SAND2007-7475C

Testing LP/MIP Solvers with EXACT

William E. Hart

Cynthia A. Phillips

Jean-Paul Watson

wehart@sandia.gov

Sandia National Laboratories

Sandia is a multiprogram laboratory operated by Sandia Corporation, a Lockheed Martin Company,
for the United States Department of Energy's National Nuclear Security Administration
under contract DE-AC04-94AL85000.

Overview

GOAL: Develop Testing Framework for LP/MIP Solvers

Idea: leverage the EXACT software testing framework

- Structured testing scripts
- Automate testing of solver status
- Summarize output for statistical analysis in R

EXACT Motivation

GOAL: Provide a software framework for defining and analyzing computational experiments

- Managing computational experiments
  - Systematic control is needed for large-scale experimentation
  - Design of experiments to limit the cost of experimentation
  - Archiving experimental results in a standard manner
  - Integration of statistical analysis capabilities

- Applications
  - Experimental evaluation of heuristics
  - Comparisons between algorithms
  - Robust (user) parameter settings (over many problem domains)

EXACT Motivation (Continued)

- Software testing
  - Automation of tests
  - Flexible notion of what a “test” means
  - Integration with diagnostic tools (e.g. valgrind, lcov)
  - Distributed test management and test summary

Observation: testing of large complex software begins to look like a computational experiment

Example: integer programming solver

- Lots of algorithmic parameters
- Lots of hard test problems
- Costly tests

Related Work

- ExpLab
  - Interactive scripts for setting up and performing computational experiments, including tools to archive data
- Research Assistant
  - Strong focus on archiving of data/environment/software to ensure reproducibility (in Java)
- Condor
  - Distributed execution framework
- Software Testing Frameworks
  - There are many of these…
  - Focus: execution of codes and evaluation of final "result"

EXACT’s Niche

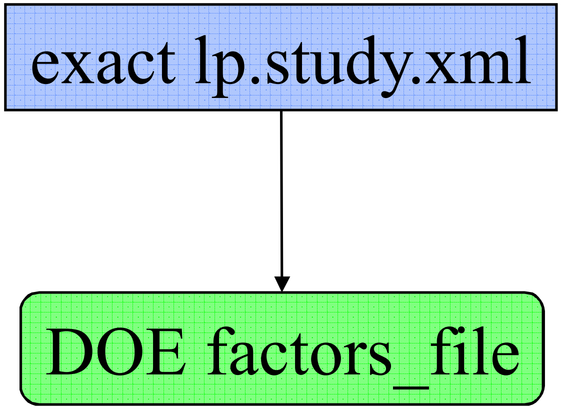
Strengths:

- Integration of DOE tools
- Experiment automation
- Self-contained, portable tool
- Very generic application interface
- Support for generic “Analysis” modules
- Simple specification of parallel tests

Domain of Application:

- Experiments to test theoretical results
- “Horse Race” experiments
- Benchmarking
- Software testing

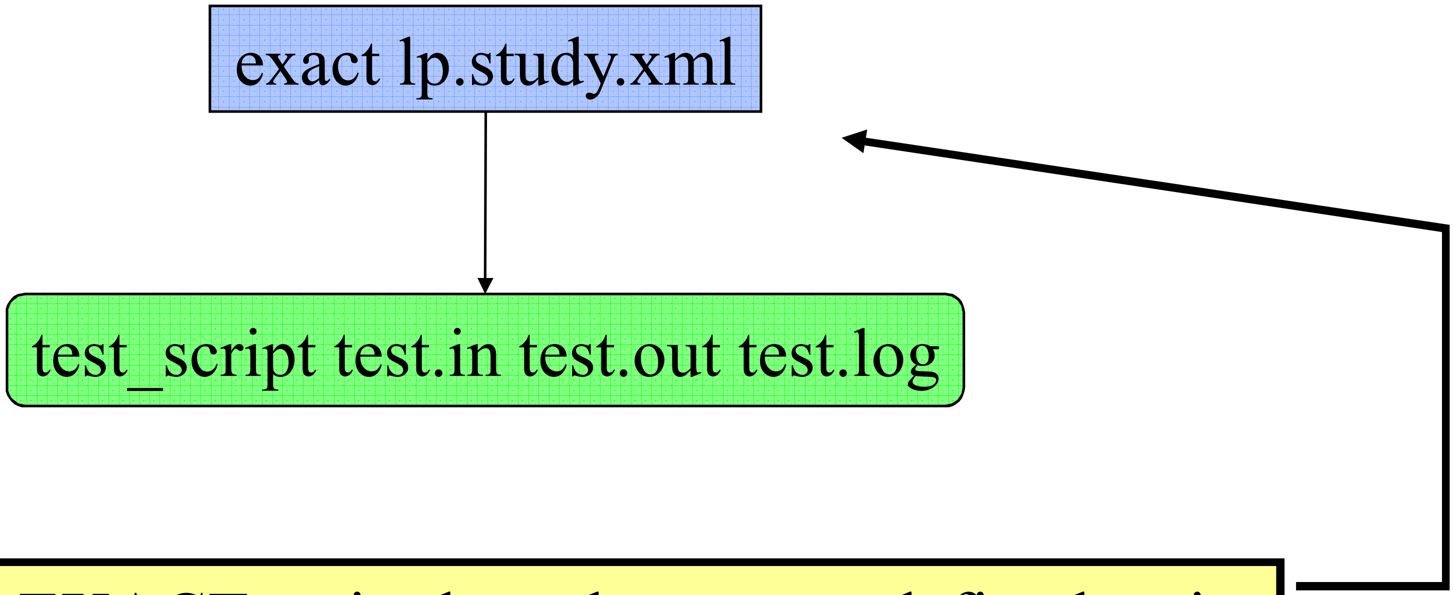
EXACT Data/Execution (1)



