The CPLEX Library: MIP Heuristics

Ed Rothberg, ILOG, Inc.

Motivation for Heuristics

Why not wait for branching?

- Produce feasible solutions as quickly as possible
  - Often satisfies user demands
  - Avoid exploring unproductive subtrees
  - Better reduced-cost fixing
- Avoid “tree pollution”
  - Good fixings in a heuristic are often not good branches
- Increase diversity of search
  - Strategies in heuristic may differ from strategies in branching
Two classes

- **Plunging heuristics:**
  - Maintain linear feasibility
  - Try to achieve integer feasibility

- **Local improvement heuristics:**
  - Maintain integer feasibility
  - Try to achieve linear feasibility

**Plunging Heuristic Structure**

- Fix a set of integer infeasible variables
  - Usually by rounding
- Perform bound strengthening to propagate implications
- Solve LP relaxation
- Repeat
**Bound Strengthening**

Propagate new bounds through inequalities

- Given a constraint:
  - \( \sum a_j x_j \leq b \)
  - Split equalities into a pair of inequalities

- Consider a single \( x_k \):
  - \( a_k x_k + \inf ( \sum_{j\neq k} a_j x_j ) \leq \sum a_j x_j \leq b \)
  - \( x_k \leq \frac{(b - \inf ( \sum_{j\neq k} a_j x_j ))}{a_k} \)
    - Assuming \( a_k \geq 0 \)

- Change in variable bound can produce changes in other bounds

---

**Bound Strengthening Example**

- \( x + 2y + 3z \leq 3 \)
  - all variables binary
  - \( x = 1 \)
- \( 3z \leq 3 - \inf (x + 2y) = 3 - 1 = 2 \)
- \( z \leq 2/3 \)
Plunging Details

Important details

• How many variables to fix per round:
  • All of them?
    • Inexpensive; no need to solve LP relaxations
    • But ‘flying blind’ after a few fixings
      – Bound strengthening helps
  • A few?
    • More expensive
    • LP relaxation can guide later choices
      – (variable values, reduced costs, etc.)

• In what order are variables fixed?
  • Variations useful for diversification

Local Improvement Heuristics

High-level structure

• Choose integer values for all integer variables
  • Produces linear infeasibility

• Iterate over integer variables:
  • Does adding/subtracting 1 reduce linear infeasibility?

• Infeasibility metrics:
  • Primary: number of violated constraints
  • Secondary: |b-Ax|
Local Improvement Details

• What initial values to assign to integer variables?
  • Rounded relaxation values
  • 0

• Move acceptance criteria?
  • Greedy

• What to do when local improvement gets stuck?
  • Reverse infeasibility metrics

Continuous variables

• What to do about continuous variables?
  • To what value should they be fixed?
  • What does the neighborhood look like?

• Our approach:
  • Don’t fix them
  • Constraint is satisfied if \( \inf(LHS) \leq RHS \)
General Heuristic Strategies

Apply 9 different variations

- Apply the least expensive heuristics after every round of root cutting planes
- Apply all heuristics before beginning the branch and bound search
- Apply them every 10 nodes in the MIP tree
- Decrease the frequency of a particular heuristic when it is not finding new feasible solutions

Sample CPLEX Output

First 1,000 nodes, default settings

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Node</th>
<th>Left</th>
<th>Objective</th>
<th>IInf</th>
<th>Best Integer</th>
<th>Cuts/</th>
<th>Best Node</th>
<th>ItCnt</th>
<th>Gap</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>0</td>
<td>346.0000</td>
<td>536</td>
<td>346.0000</td>
<td>551</td>
<td>94.47%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>0+</td>
<td>0</td>
<td>6262.0000</td>
<td>0</td>
<td>346.0000</td>
<td>551</td>
<td>91.06%</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>560.0000</td>
<td>490</td>
<td>6262.0000</td>
<td>Cuts:</td>
<td>1131</td>
<td>91.06%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>592</td>
<td>6262.0000</td>
<td>Cuts:</td>
<td>688</td>
<td>91.06%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>256</td>
<td>0</td>
<td>3780.0000</td>
<td>560.0000</td>
<td>3566</td>
<td>85.19%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>296</td>
<td>0</td>
<td>2992.0000</td>
<td>560.0000</td>
<td>3703</td>
<td>81.28%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>296</td>
<td>0</td>
<td>2626.0000</td>
<td>560.0000</td>
<td>3703</td>
<td>78.67%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>368</td>
<td>0</td>
<td>2590.0000</td>
<td>560.0000</td>
<td>4405</td>
<td>78.38%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>628</td>
<td>0</td>
<td>2576.0000</td>
<td>560.0000</td>
<td>6928</td>
<td>78.26%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>638</td>
<td>0</td>
<td>2538.0000</td>
<td>560.0000</td>
<td>6946</td>
<td>77.94%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>656</td>
<td>0</td>
<td>2478.0000</td>
<td>560.0000</td>
<td>7011</td>
<td>77.40%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>656</td>
<td>0</td>
<td>2448.0000</td>
<td>560.0000</td>
<td>7027</td>
<td>76.69%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>645</td>
<td>0</td>
<td>2402.0000</td>
<td>560.0000</td>
<td>7027</td>
<td>76.69%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>639</td>
<td>0</td>
<td>2360.0000</td>
<td>560.0000</td>
<td>7070</td>
<td>76.27%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>*</td>
<td>726</td>
<td>0</td>
<td>2340.0000</td>
<td>560.0000</td>
<td>8134</td>
<td>76.07%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Heuristic Results

