Optimization Algorithms for the Minimum-Cost Satisfiability Problem

Xiao Yu Li
Ph.D. Final Defense

August 12, 2004

Outline

1. The MinCostSat Problem

1. Our Contributions

3. Conclusions
The Satisfiability Problem

Is a Boolean formula in Conjunctive Normal Form satisfiable?

\[(\overline{x_1} \lor x_2 \lor x_3) \land (x_1 \lor \overline{x_2} \lor x_3 \lor x_4) \land (\overline{x_2} \lor \overline{x_3} \lor \overline{x_4})\]

<table>
<thead>
<tr>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>-1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>-1</td>
<td>-1</td>
<td>-1</td>
<td>-1</td>
</tr>
</tbody>
</table>

Solution 1: 
\[x_1 = 1, \, x_2 = 1, \, x_3 = 0, \, x_4 = 1\]

Solution 2: 
\[x_1 = 0, \, x_2 = 0, \, x_3 = 1, \, x_4 = 1\]

The Satisfiability Problem

SAT solvers search for one satisfying solution or prove it unsatisfiable
The MinCostSat Problem

What is the minimum cost solution for a SAT formula?

• Minimize $\sum c_i x_i$ where $x_i \in \{0,1\}$ and $c_i \geq 0$

• Assume unit cost for each variable:

  $(\overline{x}_1 \lor x_2 \lor x_3) \land (x_1 \lor \overline{x}_2 \lor x_3 \lor x_4) \land (\overline{x}_2 \lor \overline{x}_3 \lor \overline{x}_4)$

  $\text{cost}_\text{of}\{x_1 = 1, \ x_2 = 1, \ x_3 = 0, \ x_4 = 1\} = 3$

  $\text{cost}_\text{of}\{x_1 = 0, \ x_2 = 0, \ x_3 = 1, \ x_4 = 1\} = 2$

The MinCostSat Problem

• MinCostSat is harder than SAT
Set Covering

Three backup positions to fill. Five players to choose from. Want to minimize the number of players.

<table>
<thead>
<tr>
<th></th>
<th>Guard</th>
<th>Forward</th>
<th>Center</th>
</tr>
</thead>
<tbody>
<tr>
<td>Duncan</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>Francis</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Malone</td>
<td>1</td>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Stockton</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Yao</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Guard | 1 | 1 | 1
Forward | 1 | 1 |
Center | 1 | 1 | 1

Native MinCostSat Problems

MinCostSat

Covering

Set Covering

Logic Minimization
Crew Scheduling
Vehicle Routing

Binate Covering

Technology Mapping
FSM Minimization
Boolean Relations

Non-Covering

ATPG
Minimum-Length Plan

2/10/17

Duncan Francis Malone Stockton Yao

Guard
Forward
Center

2/10/17
Set Covering

• Conjunctive Normal Form

  Guard  \( F \lor S \)
  Forward \( D \lor M \)
  Center \( D \lor M \lor Y \)

• Goal: minimize \( F + S + D + M + Y \)

Min-cost solutions: \( F = 1, D = 1, F = M = Y = 0 \)
\( M = 1, S = 1, D = M = Y = 0 \)

Binate Covering Problem

• Additional constraint 1:
  ➢ Malone and Stockton are together ( \( M \Leftrightarrow S \) )

\( \overline{M} \lor S \)
\( M \lor \overline{S} \)

• Additional constraint 2:
  ➢ Duncan and Stockton can’t be signed together ( \( D \land S \) )

\( D \lor \overline{S} \)
Binate Covering Problem

<table>
<thead>
<tr>
<th>constraints</th>
<th>covering matrix</th>
</tr>
</thead>
<tbody>
<tr>
<td>F ∨ S</td>
<td>D</td>
</tr>
<tr>
<td>D ∨ M</td>
<td>1</td>
</tr>
<tr>
<td>D ∨ M ∨ Y</td>
<td>1</td>
</tr>
<tr>
<td>M ∨ S</td>
<td></td>
</tr>
<tr>
<td>M ∨ S̄</td>
<td></td>
</tr>
<tr>
<td>D̄ ∨ S̄</td>
<td>-1</td>
</tr>
</tbody>
</table>

Non-Uniform Weights

<table>
<thead>
<tr>
<th>Guard</th>
<th>1</th>
<th>1</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Forward</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Center</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Goal: minimize 5D + 4F + 6M + 2S + 3Y
Non-Native MinCostSat Problems

