# Testing and Tuning a New Solver Version Using Performance Tests

Arne Stolbjerg Drud
ARKI Consulting & Development A/S
Bagsvaerd
Denmark
adrud@arki.dk

# Overview:

- The Basic Types of Test for a new Solver:
  - Absolute Tests (all Solvers must satisfy)
  - Relative Tests (comparing with an old version)
- Tools from the GAMS Performance World
- Analyzing good Options values
  - Standard Continuous Options
  - Options that affect only few models
- Summary and Conclusions

# Preparing a new Solver Version

A Solver consist of a Solver Library plus an Interface to some model provider / modeling system. Before any new Solver can be released we need to run a number of **absolute tests:**

- The solver must never Crash.
- The solver must not go into an infinite loop.
- The solver must always return a well defined status.
- The solver must obey certain limits: iteration limits, resource limits, and function evaluation error limits.
- The primal/dual solution must obey certain sign conventions (set by the modeling system) and a solution labeled Locally Optimal or Locally Infeasible must indeed be Locally Optimal or Locally Infeasible.

# Preparing a new Solver Version (cont)

Before a new Version of an existing Solver is released we will usually run some additional Relative Tests. The objective is to ensure that the solver in some sense is "Better". This concept of a Pareto Optimal solver implies:

- Reliability: Models that could be solved before should still be solvable.
- Efficiency: Models should run more quickly.

Reliable testing involves a large number of models each run with many sets of options, so some automatic generation of test runs and automatic collection an analysis of test results is necessary.

This talk will describe the GAMS Performance World environment used for testing GAMS/CONOPT3 compared to GAMS/CONOPT2.

# Tools in the GAMS Performance World
# The Absolute Tests

**Crbatch.gms:** Creates a job that manages the execution of a large number of models and the accumulation of test information in a so called “Trace File”.

The Trace File has information about the size of each model (number of variables, equations, nonzeros, nonlinearities etc.) and information about the solution status (optimal, infeasible, resource interrupt, no solution, etc.) and solution statistics (iterations, resources, function evaluation errors, objective value etc.)

The Trace File is a comma delimited file, i.e. it has a format that easily can be read by many programs including Excel and GAMS.

**Schulz.gms**: Schulz catches runaway jobs (i.e. jobs with infinite loops or jobs that do not obey the prespecified resource limits) and terminates them. Can be used in environments where job resource limits are not provided by the operating system (e.g. Windows).

# Tools in the GAMS Performance World
# The Absolute Tests (cont)

**Model Collections**: Collections of models of different types (e.g. LINLib, GLOBALLib, MINLPLib, MPECLib) are available for the initial tests.

**GAMS/Examiner:** A new GAMS "Solver" that analyses whether a solution (computed by a "Real" solver) satisfies the standard optimality criteria. The tests includes primal and dual feasibility and complementary slackness.

# Tools in the GAMS Performance World
# The Absolute Tests (cont)

These tools are used in our initial **Absolute tests**:

- The solver must never Crash: *A crash is translated into a special status in trace file. It does not disappear.*
- The solver must not go into an infinite loop. *Schulz will catch these jobs and the trace file will have a special status.*
- The solver must always return a well defined status. *A missing or wrong status is translated into a special status in trace file.*
- The solver must obey certain limits: iteration limits, resource limits, and function evaluation error limits. *The values are in the trace file and can be compared against the limits.*
- The primal/dual solution must obey certain sign conventions (set by the modeling system) and a solution labeled Locally Optimal or Locally Infeasible must indeed be a Locally Optimal or Locally Infeasible. *GAMS/Examiner performs these checks.*

# Tools in the GAMS Performance World
# The Relative Tests

**Square.gms**: Compares all solver status codes (optimal, infeasible, unbounded, interrupt, fail, etc) of one solver with all return outcomes of another solver. The output is an html file

This can be used to compare the **reliability** of a new version of a Solver with the old version.