The EXACT script can generate an experimental design with an external code. By default, EXACT uses a full factorial design.

This process generates a set of experimental treatments that will be executed in this experiment.

EXACT Data/Execution (2)



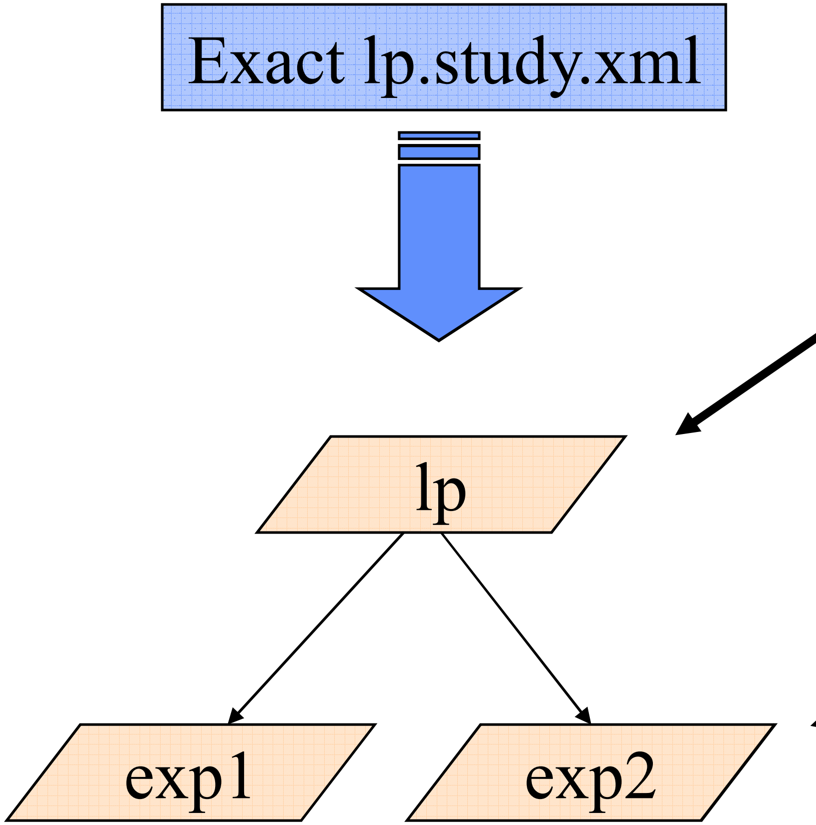
```
exact lp.study.xml
```

```
test_script test.in test.out test.log
```

The EXACT script launches a user-defined script to execute each experimental treatment. Measurements are extracted from the *.log file to generate a *.out file.

Repetitions with random number seeds can also be specified

EXACT Data/Execution (3)



Exact lp.study.xml

lp

exp1

exp2

One or more results files can be analyzed to generate a *.analysis.xml file.

Output files are combined to generate a *.results.xml file of experimental measurements.

XML Description

```
<experimental-study name=“example1”>
  <tags>
     <tag> example </tag>
  </tags>

<experiment name=“lp”>
  <factors>
     <factor name=“lp-type”>
        <level> primal </level>
        <level> dual      </level>
     </factor>
  </factors>
```

XML example continued

```
<controls>
  <executable> test_script </executable>
</controls>
</experiment>

<analysis name="lp-type" type="validation">
  <data experiment = ""/>
  <options> _measurement = FinalValue
            _value        = 0.75
  </options>
</analysis>

</experimental-study>
```

EXACT Input File

```
_exact_debug  0
_experiment_name   example1.lp
_test_name  3
_num_trials  1

Seed $PSEUDORANDOM_SEED

_factor_1_name  lp-type
_factor_1_level   level_1
_factor_1_value  primal

_factor_2_name  lp-solver
_factor_2_level   level_2
_factor_2_value  soplex
```

