Advanced Use of GAMS Solver Links

Michael Bussieck, Steven Dirkse, Stefan Vigerske

GAMS Development

8th January 2013, ICS Conference, Santa Fe

```
Solve william minimizing cost using mip;
```

generates

```
-- Generating MIP model william
-- magic.gms(81) 4 Mb
--   56 rows  46 columns  181 non-zeroes
--   15 discrete-columns
-- Executing SCIP: elapsed 0:00:00.005
...
-- Restarting execution
-- magic.gms(81) 2 Mb
-- Reading solution for model william
```

```
Solve william minimizing cost using mip;
```

generates

```
-- Generating MIP model william
-- magic.gms(81) 4 Mb
--   56 rows  46 columns  181 non-zeroes
--   15 discrete-columns
-- Executing SCIP: elapsed 0:00:00.005
...
-- Restarting execution
-- magic.gms(81) 2 Mb
-- Reading solution for model william
```

- Returned to the user: solving and model status, solve statistics (solve time), objective value, bound on optimal value,
  primal/dual values for variable and equations with infeasibility markers, ...
- During solve, feedback solely via log output
- No interaction during solve

Log

```
-- Executing CPLEX: elapsed 0:00:00.004

IBM ILOG CPLEX   Dec 18, 2012 24.0.1 LEX 37366.37409 LEG x86_64/Linux
...
LP status(3): infeasible
Cplex Time: 0.00sec (det. 0.01 ticks)

Model has been proven infeasible.
```

Listing

```
               S O L V E      S U M M A R Y

     MODEL    transport              OBJECTIVE  z
     TYPE     LP                     DIRECTION  MINIMIZE
     SOLVER   CPLEX                  FROM LINE  74

**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      4 Infeasible
**** OBJECTIVE VALUE             130.0000
```

Analyzing Infeasible Models: Summary

- LP/NLP solvers usually compute minimal infeasible points
- check `INFEAS` markers in listing file

Analyzing Infeasible Models: Summary

- LP/NLP solvers usually compute minimal infeasible points
- check `INFEAS` markers in listing file

`feasopt` option:

- allows to "price infeasibility", i.e., minimize infeas. w.r.t a certain norm
- also available for MIPs
- available for GAMS/CPLEX and GAMS/Gurobi
- see `feasopt1` in GAMS model library

Analyzing Infeasible Models: Summary

- LP/NLP solvers usually compute minimal infeasible points
- check `INFEAS` markers in listing file

`feasopt` option:

- allows to "price infeasibility", i.e., minimize infeas. w.r.t a certain norm
- also available for MIPs
- available for GAMS/CPLEX and GAMS/Gurobi
- see `feasopt1` in GAMS model library

EMP `adjustequ` option:

- automatic reformulation of constraints as soft constraint
- works also with nonlinear models

Running `gams sp98ir.gms mip=scip optfile=1` with

options file `scip.opt`

```
gams/solvetrace/file = "SCIP.miptrace"
gams/solvetrace/nodefreq = 100
gams/solvetrace/timefreq = 1
```

Running `gams sp98ir.gms mip=scip optfile=1` with

options file scip.opt

```
gams/solvetrace/file = "SCIP.miptrace"
gams/solvetrace/nodefreq = 100
gams/solvetrace/timefreq = 1
```

generates during solve

solve trace file SCIP.miptrace

```
* solvetrace file SCIP.miptrace: ID = SCIP 3.0.1
* fields are lineNum, seriesID, node, seconds, bestFound, bestBound
1, S, 1, 0, 260614197.6, 216717059.8
2, T, 3, 1.12054, 260614197.6, 217028062.2
...
63, E, 2550, 38.3884, 220249516.8, 217928729.7
* solvetrace file closed
```

- common format among all solvers that support this option
- available with Bonmin, CBC, CPLEX, Couenne, GloMIQO, Gurobi, SBB, SCIP, Xpress

Generate GAMS trace files (not to confuse with “solve trace files” from previous slide):

```
gams <model> mip=<solver> trace=<solver>.trc traceopt=3
  reslim=1800 optcr=0 pf4=0 threads=1
```

GAMS trace file <solver>.trc

```
* Trace Record Definition
* GamsSolve
* InputFileName,ModelType,SolverName,OptionFile,Direction,NumberOfEquations,
* NumberOfVariables,NumberOfDiscreteVariables,NumberOfNonZeros,
* NumberOfNonlinearNonZeros,ModelStatus,SolverStatus,ObjectiveValue,
* ObjectiveValueEstimate,SolverTime,ETSolver,NumberOfIterations,NumberOfNodes
30n20b8,MIP,SCIP,1,0,577,18381,11098,109709,0,1,1,302,302,186.8,189.833,464659,466
acc-tight5,MIP,SCIP,1,0,3053,1340,1339,16136,0,1,1,0,0,366.28,367.651,1788064,1971
aflow40b,MIP,SCIP,1,0,1443,2729,1364,8148,0,1,1,1168,1168,1411.99,1425.472,5232401,3
```