Cases where the new solver fails and the old did not needs special attention.

Cases where the new solver returns Locally Infeasible and the old solver returned Locally Optimal must be checked with the GAMS/Examiner.

The following tables are results with the content of GLOBALLib as of November 14, 2002 using the Standard CONOPT2 and the Beta-test CONOPT3.

# Tools in the GAMS Performance World
# The Relative Tests (cont)

The tables are html documents:

Result Totals in Number of Models:

| | optimal | feasible | infeasible | unbounded | fail | total CONOPT2 |
|---|---|---|---|---|---|---|
| optimal | 2 | - | - | - | - | 2 |
| feasible | - | 231 | - | - | 3 | 234 |
| infeasible | - | - | - | - | - | - |
| unbounded | - | - | - | 2 | - | 2 |
| fail | - | 15 | - | - | 3 | 18 |
| total CONOPT3 | 2 | 246 | - | 2 | 6 | 256 |

CONOPT2: feas -- CONOPT3: f Back to top

| Modelname | CONOPT2 | CONOPT3 | Ratio (CONOPT2/CONOPT3) | Obj (CONOPT2) | Status (CONOPT3) |
|---|---|---|---|---|---|
| EX8_3_13 | 0.1406 | 0.8711 | --- | -36.08469789 | mstat( 6)<br>sstat( 4) |
| ROCKET200 | 13.8601 | 9.5430 | --- | -1.01283538 | mstat( 6)<br>sstat( 4) |
| ROCKET400 | 37.9199 | 40.9805 | --- | -1.01283584 | mstat( 6)<br>sstat( 4) |

# Tools in the GAMS Performance World
# The Relative Tests (cont)

**Restime.gms**: Compares the solution times for two solvers, e.g. a new and old version of a solver.

The output are two html files. One compares the solution time (and quality) for all models that are solved locally optimal, and one compares the solution time (and quality) for all model.

| | Total | Obj. CONOPT2 better | Obj. same | Obj. CONOPT3 better |
|---|---|---|---|---|
| Solver CONOPT2 much faster : | - | - | - | - |
| Solver CONOPT2 faster : | 23 | 2 | 20 | 1 |
| Solvers perform the same: | 179 | 5 | 169 | 5 |
| Solver CONOPT3 faster : | 23 | 2 | 17 | 4 |
| Solver CONOPT3 much faster : | 8 | - | 8 | - |
| Total models solved optimal/feasible : | **233** | **9** | **214** | **10** |

Solver CONOPT2 faster - Obj same for both solvers:

