Performance Test Harness (PTH)
Jim Coyle – ISU – 4/17/09

- Design Goals
- Commands
- Restart
- Packaging/Running a Test Suite
PTH Design Goals

- Check for performance regression
- Provide consistency in running tests
- Ease of performance comparison
- Ease of reporting
- Ease of use
- Extensibility
Check for Regression

- PTH can run the same set of tests under multiple configurations (modules, flags...)
- Compare timings for each test
- Identify test that are slower (within some tolerance) in the newer configuration compared with the standard configuration
Consistency in Running Tests

- Configuration files provide for consistent
  - environment for all runs: CONFIG
  - additional environment for each separate run: CONFIG.0 CONFIG.1 ...

- Special files allowed for compilation and execution
  - `$(Makefile)` and `$(APRUN)` by default
  - `compile_script` and/or `exec_script` if present
  - `$(testname).compile_script` and/or `$(testname).exec_script` if present
Ease of Comparison

- Two-way text comparisons are printed to STDOUT after each set of runs completes.
- At the end of all runs, an HTML file is created. This shows a table of all timings, with slower timings in red.
- The HTML table is saved but either comparison can also be re-created later.
Ease of Reporting

- Information on machine, OS, date kept for reporting
- Each CONFIG file reported in HTML output
- Slower tests flagged with \textless \textless SLOWER in pth\_compare2
- Slower tests flagged as \textcolor{red}{RED} in HTML output
Ease of Use

• Packages are directories with
  – tests,
  – machine independent files (e.g. exec_script, Makefile)
  – CONFIG files in a machine_type directory
• Once set up, just issue:
  pth_run directory_name machine_type
  e.g. pth_run /lus/nid00008/n4734/tst XT
Extensibility

• Adding a new configuration to be tested just requires a new CONFIG.# file, with likely just a single line like:
  – LOAD_MODULES="Base-opt/PrgEnv-pgi xt-mpt/3.1.2.2"

• Test added by copying program in and modifying a single variable TESTNAMES

• Can add another # of processors by adding to a single variable NUMP
Extensibility (cont.)

• Can add a new size test to run by adding to a single variable SIZES (e.g. SIZES="B CD" in NAS Parallel Benchmark)
• Can compare performance where increases or decreases are expected: e.g. using quad cores
  – APRUN="aprun –N 1 –n " in CONFIG.0
  – APRUN="aprun –N 4 –n " in CONFIG.1
Commands

• `pth_run packagename(directory) XT`
  - Complete run, records date, machine, OS, CONFIG and module list for each CONFIG run.
  - Runs `pth_compare2` and `pth_compare` automatically
  - Output files `out0`, `out1`, `out2` ... one for each `CONFIG[0-9]`
pth_compare2, pth_compare

- pth_compare2 out0 out1
  - Produces a text table with
    - Testname  # of proc  size  time0  time1  ratio
    - Marks with <------- SLOWER if ratio < .95
- pth_compare out0 [out1 [out2] ..]
  - Multi-way compare using out0 as standard
  - Creates HTML table with:
    - Testname # of proc  size  time0  time1 ...
    - Color red used when time[1-9] > 1.05* time0
History

- Results are kept in a directory named HISTORY/date_name e.g.
  2009_Apr_08_16:37:00
Restart

Performance tests take a long time, so many tests may take a very long time. PTH uses a file RESUME_FLAG which it creates at the start of a run, and removes at the end of a run. If pth_run is invoked on a directory where this file exists, it skips all tests already in the out[0-9] files, and starts appending at the first new test.
Packaging a test suite

- Check that test compiles/runs outside of PTH
- Create a package directory
- Copy test, input files, and any machine indep. Makefile or build scripts into this directory.
- Modify build/run scripts as needed to inherit environment variables like:
  - $testname for the name of the test
  - $p for the number of processors to use
Packaging a Test Suite (cont.)

• Create a machine dependent directory within the package directory (e.g. XT)

• In the machine dep. directory:
  – Create CONFIG file which contains overall variables like:
    – APRUN="aprun -n "
    – USE_MODULES="yes"
Packaging a test suite (cont.)

- Create CONFIG.0 CONFIG.1 …, for each test configuration (CONFIG always applied first)
  - LOAD_MODULES="PrgEnv.pgi BaseModules xt-mpt/3.1.1"
  - HEADER="PGI/mpt3.1.1"
- If using a Makefile named Makefile.x, in CONFIG set MAKEFILE="Makefile.x"
Packaging a test suite (cont.)

- If a build script is used, name it compile_script
- If a test needs its own build script, name it testname.compile_script
- Execution should print to STDOUT only a line
  - testname size np time
- Can create or modify test to do this, or
- Use exec_script to run/filter the output or use time.benchmark in exec_script to extract realtime from time ./testname
Using PTH

- pth_run directoryname XT
- firefox compare.html
- Reverse order:
  - pth_compare out1 out0 out3 > 103.html
  - firefox 103.html
- Two way text compare:
  - Pth_compare2 out0 out2
  - Output is like:
  - testname size np time0 time1 ratio ← SLOWER