Benchmarking with GAMS trace files

Generate GAMS trace files (not to confuse with "solve trace files" from previous slide):

```
gams <model> mip=<solver> trace=<solver>.trc traceopt=3
  reslim=1800 optcr=0 pf4=0 threads=1
```

GAMS trace file <solver>.trc

```
* Trace Record Definition
* GamsSolve
* InputFileName,ModelType,SolverName,OptionFile,Direction,NumberOfEquations,
* NumberOfVariables,NumberOfDiscreteVariables,NumberOfNonZeros,
* NumberOfNonlinearNonZeros,ModelStatus,SolverStatus,ObjectiveValue,
* ObjectiveValueEstimate,SolverTime,ETSolver,NumberOfIterations,NumberOfNodes
30n20b8,MIP,SCIP,1,0,577,18381,11098,109709,0,1,1,302,302,186.8,189.833,464659,466
acc-tight5,MIP,SCIP,1,0,3053,1340,1339,16136,0,1,1,0,0,366.28,367.651,1788064,1971
aflow40b,MIP,SCIP,1,0,1443,2729,1364,8148,0,1,1,1168,1168,1411.99,1425.472,5232401,3
```

- makes it easy to compare solver runs (checkout GAMS Performance Tools)
- e.g., 3 different solvers on MIPLIB 2010 benchmark set:

	C	G	X
mean solve time	138.4s	130.1s	204.7s

However, in practice, solver should not only finish fast, but also find good primal solutions early.

[1] Rounding and Propagation Heuristics for Mixed Integer Programming, Operations Research Proceedings 2011; ZIB-Report 11-29

More Benchmarking with Solve Trace Files

However, in practice, solver should not only finish fast, but also find good primal solutions early. To measure the latter, Achterberg, Berthold, and Hendel (2012)[1] suggested to compute the primal integral:

$$P(T) := \int_{t=0}^{T} p(t),$$

$$\text{where} \quad p(t) = \begin{cases} 1, & \text{if } pb(t) = \infty \text{ or } pb(t) \cdot opt < 0, \\ 0, & \text{if } pb(t) = opt = 0, \\ \frac{|pb(t) - opt|}{\max(|opt|, |pb(t)|)}, & \text{else,} \end{cases}$$

where $pb(t)$ is primal bound at time t, opt is optimal value.

[1] Rounding and Propagation Heuristics for Mixed Integer Programming, Operations Research Proceedings 2011; ZIB-Report 11-29

However, in practice, solver should not only finish fast, but also find good primal solutions early. To measure the latter, Achterberg, Berthold, and Hendel (2012)[1] suggested to compute the primal integral:

$$P(T) := \int_{t=0}^{T} p(t),$$

$$\text{where} \quad p(t) = \begin{cases} 1, & \text{if } pb(t) = \infty \text{ or } pb(t) \cdot opt < 0, \\ 0, & \text{if } pb(t) = opt = 0, \\ \frac{|pb(t)-opt|}{\max(|opt|,|pb(t)|)}, & \text{else,} \end{cases}$$

where $pb(t)$ is primal bound at time t, opt is optimal value.

- ▷ small $P(T) \Rightarrow$ good solutions found early in search
- ▷ can use solve trace files to compute $P(T)$!

[1] Rounding and Propagation Heuristics for Mixed Integer Programming, Operations Research Proceedings 2011; ZIB-Report 11-29

More Benchmarking with Solve Trace Files

However, in practice, solver should not only finish fast, but also find good primal solutions early. To measure the latter, Achterberg, Berthold, and Hendel (2012)[1] suggested to compute the primal integral:

$$P(T) := \int_{t=0}^{T} p(t),$$

$$\text{where} \quad p(t) = \begin{cases} 1, & \text{if } pb(t) = \infty \text{ or } pb(t) \cdot opt < 0, \\ 0, & \text{if } pb(t) = opt = 0, \\ \frac{|pb(t) - opt|}{\max(|opt|, |pb(t)|)}, & \text{else,} \end{cases}$$

where $pb(t)$ is primal bound at time t, opt is optimal value.

- ▷ small $P(T) \Rightarrow$ good solutions found early in search
- ▷ can use solve trace files to compute $P(T)$!