MinCostSat

- Slack
  - MaxSat
  - Partial MaxSat
  - Group-Partial MaxSat

- Slack
  - Slack
  - Slack

Not realistic enough!
Course Assignment
FPGA Detailed Routing

The MinCostSat Hierarchy

MinCostSAT

- Slack
  - Native
    - Covering
      - Set Covering
    - Non-Covering
      - Binate Covering
  - Non-Native
    - MaxSat
    - Partial MaxSat
    - Group Partial MaxSat
Solve MinCostSat as ILP

1. MinCostSat is a special case of 0-1 integer linear programming
2. For each constraint in MinCostSat, replace $x_i$ with $1 - x_i$.

\[
\overline{x_1} \lor x_2 \lor \overline{x_3} \lor x_4 \\
(1 - x_1) + x_2 + (1 - x_3) + x_4 \geq 1 \\
- x_1 + x_2 - x_3 + x_4 \geq -1
\]

Solve MinCostSat as ILP

<table>
<thead>
<tr>
<th>MinCostSat</th>
<th>ILP</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\overline{x_1} \lor x_2 \lor \overline{x_3} \lor x_4$</td>
<td>$-x_1 + x_2 - x_3 + x_4 \geq -1$</td>
</tr>
<tr>
<td>$\overline{x_1} \lor \overline{x_2} \lor \overline{x_4}$</td>
<td>$-x_1 - x_2 - x_3 \geq -2$</td>
</tr>
<tr>
<td>$x_1 \lor x_2 \lor x_3$</td>
<td>$x_1 + x_2 + x_3 \geq 1$</td>
</tr>
</tbody>
</table>

Assume unit cost, minimize: $x_1 + x_2 + x_3 + x_4$
The Complete Hierarchy

- **ILP**
  - MinCostSAT
    - Native
      - Covering
    - Non-Native
      - Non-Covering
      - Slack

Why not use an ILP solver (cplex)?

- Set Covering slower than *eclipse*
- Non-covering slower than *eclipse_BF, bsolo*
- MaxSat slower than *qtmax*
- Group partial MaxSat slower than *wpack, subSAT*
Outline

1. The MinCostSat Problem
2. Our Contributions
3. Conclusions

Our Contributions

1. Branch-and-Bound MinCostSat Solver - \textit{eclipse}
   Set Covering Competitors
   - \textit{scherzo, aura, cplex}
   Binate Covering Competitors
   - \textit{scherzo, bsolo, cplex}

2. Local-Search MinCostSat Solver - \textit{eclipse-stoc}
Our Contributions

3. Branch-and-Bound MaxSat Solver – \textit{qtmax}
   Competitors:
   \textit{maxsat, LB2+MOMS, LB2+JW, cplex}

4. Local-Search group-partial MaxSat Solver – \textit{wpack}
   Competitors:
   \textit{sub_SAT, cplex}

Branch-and-bound Solver - Eclipse

- Seven Performance Factors
  1. Lower-Bounding
  2. Upper-Bounding
  3. Search-Tree Exploration Strategies
  4. Branching Variable Selection
  5. Search Pruning
  6. Reductions
  7. Data Structures
Lower-Bounding

1. Maximum Independent Set
2. Linear Programming Relaxation
3. Cutting Planes

Lower-Bounding With MIS

- MIS - maximum independent set of rows

<table>
<thead>
<tr>
<th></th>
<th>x1</th>
<th>x2</th>
<th>x3</th>
<th>x4</th>
<th>x5</th>
<th>x6</th>
</tr>
</thead>
<tbody>
<tr>
<td>row1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>row2</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>row3</td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>row4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>row5</td>
<td></td>
<td></td>
<td></td>
<td>1</td>
<td></td>
<td>-1</td>
</tr>
</tbody>
</table>

- Row1, 2 form an independent set $\implies$ Cost$_{lower} = 2$
- Row1, 3, 4 form an independent set $\implies$ Cost$_{lower} = 3$
Lower-Bounding with LPR

- Relax the integer constraints
- Minimization: opt for ILP is greater than opt for LP

![Diagram of Case 1 and Case 2 with Opt and R, Z]

Lower-Bounding with Gomory Cuts

Feasible region

Cut 1

Cut 2

(0.5, 0.5)

