CSC 591C - Cluster Computing Project Home page

Jaydeep Marathe (jpmarath@unity.ncsu.edu)

HW6-Final Report

Address Trace Generation for OpenMP programs using Dynamic Instrumentation

Tasks Completed (Overall)

1. Finalizing Benchmark/Test Suite

- Need good benchmarks to test tracing capability
- Looked at SPEC OMP and NAS NPB-3.0 set of benchmarks
- Categorized OpenMP directives by their type and frequency, for each NAS NPB-3.0 benchmark
- Determined the NAS NPB.30 tests to be better for testing, compared to SPEC-OMP, since they are simpler and have fewer OpenMP directives.

2. Building the Framework

- Started from uniprocessor tracing framework –METRIC [1]
- Analyzed trace instrumentation for thread-safety, modified instrumentation to be thread-safe (with pthread mutex lock/unlock calls)
- Updated tool-chain to better support target load/store instructions (support for new instructions).
- Sorting component to sort huge files – system sort breaks down after 5 Million lines of text.
- Region component to delineate barrier regions in trace, using OpenMP function calls tracing for the IBM xlc/xlf compilers.

3. Initial Testing & Evaluation

- Initial test – benchmark MG from the NAS NPB suite.
- 430 million memory accesses logged – 4 active OpenMP threads.
- Total time required = 180 minutes.
- Total disk space required = 4.3 GB (decompressed trace), 16 MB (compressed trace)

4. Bottleneck Analysis & Redesign

- Total Time and Space required for tracing were deemed unacceptable.
- Analyzed critical paths in the instrumentation framework
- Redesigned framework to optimize these paths.
- New design removes requirement for “sync” stream completely, does region detection online.
- New design does away with mutex locks , thread library now fully multithreaded (threads synchronize only at barrier points, as in the uninstrumented executable) .
- Aim is to bring down instrumentation overhead by an order of magnitude.

5. **OpenMP directives support**

- Currently support “#pragma omp”, “#pragma omp parallel for/do”, “#pragma omp for/do…wait” directives.
- Support could be easily extended to “#pragma omp barrier”, “#pragma omp workshare”.
- Time-sensitive constructs like “omp critical”, “omp master”, “omp single” , “omp flush” will require more work, as their effects are susceptible to perturbation from the instrumentation.
- Most difficult to support “nowait” directive, no satisfactory solution currently due to restrictions on instrumentation (cannot instrument the shared library which actually implemens the OpenMP instrumentation on AIX, due to DynInst instrumentation restrictions. So, we have to manage with instrumenting the calls to the API functions, which are in user code (i.e. the 0x1 segment of memory)).

Tasks Planned

1. **Supporting additional OpenMP directives**

- Support all clauses and directives listed in (5) above, except for the “nowait” directive.

2. **Completing traces for NAS NPB suite**

- Get memory traces for the 8 NAS NPB benchmarks.

References

1. **METRIC**: a framework for extracting uni-processor address traces. http://moss.csc.ncsu.edu/~mueller/ftp/pub/mueller/papers/cgo03.ps.gz

Project Web Page

http://www4.ncsu.edu/~jpmarath/index.html