	C	G	X
mean solve time	138.4s	130.1s	204.7s
mean $P(1800; \cdot)/P(1800; C)$	1	1.034	2.099

[1] Rounding and Propagation Heuristics for Mixed Integer Programming, Operations Research Proceedings 2011; ZIB-Report 11-29

Obtaining more than one solution

- Several solver links can write out alternative solutions as GDX files: AlphaECP, BARON, CBC, CPLEX, GloMIQO, Gurobi, SCIP, Xpress
- BARON, CPLEX, and Xpress also offer functionality to explicitly search for alternative solutions
- see GAMS model library model `solnpool`

- Branch-and-cut solvers can benefit from user supplied cutting planes and integer solutions

$\Rightarrow$ callback functions

Branch-Cut-Heuristic Facility

- ▷ Branch-and-cut solvers can benefit from user supplied cutting planes and integer solutions

⇒ callback functions

- ▷ implementation requires knowledge of programming and solver API
- ▷ solver specific

- ▷ Branch-and-cut solvers can benefit from user supplied cutting planes and integer solutions

⇒ callback functions

- ▷ implementation requires knowledge of programming and solver API
- ▷ solver specific

⇒ BCH Facility: pass solver callbacks back into GAMS model space

Branch-Cut-Heuristic Facility

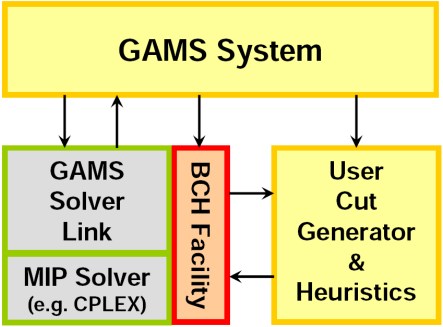
- ▷ Branch-and-cut solvers can benefit from user supplied cutting planes and integer solutions

⇒ callback functions

- ▷ implementation requires knowledge of programming and solver API
- ▷ solver specific

⇒ BCH Facility: pass solver callbacks back into GAMS model space

- ▷ represent cut generator and heuristic in terms of original GAMS formulation
- ▷ independent of specific solver
- ▷ can use any other solvers in GAMS for computations
- ▷ available only for CPLEX and SBB currently

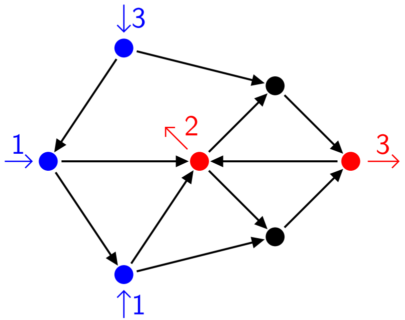


Single-commodity, uncapacitated, fixed-charge network flow problem:

$$\begin{aligned} \min \quad & \sum_{(i,j)\in A} f_{ij} y_{ij} + c_{ij} x_{ij} \\ \text{s.t.} \quad & \sum_{(j,i)\in\delta^-(i)} x_{ij} - \sum_{(i,j)\in\delta^+(i)} x_{ij} = b_i, && i \in V \\ & 0 \le x_{ij} \le M y_{ij}, \quad y_{ij} \in \{0,1\}, && (i,j) \in A \end{aligned}$$

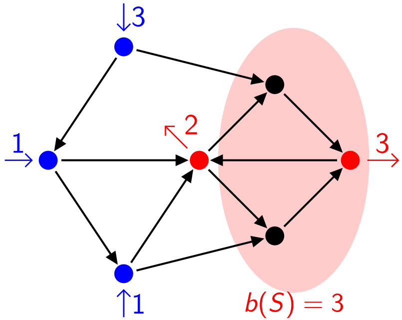
Reference: F. Ortega, L. Wolsey, A branch-and-cut algorithm for the single-commodity, uncapacitated, fixed-charge network flow problem. Networks 41 (2003), No. 3, 143–158

GAMS model library: `bchfcnet`



Dicut: For $S \subset V$ with $b(S) > 0$:

$$\sum_{(i,j)\in\delta^-(S)} y_{ij} \geq 1$$



Dicut: For $S \subset V$ with $b(S) > 0$:

$$\sum_{(i,j)\in\delta^-(S)} y_{ij} \geq 1$$

Separation problem: find a good set S

$$\begin{aligned} \min \quad & \sum_{(i,j)\in A} \bar{y}_{ij} z_j (1 - z_i) \\ \text{s.t.} \quad & \sum_{i\in V} b_i z_i > 0 \\ & z_i \in \{0,1\}, \quad i \in V \end{aligned}$$

$\Rightarrow$ nonconvex quadratic binary program

$\Rightarrow$ let's use GAMS MIQCP solver