| Modelname | CONOPT2 | CONOPT3 | Ratio (CONOPT2 / CONOPT3) | Obj CONOPT2 | Obj CONOPT3 |
|---|---|---|---|---|---|
| CAMSHAPE200 | 0.2344 | 0.3203 | 0.732 | -4.27850023E+00 | -4.27850023E+00 |
| CAMSHAPE400 | 0.6641 | 0.9531 | 0.697 | -4.27568848E+00 | -4.27568848E+00 |
| CAMSHAPE800 | 4.5000 | 5.3359 | 0.843 | -4.27427414E+00 | -4.27427414E+00 |
| CHAIN25 | 0.0703 | 0.0820 | 0.857 | 5.07226149E+00 | 5.07226149E+00 |
| CHAIN50 | 0.0703 | 0.0820 | 0.857 | 5.07226149E+00 | 5.07226149E+00 |
| ELEC25 | 0.2734 | 0.3320 | 0.824 | 2.43812760E+02 | 2.43812760E+02 |
| EX8_3_5 | 0.1484 | 0.2578 | 0.576 | -6.91196701E-02 | -6.91196705E-02 |
| GASOIL100 | 1.3203 | 2.1055 | 0.627 | 5.23659407E-03 | 5.23659431E-03 |
| GASOIL200 | 2.8359 | 4.5859 | 0.618 | 5.23659560E-03 | 5.23659587E-03 |
| GASOIL400 | 5.8828 | 10.0742 | 0.584 | 5.23659554E-03 | 5.23659583E-03 |
| GASOIL50 | 0.5391 | 0.9414 | 0.573 | 5.23663972E-03 | 5.23664002E-03 |
| LNTS100 | 0.7000 | 1.3906 | 0.503 | 5.54595401E-01 | 5.54595401E-01 |
| LNTS50 | 0.2400 | 0.3086 | 0.778 | 5.54668765E-01 | 5.54668765E-01 |
| METHANOL100 | 1.3500 | 1.6094 | 0.839 | 9.02229107E-03 | 9.02229043E-03 |
| METHANOL200 | 3.3200 | 4.7656 | 0.697 | 9.02228980E-03 | 9.02228976E-03 |
| METHANOL400 | 7.0900 | 9.1914 | 0.771 | 9.02228985E-03 | 9.02228985E-03 |
| METHANOL50 | 0.6100 | 0.7422 | 0.822 | 9.02228659E-03 | 9.02228684E-03 |
| QP1 | 0.3000 | 0.3594 | 0.835 | 8.09315050E-04 | 8.09315050E-04 |
| ROBOT50 | 0.3499 | 0.4414 | 0.793 | 9.14687850E+00 | 9.14687850E+00 |
| ROCKET50 | 1.0300 | 1.1719 | 0.879 | -1.01281706E+00 | -1.01281711E+00 |

# Tools in the GAMS Performance World
# The Relative Tests (cont)

The New Solver version is Pareto Optimal if there are no models in the groups “Old Solver is much faster than the New Solver” and “Old Solver is faster than the New Solver”.

This is in practice impossible to achieve because of the large variability of solution times.

**Important client models** are included in separate test collections and we analyze the models where the Old solver is “much faster” than the New solver very carefully.

On **General Purpose model** collections we rely more on average behavior and degradation in performance on some models is accepted as long a the number of model with improved performance is much larger.

The performance tools do not take into account the size of the models. We will generally put more emphasis on larger models by having special collections of larger models.

# Selection of Optimal Options using Performance Tools

A solver like CONOPT has several types of "options":

Discrete options for selection of sub-methods:

- Use preprocessor method A, B or C
- Turn SQP or SLP On or Off
- Turn Scaling On or Off.
- Use line search with or without Bending.

Numerical Tolerances:

- Lower and upper bound on Relative pivot tolerance in sparse LU factorization (actual tolerance is determined dynamically based on behavior).
- Lower and upper bound on Feasibility tolerance (actual tolerance is determined dynamically based on behavior).

# Selection of Optimal Options using Performance Tools (cont)

Small integers:

- After NN iterations with a particular behavior (e.g. all line searches end at a bound), switch to sub-method XX (e.q. SLP).
- When close to optimal in a subspace, add up to NN new superbasic variables.

Size limits for switching between sub-methods:

- If number of superbasic variables is above NN then use conjugate gradient method otherwise store and use an estimated Reduced Hessian.

Most Options have an influence on almost all models and it is therefore possible to use standard statistical methods to compare one Option setting against another:

**Square.gms** is used to find reliable option settings. Within these groups, **Restime.gms** or Excel spreadsheets can be used as a basis for selecting between alternatives.

# Selection of Optimal Options using Performance Tools (cont)

Some Options will only influence a small number of models and the statistical analysis must be adjusted.

