SBB: A New Solver for Mixed Integer Nonlinear Programming

Michael R. Bussieck
GAMS Development Corp.

Arne S. Drud
ARKI Consulting & Development A/S

OR2001, Duisburg
Overview

- SBB = Simple Branch & Bound
- Introduction: The MINLP Model
- The B&B Algorithm
- The SBB Design Criteria and Implementation
- Pseudo Cost Variable Selection
- Computational Experiments
- Conclusions and Future Work
The MINLP Model

- **Min or max** $f(x,y)$
  \[ s.t. \quad g(x,y) = b \]
  \[ l \leq x \leq u \text{ continuous} \]
  \[ li \leq y \leq ui \text{ integer} \]

- For convenience we only discuss “Min”

- SBB handles also: SOS1, SOS2, SemiContinuous, and SemiInteger Variables

- Integrality Constraints on $y$ removed: **RMINLP**

- A **Node** is the RMINLP model with tightened bounds on $y$
The B&B Algorithm

Solve RMINLP

Initialize/Store In a Node Table

Is the Node Table empty?

Select a Node, remove from Table, and Solve

Infeasible or Obj worse than Best?

Integer Solution?

Select a fractional discrete Variable. Create/store 2 nodes with Tighter Infeasible Bounds

Unbounded Infeasible Integer

Report and Stop

Yes

Report and Stop

Yes

Store Solution Update Best Remove Nodes With worse Obj

Yes

Report and Stop

Yes
Local/Global Optimization Issues

- Nodes solved with an NLP Local Optimization Algorithm. Potential Problems are:
  - Search Tree can incorrectly be cut off:
    - A node is declared “Infeasible” even though it is only “locally infeasible” and there exist feasible regions.
    - A node is fathomed because the objective for the node is a poor approximation for the global optimum for the node.
  - Problems will occur if the NLP solver fails
- RMINLP model is **convex** in both $x$ and $y$

  Global Optimum
Design Objectives

- Use any existing GAMS NLP solver
  - CONOPT, MINOS, SNOPT
  - PATHNLP, CPLEXSLP
- Handle solver failures
- Help against incorrect infeasibilities
- Report information on non-convexities
Implementation - Concept

- GAMS Process
  - Model Files
  - Solution Files
- Solver Process
  - Revised Model Files
  - RMINLP Solution Files
- SBB Process
  - NLP Solver Process
Efficient SBB Implementation

Model Files

GAMS Process

Solution Files

Solver Process

SBB Process

Bound Revision File

Node File Primal + Dual Solution

NLP Solver Process

RMINLP Solution Files
Solving the RMINLP Submodels

Parent Node with known Optimal Solution

Add One or a few bounds

Child Node with few primal Infeasibilities, Dual Feasible
Some SBB Options

- **failseq solver1.opt1 solver2.opt2**
  - Try solver1 with options defined by opt1. If it fails, try solver2 with options defined by opt2, etc. If all fail, **ignore** the node and continue the search

- **infeasseq level solver1.opt1 solver2.opt2**
  - If a node with depth ≤ level is locally infeasible then try solver1 with options defined by opt1, etc.

- **rootsolver solver.opt**
  - Use a special solver/option pair for the root node

- **subiter/subres max**
  - Avoid that one node uses all resources
F set of *binary* variables with fractional value in optimum solution of RMINLP

|F| > 1: Variable selection problem

- Min/max (integer) infeasibility
- Pseudo cost (PC)
  - Make important decisions early in the B&B tree
  - Measured by change in the objective of both children nodes
  - Reduces tree size
  - Improves the best bound quickly
Definition:

- $obj^*$ objective in parent node
- $x^*$ optimum solution in parent node

Upbranch:  
\[ x_i = 1, c_i = \frac{obj - obj^*}{1 - x_i^*} \]

Downbranch:  
\[ x_i = 0, c_i = \frac{obj - obj^*}{x_i^*} \]

- We don’t want $c_i = \text{big}, c_i = \text{tiny}$
- $i^* = \arg \max_{i \in F} (\min(c_i, c_i^*))$
PC: Initialization and Update

- Initialization
  - Objective difference vs. Number of nodes
  - Non linear ???
    - Ask solver (CONOPT) to produce PC by solving 2|F| NLPs internally (without overhead of I/O, presolve, scaling, …)
    - Terminate early if
      - Have large $c_i$ and $\underline{c}_i$
      - Feasible and objdiff tiny

- Update
  - $n$th update
    - After solving child node with $x_i = 0$
    - $c_i = \left( \frac{c_i + \frac{obj - obj^*}{x_i^*}}{n} \right)$
      - After … $x_i = 1$
      - ...
Computational Experiments

- **MINLP Solver**
  - SBB (Branch & Bound)
  - DICOPT (Outer Approximation)
  - FilterBB (B&B)
  - AlphaECP/mittlp (Extended Cutting Plane)
  - MINOPT/LINGO algorithms

- **MINLP Solver (Global Optimization)**
  - BARON (Branch and Reduce)
  - OQGRG (Local NLP + Global Search)
Tests

- GAMS/DICOPT
- GAMS/SBB
  - Default, Pseudo Cost Variable Selection
- FilterBB (AMPL Interface)
- Test library MINLPLib
  - 139 MINLP models (existing collections + difficult client models)
  - http://www.gamsworld.org/minlp
  - DICOPT/FilterBB do not support SOS1/2 and semi continuous variables (7 models).
- 10 minutes time limit (1GHz PC)
- NLP: CONOPT, MIP: CPLEX
Welcome to the MINLP World!

