
C90 --> SV1

• But, the SV1 has a 32 KW data cache.

• When this cache was used, 57% improvement (603 -> 254 sec.).

• Could this lead towards ‘cache orientation’ on Origins, during code conversion?
OpenMP Comparisons
SV1 --> Origins

• SV1ex - vector orientation, flat memory, CPUs not dedicated; optimization via options as compiler parameters

• Origin - shared, distributed memory, CPUs dedicated (MISER, cpusets); optimization via a long series of SGI extensions

Parallel Loop Execution Options

Memory Locality Features

• shmemb (from Cray, data oriented, MPI-2 one-sided)
OpenMP Comparisons
SV1 --> Origins

• Loop Parallelization
• -pfa (from KAI) -> -apo (separate license)
• -LNO: loop nest optimization
• -cray_mp honor autotasking directives (UNICOS, outmoded)
• pcf(Sequent, parallel computing forum, ANSI-X3H5 91-0023-B)
  C$DOACROSS, C$MP_SCHEDTYPE, C$COPYIN
  C$PAR BARRIER, C$PAR CRITICAL SECTION, C$PDO
SGI extensions to PCF - multiprocessing utility routines
  mp_block, mp_barrier, mp_set_numthreads, mp_create
OpenMP Comparisons
SV1 --> Origins

• Data Locality Tools
  • dplace - pre-execution NUMA memory placement tool
  • dlook - tool for showing memory and process placement
  • dprof - memory access profiling tool
  • numa_view - tool for showing NUMA placement info
  • SGI extensions to OpenMP
    • $SGI DISTRIBUTE, $SGI DISTRIBUTE_RESHAPE, ...
  • Data Distribution directives
    • C$DISTRIBUTE, C$DYNAMIC,…
  • Environment Variables
    • _DSM_MIGRATION, _DSM_PLACEMENT…
OpenMP Comparisons
SV1ex --> Origins

- OpenMP testing tricks and gottchas
  - Use loc() to distinguish threads (be careful if MPI involved!)
  - Use sleep() to have time to view processes (man sleep 3C on Cray)
  - Order of control of number of CPUs can be surprising
    - Job control systems (nqs, pbs, lsf) may set defaults or limits
    - On Cray, NCPUS supercedes omp_num_threads
    - On SGI, setenv OMP_NUM_THREADS or call omp_set_num_threads()
- Origins produce a directory for rii_files, used to facilitate data movement among nodes.

- Cray has both mpirun -nt and -np. Origins only have -np.
OpenMP Comparisons
MPI and OpenMP

• Like 2 dogs or 2 cats???

• Multilevel parallelism
• Motivation - better work distribution, load leveling
• Somewhat similar to ‘mlp’, created by Jim Taft at NAS.

• SGI recognized issues with data locality with these programs.
• SGI has worked on an improved data placement scheme.
OpenMP Comparisons
MPI and OpenMP

- SGI mods available in MPT 1.6 (Beta May ‘02)
- MPI processes spread out to allow room for OpenMP threads
- OpenMP threads placed near MPI parent
- Option to roundrobin MPI process’ data segment across the nodes that its threads are using. This has been found to help for higher thread/mpi process counts.
- This model seems to benefit most applications where
  1) working data set does not reside in scache
  2) more than 4 threads/ MPI process
OpenMP Comparisons
Code Conversions

• PCF --> OpenMP
c$doacross nest (i,j) to exploit parallelism across iterations of a perfectly nested loop nest.

• OpenMP (SV1) --> OpenMP (SGI)
!$OMP PARALLEL DO PRIVATE(iam)
!$OMP& FIRSTPRIVATE(sum), LASTPRIVATE(sum)
“prog.f” line 8: Error: FIRSTPRIVATE and LASTPRIVATE on same variable not yet implemented for PARALLEL DO

!$OMP PARALLEL PRIVATE(iam)
!$OMP DO FIRSTPRIVATE(sum), LASTPRIVATE(sum) compiles without problem.
OpenMP Comparisons
Code Conversions

OpenMP(IBM, Linux) --> OpenMP(SGI)

nthreads = 0
!$OMP PARALLEL REDUCTION(+ : nthreads)
    nthreads = 1
    print *, nthreads
    nthreads = nthreads + 1
!$OMP END PARALLEL
    write(*,*) nthreads
end

• What is the purpose of this code?
• Is the line nthreads = 1 legal?
• Error: Illegal reduction operator for reduction variable nthreads
OpenMP Comparisons
Code Conversions

- Conversion of ‘real’ codes
  - Theorists win 200 - 8
  - Long Call Trees
  - He who saves his code shall lose it.
  - SAVE, -static --> threadprivate common blocks.
    - f77/f90 -static_threadprivate
  - OpenMP 2.0 allows threadprivate variables.
  - PRIVATE? Case-by-case.
  - Thread safety (mpio, craylibs)
  - Reduction can mean values vary. Know your allowable precision!
OpenMP Comparisons
Summary

• As system sizes increase, models like MPI/OpenMP are increasingly important.

• As distributed shared memory systems become more prominent, OpenMP grows in significance.

• Conversion of codes to OpenMP focuses most of all on which variables must be scoped private.

• Earlier SGI data distribution directives are still supported and can still enhance performance, because of the Origins’ memory layout.
OpenMP Comparisons

References

- Parallel Programming in OpenMP (Chandra et al)
- www.omp.org is the primary site for OpenMP
- techpubs.sgi.com/library/ site for SGI documentation
- www.nas.nasa.gov for NAS activities, systems and system documentation