SYNCHRONIZING THE REPLAY ENGINE WITH THE RECORD ENGINE FOR X86 ARCHITECTURE

By Apoorva Kulkarni(askulkar@ncsu.edu) And Vivek Thakkar(vthakka@ncsu.edu)

PROBLEM DESCRIPTION

An analysis tool has been developed which characterizes the communication behavior of large scale massively parallel applications which use MPI to communicate amongst parallel tasks. This tool has essentially two components – record engine and replay engine. Record engine efficiently produces the communication traces in compressed file(s) and replay engine replays the execution and generates statistical data for analysis purpose. The current record engine has some extensions done over a period of time. However, the corresponding changes have not been done to the replay engine. So, our task is to identify the changes that are needed and to make those changes and do the benchmarking with small and large benchmarks.

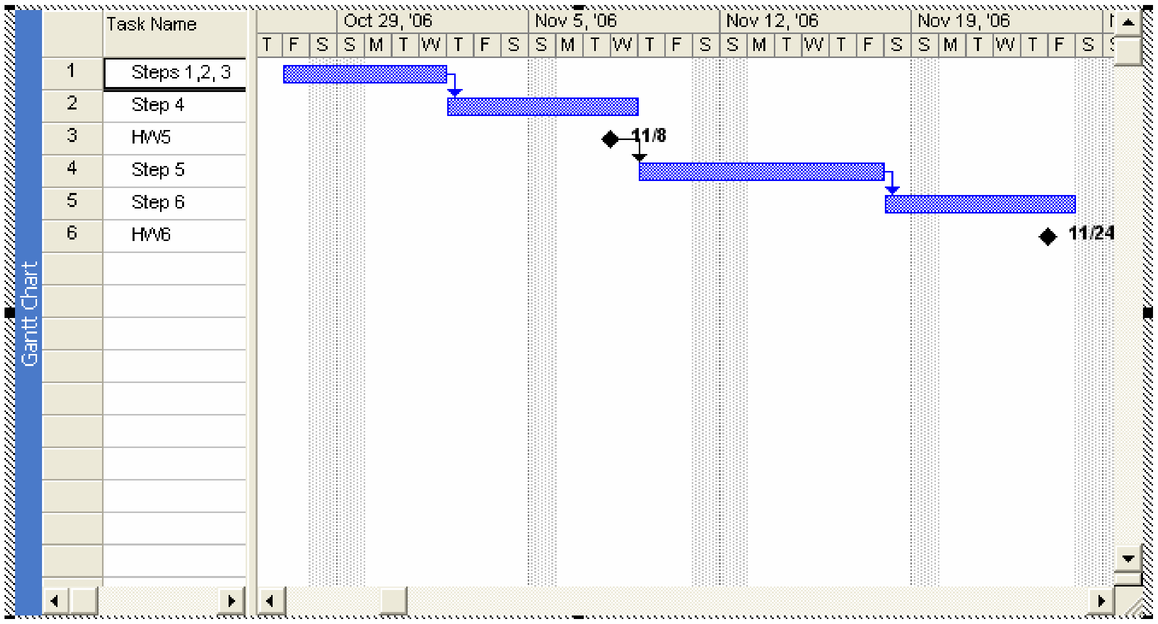
ACTION POINTS

1. Take the source code for old record engine and old replay engine (given in src.tar.gz).
2. Compile record and replay. Make changes in config/Makefile.config, libsrc/stack_sig.h and libsrc/util.c by comparing with the same files given in the new record. Make other changes if required.
3. Try to run sample programs given in “tests” subdirectory. First run by linking them with libumpire.a which is generated as a result of compiling the record engine. Do the initial runs on 1 processor. The record engine should produce the desired trace file(s). Now run the replay engine (linked with mpiP) giving it the path of the trace files Verify the outputs against outputs produced by mpiP. Test 2-3 sample programs. Do the runs on 2, 3 and 4 processors. Perform testing on at least one large benchmark like NAS.
4. Completion of step 3 ensures that old record and replay are synchronously working. Now, make this old replay file as the base file on which the extensions need to be merged. Doing a diff on the libsrc sub directories of the old record and new record shows some minor differences in the following files: Makefile, mpi-spec.umpi.extract, rsd_queue.h, umpi_internal.c, umpi_internal.h,,transfer.h, umpi_mpi.c, umpi_mpi_lookup.c. The action point would be to understand the changes and their motivation. Also an intermediate report (HW5) has to be submitted.
5. Establish a common understanding to identify the changes that may be needed in the replay engine corresponding to the changes done in the record engine. Make those changes on the base replay file (refer 4) to get the new replay engine in sync with the new record engine. They are anticipated to be very few (nonetheless very essential).
6. Compile the code and run small benchmarks as done in step 3. Run the tests on large benchmarks like NAS,ASCI, purple etc. Use different DEBUG switches

(ref. README of record engine) to ensure the correctness of intermediate and final outputs. While benchmarking try to find if some changes can be made in the algorithm to ensure better compression.

PROJECT PLANNING

We believe that the major portion of this project involves understanding of the code and testing & debugging. Minor efforts in changing the code and compilation may also be needed. We think that the only task that can be divided amongst the team members is testing and debugging since both the members are required to know about every other module. Hence the division of work will be performed dynamically as the project proceeds. The planning chart can be shown as below:



	Task Name
1	Steps 1,2, 3
2	Step 4
3	HW5
4	Step 5
5	Step 6
6	HW6

REFERENCES

[1] Class Slides and Paper by M. Noeth, F. Mueller, M. Schulz, B. de Supinski, Scalable Compression and Replay of Communication Traces in Massively Parallel Envrionments, submitted to IPDPS 2007
[2] Dynamic Software Testing of MPI Applications with **Umpire**
www.llnl.gov/CASC/people/vetter/pubs/sc00-umpire-vetter.pdf
[3] ASCI purple benchmark
http://www.llnl.gov/asci/purple/benchmarks/limited/code_list.html
[4] NAS parallel benchmark
http://www.nas.nasa.gov/Resources/Software/npb.html

LINK TO OUR WEBPAGE

http://www4.ncsu.edu/~askulkar/mpitrace.html