MINLP World is a forum for discussion and dissemination of information about all aspects of Mixed Integer Nonlinear Programming (MINLP).

MINLP models are models that combine combinatorial aspects with nonlinearities. MINLP models are much more difficult than both Mixed Integer Linear Programming (MIP) and Nonlinear Programming (NLP) models.

MINLP is still a new field, and we cannot yet solve all the problems that naturally fall within this area. It is the purpose of this site to bring people that work with MINLP together. We are interested in practical software (MINLP Solvers), testing, comparison, and quality of solvers (MINLPLib), research in both solution methods and in good model formulations, and in improving the communication between people interested in these topics (Related Links and MINLP list).
## MINLPLib Model Statistics

<table>
<thead>
<tr>
<th>Name</th>
<th>#Eqns</th>
<th>#Vars</th>
<th>#DVars</th>
<th>#NZ</th>
<th>#NNZ</th>
<th>BestInt</th>
<th>at Point</th>
</tr>
</thead>
<tbody>
<tr>
<td>4stufen</td>
<td>99</td>
<td>150</td>
<td>48</td>
<td>319</td>
<td>87</td>
<td>116329.7000</td>
<td>p1</td>
</tr>
<tr>
<td>alan</td>
<td>8</td>
<td>9</td>
<td>4</td>
<td>24</td>
<td>3</td>
<td>2.9250</td>
<td>p1</td>
</tr>
<tr>
<td>batch</td>
<td>74</td>
<td>47</td>
<td>24</td>
<td>191</td>
<td>22</td>
<td>285506.5000</td>
<td>p1</td>
</tr>
<tr>
<td>batchdes</td>
<td>20</td>
<td>20</td>
<td>9</td>
<td>53</td>
<td>10</td>
<td>167427.7000</td>
<td>p1</td>
</tr>
<tr>
<td>beuser</td>
<td>115</td>
<td>158</td>
<td>52</td>
<td>398</td>
<td>159</td>
<td>116348.0000</td>
<td>p1</td>
</tr>
<tr>
<td>cecil.13</td>
<td>899</td>
<td>841</td>
<td>162</td>
<td>2812</td>
<td>360</td>
<td>-115570.3000</td>
<td>p1</td>
</tr>
<tr>
<td>contvar</td>
<td>285</td>
<td>297</td>
<td>87</td>
<td>1281</td>
<td>530</td>
<td>809149.8000</td>
<td>p1</td>
</tr>
<tr>
<td>csced1</td>
<td>23</td>
<td>77</td>
<td>63</td>
<td>174</td>
<td>8</td>
<td>-30639.2600</td>
<td>p1</td>
</tr>
<tr>
<td>csced2</td>
<td>138</td>
<td>401</td>
<td>308</td>
<td>958</td>
<td>58</td>
<td>-166102.0000</td>
<td>p1</td>
</tr>
<tr>
<td>deb10</td>
<td>130</td>
<td>183</td>
<td>11</td>
<td>692</td>
<td>432</td>
<td>209.4278</td>
<td>p1</td>
</tr>
<tr>
<td>deb6</td>
<td>508</td>
<td>476</td>
<td>20</td>
<td>2342</td>
<td>1432</td>
<td>201.7393</td>
<td>p1</td>
</tr>
<tr>
<td>deb7</td>
<td>898</td>
<td>814</td>
<td>10</td>
<td>4116</td>
<td>2816</td>
<td>116.5846</td>
<td>p1</td>
</tr>
<tr>
<td>deb8</td>
<td>898</td>
<td>824</td>
<td>10</td>
<td>4136</td>
<td>2816</td>
<td>116.5846</td>
<td>p1</td>
</tr>
<tr>
<td>deb9</td>
<td>918</td>
<td>814</td>
<td>10</td>
<td>4156</td>
<td>2816</td>
<td>116.5846</td>
<td>p1</td>
</tr>
<tr>
<td>detft1</td>
<td>6206</td>
<td>4408</td>
<td>400</td>
<td>26610</td>
<td>15400</td>
<td>12.7753</td>
<td>p1</td>
</tr>
<tr>
<td>dosemind2</td>
<td>119</td>
<td>166</td>
<td>32</td>
<td>4379</td>
<td>4080</td>
<td>173.9806</td>
<td>p1</td>
</tr>
<tr>
<td>dosemind3</td>
<td>1003</td>
<td>1047</td>
<td>18</td>
<td>24614</td>
<td>22095</td>
<td>1.4626</td>