An example is the Size limit of a Reduced Hessian: If number of superbasic variables is above NN then use a conjugate gradient method otherwise store and use an estimated Reduced Hessian.

| | Total | Obj. CONOPT3_300 better | Obj. same | Obj. CONOPT3_500 better |
|---|---|---|---|---|
| Solver CONOPT3_300 much faster : | 1 | - | 1 | - |
| Solver CONOPT3_300 faster : | 5 | - | 5 | - |
| Solvers perform the same: | 238 | - | 238 | - |
| Solver CONOPT3_500 faster : | 6 | - | 6 | - |
| Solver CONOPT3_500 much faster : | - | - | - | - |
| Total models solved optimal/feasible : | 250 | - | 250 | - |

If NN is going to be selected in the interval 100-1000 then the behavior of models that always have less than 100 superbasic variables and models that always have more than 1000 superbasics will not be changed. If we are comparing 300 against 500, 500 against 700, and 700 against 1000, an even smaller number of models become relevant.

We should only include the models that actually are affected in the analysis. If we do this, the table above is replaced by:

| | Total | Obj. CONOPT3_300 better | Obj. same | Obj. CONOPT3_500 better |
|---|---|---|---|---|
| Solver CONOPT3_300 much faster : | 1 | - | 1 | - |
| Solver CONOPT3_300 faster : | 2 | - | 2 | - |
| Solvers perform the same: | 3 | - | 3 | - |
| Solver CONOPT3_500 faster : | 2 | - | 2 | - |
| Solver CONOPT3_500 much faster : | - | - | - | - |
| Total models solved optimal/feasible : | 8 | - | 8 | - |

The sample is in this case reduced from 250 to 8 models.

The number of models for various comparisons are:

| | 300 | 500 | 700 | 1000 |
|---|---|---|---|---|
| 100 | 15 | 16 | 16 | 17 |
| 300 | | 8 | 8 | 9 |
| 500 | | | 6 | 7 |
| 700 | | | | 6 |

The mean value of the difference in solution time relative to the mean:

| | 300 | 500 | 700 | 1000 |
|---|---|---|---|---|
| 100 | -0.02 | -0.04 | -0.08 | -0.18 |
| 300 | | -0.05 | -0.11 | -0.32 |
| 500 | | | -0.08 | -0.36 |
| 700 | | | | -0.35 |

And the corresponding standard deviations:

| | 300 | 500 | 700 | 1000 |
|---|---|---|---|---|
| 100 | 0.19 | 0.33 | 0.34 | 0.49 |
| 300 | | 0.37 | 0.39 | 0.61 |
| 500 | | | 0.12 | 0.57 |
| 700 | | | | 0.60 |

Even though all mean values are negative (meaning that lower NN seems better) it is not possible using this dataset to make a reliable decision about where to switch algorithm.

Adding 24 medium to large client models we get the following results (after about 12 hours of computing time on a laptop):

| | 300 | 500 | 700 | 1000 |
|---|---|---|---|---|
| 100 | 20 | 22 | 24 | 25 |
| 300 | | 11 | 13 | 14 |
| 500 | | | 9 | 10 |
| 700 | | | | 6 |

The mean value of the difference in solution time relative to the mean:

| | 300 | 500 | 700 | 1000 |
|---|---|---|---|---|
| 100 | +0.05 | +0.04 | -0.05 | -0.12 |
| 300 | | +0.03 | -0.13 | -0.26 |
| 500 | | | -0.22 | -0.40 |
| 700 | | | | -0.35 |

And the corresponding standard deviations:

| | 300 | 500 | 700 | 1000 |
|---|---|---|---|---|
| 100 | 0.32 | 0.39 | 0.44 | 0.54 |
| 300 | | 0.35 | 0.45 | 0.58 |
| 500 | | | 0.40 | 0.56 |
| 700 | | | | 0.60 |

The larger data set is still not sufficient for any serious conclusions although is seems that values above 500 are bad.

# Summary and Conclusions

Making a new solver ready for distribution requires a large amount of testing and many comparisons.

We have shown some tools that reduces the amount of manual intervention in the process of running models, collecting data, and comparing the results.

The size of the test sets can therefore be increased without increasing the time spend on analysis. The result will be more reliable decisions.

Adjustments to the solver based on the data is still done by hand.