Effectiveness

• Feasible solution found for most models before branch and bound begins

• Roughly 10% improvement in time to proven optimality (978 model test set)

• Often find solutions branching does not
Local Search

Powerful optimization framework

• Local search is a very powerful heuristic approach to solving difficult combinatorial optimization problems
• Example local search methods:
  • Simulated annealing
  • Tabu search
  • Genetic algorithms
  • ...

Local Search

Three key ingredients

• Neighborhood:
  • A set of solutions that are in the vicinity of the current solution

• Intensification:
  • A temporary focus on a part of the solution space

• Diversification:
  • A mechanism for changing focus on occasion
Applying Local Search to MIP?

- Neighborhoods:
  - Local search neighborhoods generally based on problem structure
    - Example: Nodes and edges in a graph
  - No high level structural information available in an arbitrary MIP model
  - Given an incumbent $x^*$, can we generate and explore an interesting neighborhood?

Two Recent Proposals

- Local Branching [Fischetti and Lodi, 2002]
  - Add a local branching constraint to MIP model:
    - $|x - x^*| \leq k$
  - Solve a (truncated) sub-MIP

- Relaxation Induced Neighborhood Search (RINS) [Danna, Rothberg, and Le Pape, 2003]
  - Fix all variables that agree in the current relaxation solution and $x^*$
  - Solve a sub-MIP on the variables that differ
**Intensification through sub-MIPs**

**Standard MIP tree**

- First incumbent found
- Sub-MIP
- Later in search

**Local Branching Details**

- Explore vicinity of incumbent

  - Constrain sub-MIP to explore a small neighborhood of incumbent $x^*$
    - $|x - x^*| \leq k$
    - $k$ chosen to be ~20
  - Apply whenever a new incumbent is found
    - Including those found by local branching
  - A succession of improving, neighboring solutions
RINS Details

Explore portion where solutions differ

- Combine desirable properties of two solutions:
  - Incumbent: feasible
  - Relaxation: optimal
- Neighborhood contains both solutions
- Extend promising partial solution

Local branching vs. RINS

<table>
<thead>
<tr>
<th>Local branching</th>
<th>RINS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Explores a neighborhood of the incumbent</td>
<td>Explores a neighborhood of both the incumbent and the continuous relaxation</td>
</tr>
<tr>
<td>Can be called only each time a new incumbent is found</td>
<td>Can be called at each node of the branch-and-cut tree</td>
</tr>
<tr>
<td>Expensive sub-MIP: original model + dense constraint</td>
<td>Sub-MIP on a reduced number of variables</td>
</tr>
<tr>
<td>Not efficient on general integer variables</td>
<td>Can handle any type of variable</td>
</tr>
</tbody>
</table>
Results

"Intermediate" problems

<table>
<thead>
<tr>
<th>Gap</th>
<th>Time (s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.30</td>
<td></td>
</tr>
<tr>
<td>1.25</td>
<td></td>
</tr>
<tr>
<td>1.20</td>
<td></td>
</tr>
<tr>
<td>1.15</td>
<td></td>
</tr>
<tr>
<td>1.10</td>
<td></td>
</tr>
<tr>
<td>1.05</td>
<td></td>
</tr>
<tr>
<td>1.00</td>
<td></td>
</tr>
</tbody>
</table>

Extensions

- Other interesting neighborhoods?
- More efficient ways to explore neighborhoods?