<td>p1</td>
</tr>
<tr>
<td>elf</td>
<td>39</td>
<td>55</td>
<td>24</td>
<td>178</td>
<td>30</td>
<td>0.1917</td>
<td>p1</td>
</tr>
<tr>
<td>enunplac</td>
<td>190</td>
<td>142</td>
<td>24</td>
<td>511</td>
<td>48</td>
<td>-131863.6000</td>
<td>p1</td>
</tr>
<tr>
<td>enpro48</td>
<td>215</td>
<td>154</td>
<td>92</td>
<td>742</td>
<td>29</td>
<td>187277.3000</td>
<td>p1</td>
</tr>
<tr>
<td>enpro56</td>
<td>192</td>
<td>128</td>
<td>73</td>
<td>651</td>
<td>24</td>
<td>263428.3000</td>
<td>p1</td>
</tr>
<tr>
<td>ex1221</td>
<td>6</td>
<td>6</td>
<td>3</td>
<td>17</td>
<td>2</td>
<td>7.6672</td>
<td>p1</td>
</tr>
<tr>
<td>ex1222</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>9</td>
<td>2</td>
<td>1.0765</td>
<td>p1</td>
</tr>
<tr>
<td>ex1223</td>
<td>14</td>
<td>12</td>
<td>4</td>
<td>40</td>
<td>17</td>
<td>4.5796</td>
<td>p1</td>
</tr>
<tr>
<td>ex1223a</td>
<td>10</td>
<td>8</td>
<td>4</td>
<td>32</td>
<td>9</td>
<td>4.5796</td>
<td>p1</td>
</tr>
<tr>
<td>ex1223b</td>
<td>10</td>
<td>8</td>
<td>4</td>
<td>32</td>
<td>17</td>
<td>4.5796</td>
<td>p1</td>
</tr>
<tr>
<td>ex1224</td>
<td>8</td>
<td>12</td>
<td>8</td>
<td>31</td>
<td>6</td>
<td>-0.9435</td>
<td>p1</td>
</tr>
<tr>
<td>ex1225</td>
<td>11</td>
<td>9</td>
<td>6</td>
<td>27</td>
<td>2</td>
<td>31.0000</td>
<td>p1</td>
</tr>
<tr>
<td>ex1226</td>
<td>6</td>
<td>6</td>
<td>3</td>
<td>15</td>
<td>2</td>
<td>-17.0000</td>
<td>p1</td>
</tr>
<tr>
<td>ex1233</td>
<td>65</td>
<td>53</td>
<td>12</td>
<td>221</td>
<td>28</td>
<td>155010.7000</td>
<td>p1</td>
</tr>
<tr>
<td>ex1243</td>
<td>97</td>
<td>69</td>
<td>16</td>
<td>329</td>
<td>36</td>
<td>83402.5100</td>
<td>p1</td>
</tr>
<tr>
<td>ex1244</td>
<td>130</td>
<td>96</td>
<td>23</td>
<td>469</td>
<td>52</td>
<td>82042.9100</td>
<td>p1</td>
</tr>
<tr>
<td>ex1252</td>
<td>44</td>
<td>40</td>
<td>15</td>
<td>118</td>
<td>36</td>
<td>128393.7000</td>
<td>p1</td>
</tr>
<tr>
<td>ex1252a</td>
<td>35</td>
<td>25</td>
<td>9</td>
<td>94</td>
<td>36</td>
<td>128393.7000</td>
<td>p1</td>
</tr>
</tbody>
</table>
Opt = “optimal”
Int = integer solution
Nis = no int solution
Quality/Speed SBB/DICOPT

- Test on 80 models that were reported “optimal” in both cases.
- All experiments on 1GHz Linux PC
- Total times
  - SBB: 1572s
  - DICOPT: 258s
SBB/SBB-PC Comparison
Quality/Speed SBB-PC/SBB

- Test on 103 models that were reported “optimal” in both cases.
- Total times
  - Faster
    - SBB-PC: 1246s
    - SBB: 2187s
  - Slower
    - SBB-PC: 1874s
    - SBB: 599s
Global Solutions???

- Test on 62 models for which the global optimum is known (Global optimization solver BARON)
- Relative gap:
  \[ rgap = \frac{|obj - obj_{\text{glob}}|}{|obj_{\text{glob}}|} \]
Future Work/Conclusions

- Today MINLP is at the stage where MIP was long time ago 10-15 years ago

- Model Formulations:
  - Recommendations for good formulations and warnings against bad formulations
  - Preprocessing

- Algorithmic/Theoretical Work:
  - Preprocessing, Probing
  - Cut Generation, …

- Serious MINLP modelers/users need more than one MINLP solver (similar to NLP)