FilMINT: A Solver for Mixed Integer Nonlinear Programs

Kumar Abhishek[1] Sven Leyffer[2] Jeff Linderoth[3]

[1]ISE Department
Lehigh University
kua3@lehigh.edu
[2]Mathematics and Computer Science Division
Argonne National Laboratory
leyffer@mcs.anl.gov
[3]ISE Department
Lehigh University
jtl3@lehigh.edu

MIP 2006 Poster Presentation

Problem Definition

Mixed Integer Nonlinear Programming (MINLP) problem:

$$\begin{aligned} Z = \text{Minimize} \quad & f(x, y) \\ \text{subject to } 0 \geq g_j(x, y) \quad & j = 1 \dots m \\ x \in X, y \in Y & \end{aligned} \qquad \text{(MINLP)}$$

- f, g_j are *convex*, differentiable functions.
- x continuous variables, y integer variables.
- Polyhedral set: $X = \{x | x \in \mathbb{R}^n, Dx \leq d, x^L \leq x \leq x^U\}$
- Set of integer points: $Y = \{y | y \in \mathbb{Z}^m, Ay \leq a\}$.

- An NP-Hard Problem.
- A number of interesting Applications. . .

NLP Subproblem Definitions

NLP subproblem for a fixed integers y (say y^k):

$$\begin{aligned} Z_U^k = \min \quad & f(x, y^k) \\ \text{s.t. } \; & g_j(x, y^k) \leq 0 \quad j \in J \\ & x \in X \end{aligned} \qquad \text{(NLP}(y^k))$$

$\Rightarrow$ Solution is x^k.
(l_∞ norm) Feasibility subproblem for fixed y^k:

$$\begin{aligned} \min \quad & u \\ \text{s.t. } \; & g_j(x, y^k) \leq u \quad j \in J \\ & x \in X, u \in \mathbb{R}^1_+ \end{aligned} \qquad \text{(NLPF}(y^k))$$

$\Rightarrow$ Solution is x^k. Other norms possible. . .

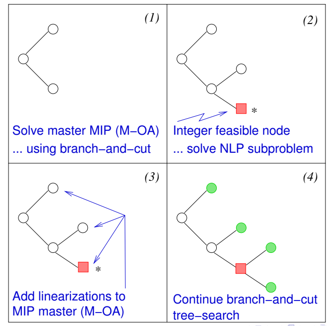
LP/NLP based Branch and Bound Algorithm

MIP Master Problem defined as:

$$\begin{aligned} \min \quad & Z_L^K = \eta \qquad \text{(M-OA)} \\ \text{s.t. } \; & \eta \geq f(x^k, y^k) + \nabla f(x^k, y^k)^T \begin{bmatrix} x - x^k \\ y - y^k \end{bmatrix} \quad k = 1 \dots K \\ & 0 \geq g_j(x^k, y^k) + \nabla g_j(x^k, y^k)^T \begin{bmatrix} x - x^k \\ y - y^k \end{bmatrix} \quad k = 1 \dots K \quad j \in J \\ & x \in X, y \in Y \end{aligned}$$

(Quesada and Grossmann (1992))

LP/NLP based Branch and Bound Algorithm



FilMINT: A linearizations based Solver

- Implementation in a branch and cut framework.
- Use existing software for MIP (MINTO) and NLP (filterSQP).
- Use MIP branch and cut framework's advanced features:
  - Cutting planes.
  - Branching and Node selection rules.
  - Heuristics...
- Adding and managing new linearizations

Flow Chart for FilMINT

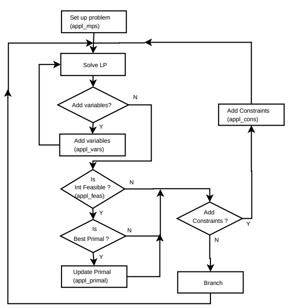


Figure: The application functions

Novel Features:

- Linearizations from NLP for fractional y^k.
  - Don't have to wait for integer feasible solutions.
- Also use Gradient Evaluations instead of solving NLP.
- Strategy for adding these linearizations.
  - Improvement, Depth of tree
- Managing linearizations.
  - Aggregate linearizations at a later stage.
  - Use a cut pool and a mapping with generated points. . .
  - Use MINTO's row management.
- Solving NLP Relaxations every k-th node.
- Heuristics to handle nonconvexity.
  - Penalize constraints, Equality relaxation, local cuts.

Computational Experiments

- Test suite of 400+ problems (250+ convex, 150+ non-convex)
- Sources : GAMS, MacMINLP, IBM-CMU
- Classification: Convex(easy, moderate, hard), and nonconvex.
- Use the Condor Grid running over Beowulf Cluster (120 processors, 1.8 GHz., 2 Gb RAM)
- Time limit: 4 hours.
- Test various features in FilMINT.
- Compare FilMINT with MINLP-BB.

Computational Experiments...

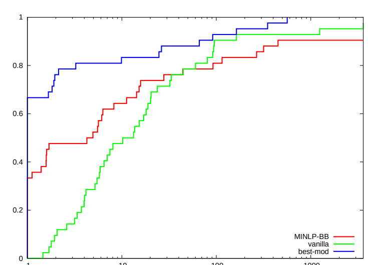


Figure: Performance Profile for Moderate Convex Instances

Computational Experiments...

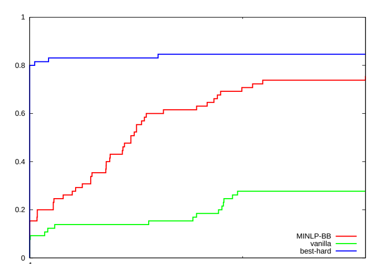


Figure: Performance Profile for Hard Convex Instances (gap to best known solution)

Conclusion

- Introduce new solver, FilMINT, for solving MINLP problems.
- Use existing software for MIP and NLP to create the solver.
- Use MIP framework's advanced features to solve the problem.
- New linearizations added and managed.
- Flexible algorithmic framework.
- FilMINT rules!!

Questions ?? Suggestions ??