EXACT Measurement File

```
“User Time” numeric/double 110.0
“FinalValue” numeric/double 0.0001231
“Termination Condition” text/string “Optimal Solution”
exit_status numeric/integer 0
```

XML Specification with Experimental Options

```
<factors>
  <factor name="search">
    <level> </level>
    <level>initialDive=true</level>
    <level>initialDive=true integralityDive=true</level>
  </factor>
  <factor name="problem">
    <level>_data=bm23 _optimum=34 _opttol=1e-8</level>
    <level>_data=p0033 _optimum=3089 _opttol=1e-6</level>
  </factor>
</factors>
```

EXAMPLE: MIP Performance Comparison

This required a relatively simple application of EXACT

- XML test file: mip-miplib2003.study.xml
- EXACT testing script: mip_test
- MIP solver driver: mip
- XML analysis file: mip-miplib2003-analysis.study.xml

mip-miplib2003.study.xml

```
<experimental-study name="mip-miplib2003">
  <tags>
    <tag>nightly</tag>
  </tags>

  <experiment>
    <factors>
      <factor name="test" filename="../studies/mip_miplib2003"/>
      <factor>
        <level>timelimit=3600 solver=$solver _datadir=$miplib2003dir</level>
      </factor>
    </factors>
    <controls>
       <executable timelimit="3660">mip_test --mip</executable>
    </controls>
  </experiment>

</experimental-study>
```

Scripts

mip_test: this is not a simple script

- Can use EXACT Python routines to parse/write the EXACT files
- Need to parse the output of the command that this script executes
- In this case, mip_test executes mip, which makes this easy

mip: a Python script to run lots of different LP/MIP solvers

- Uses a simple command-line interface that exposes the controls that are relevant to these experiments
- Writes out solver status in a consistent manner

mip script options

```
 usage: mip [options] <mps-file>

options:
  -h, --help              show this help message and exit
  --lp                    Use linear programming solver
  --mip                   Use MILP solver
  --solver=SOLVER         This option specifies the type of solver that is
  used
                          to solve the MIP or LP.  The following solver types
                          are currently supported:
                          ..cbc ..clp ..cplex ..glpk ..lpsolve ..minto
  ..pico-
                          clp ..pico-cplex ..pico-glpk ..pico-soplex
  ..picoOLD-
                          clp ..picoOLD-cplex ..picoOLD-glpk ..picoOLD-soplex
                          ..scip ..symphony The default solver is 'pico'.
  --timelimit=TIMELIMIT
                          Limit to the number of seconds that the solver is
  run
  --seed=SEED             Specify a seed to derandomize the solver
  -q, --quiet             Turn off solver output
  --first-feasible        Terminate after the first feasible incumbent
  --solver-options=OPTIONS
                          Options passed into the solver
```

mip-miplib2003-analysis.study.xml

```
<experimental-study name="mip-miplib2003">
  <tags>
    <tag>nightly</tag>
  </tags>

  <analysis name="mip-miplib2003" type="table">
    <data experiment="exp1" import="mip-miplib2003.study.xml"/>
    <options>_solver=$solver</options>
    <format type="csv"/>
    <column option="data"/>
    <column measurement="Solver"/>
    <column measurement="exit_status"/>
    <column measurement="Value"/>
    <column measurement="UserTime"/>
    <column measurement="Optimality"/>
    <column measurement="FinalGAP"/>
    <column measurement="FinalLowerBound"/>
    <column measurement="NodesBounded"/>
  </analysis>

</experimental-study>
```

Issues/Challenges

- Need to make sure that all scripts gracefully terminate subprocesses when terminated
  - Example: time-limit interrupts
  - ISSUE: cplex seemed to ignore interrupts generated by EXACT

- Enforcing time limits
  - Needed to develop utility for this
    - UNIX ‘time’ does not support graceful termination
  - ISSUE: solvers did not uniformly die gracefully and generate summary information
    - Solver I/O can get interrupted abruptly
    - Solver can terminate without summarizing status
  - Ultimately decided to rely on solver time-limit controls
  - ISSUE: are solver time-limit controls consistent?
  - ISSUE: solvers do not manage time –limits with the same granularity

Issues/Challenges

- Analysis of Experimental Results
  - Data extracted by EXACT can be tested in various ways
  - ISSUE: need to visualize and interact with data
  - Developed ability to export data in standard formats (e.g. CSV)
  - Developed R scripts to visualize/analyze data

- Large-Scale Experiments
  - Initial experiments used a limited number of test problems.
  - Larger experiments will not be feasible, particularly when allowing runs to continue for an extended period
  - IDEA: leverage EXACT's ability to use fractional factorial experimental designs

More Information About EXACT

Released under gnu lesser public license:

http://software.sandia.gov/Acro/EXACT

Please send questions/comments to me:

Bill Hart
wehart@sandia.gov

Thank You!