(1, 1)
## Lower-Bounding Comparisons

<table>
<thead>
<tr>
<th>benchmark</th>
<th>opt</th>
<th>MIS1</th>
<th>MIS2</th>
<th>LPR</th>
<th>CP</th>
</tr>
</thead>
<tbody>
<tr>
<td>count.b</td>
<td>24</td>
<td>17</td>
<td>17</td>
<td>24</td>
<td>24</td>
</tr>
<tr>
<td>clip.b</td>
<td>15</td>
<td>10</td>
<td>10</td>
<td>14</td>
<td>14</td>
</tr>
<tr>
<td>9sym.b</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>jac3</td>
<td>15</td>
<td>12</td>
<td>12</td>
<td>15</td>
<td>15</td>
</tr>
<tr>
<td>f51m.b</td>
<td>18</td>
<td>14</td>
<td>15</td>
<td>16</td>
<td>17</td>
</tr>
<tr>
<td>sao2.b</td>
<td>25</td>
<td>24</td>
<td>25</td>
<td>25</td>
<td>25</td>
</tr>
<tr>
<td>5xp1.b</td>
<td>12</td>
<td>9</td>
<td>9</td>
<td>11</td>
<td>11</td>
</tr>
<tr>
<td>apex4.a</td>
<td>776</td>
<td>525</td>
<td>525</td>
<td>756</td>
<td>773</td>
</tr>
<tr>
<td>rot.b</td>
<td>115</td>
<td>95</td>
<td>98</td>
<td>111</td>
<td>114</td>
</tr>
<tr>
<td>alu4.b</td>
<td>50</td>
<td>38</td>
<td>39</td>
<td>47</td>
<td>47</td>
</tr>
<tr>
<td>e64.b</td>
<td>≤ 48</td>
<td>32</td>
<td>32</td>
<td>37</td>
<td>40</td>
</tr>
<tr>
<td>e432.F37gat@1</td>
<td>9</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>misc3.Fb@1</td>
<td>8</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>c1908.F469@0</td>
<td>11</td>
<td>1</td>
<td>1</td>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>c6288.F69gat@1</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>2</td>
</tr>
<tr>
<td>c3540.F20@1</td>
<td>6</td>
<td>1</td>
<td>1</td>
<td>3</td>
<td>3</td>
</tr>
</tbody>
</table>

## Upper-Bounding

- Closing the window between lower-bound and upper-bound
- Best possible upper-bound = cost of optimal solution
- Local-Search for optimal solution
  1. Apply at each node
  2. Initialization – solution from lower-bounding
  3. Termination – exponential decay
Runtime Breakdown

Comparison on the References

<table>
<thead>
<tr>
<th>benchmark</th>
<th>scherzo</th>
<th>aura</th>
<th>cplex</th>
<th>eclipse_lpr</th>
<th>eclipse_cp</th>
</tr>
</thead>
<tbody>
<tr>
<td>exam.pi</td>
<td>3600*</td>
<td>3600*</td>
<td>3.6</td>
<td>162.2</td>
<td>22.4</td>
</tr>
<tr>
<td>ex5.pi</td>
<td>3600*</td>
<td>488.8</td>
<td>25.6</td>
<td>7.7</td>
<td>9.0</td>
</tr>
<tr>
<td>max1024</td>
<td>1749.3</td>
<td>3600*</td>
<td>18.6</td>
<td>39.3</td>
<td>15.9</td>
</tr>
<tr>
<td>prom2</td>
<td>376.9</td>
<td>3600*</td>
<td>5.2</td>
<td>2.2</td>
<td>8.0</td>
</tr>
<tr>
<td>bench1.pi</td>
<td>1349.8</td>
<td>3600*</td>
<td>4.4</td>
<td>3.8</td>
<td>2.2</td>
</tr>
<tr>
<td>steiner45</td>
<td>6.81</td>
<td>8.58</td>
<td>38.94</td>
<td>268.17</td>
<td>214.11</td>
</tr>
<tr>
<td>m200...50</td>
<td>71.4</td>
<td>14.1</td>
<td>129.1</td>
<td>65.3</td>
<td>200.7</td>
</tr>
</tbody>
</table>

* Solver times out
## Comparison on the P-classes

### Average Runtime

<table>
<thead>
<tr>
<th>benchmark</th>
<th>cplex</th>
<th>eclipse_lpr</th>
<th>eclipse_cp</th>
</tr>
</thead>
<tbody>
<tr>
<td>exam.pi</td>
<td>26.1</td>
<td>160.6</td>
<td>5.4</td>
</tr>
<tr>
<td>ex5.pi</td>
<td>65.7</td>
<td>12.8</td>
<td>16.7</td>
</tr>
<tr>
<td>max1024</td>
<td>26.6</td>
<td>52.7</td>
<td>27.5</td>
</tr>
<tr>
<td>prom2</td>
<td>5.2</td>
<td>2.8</td>
<td>8.0</td>
</tr>
<tr>
<td>bench1.pi</td>
<td>7.3</td>
<td>2.4</td>
<td>4.7</td>
</tr>
<tr>
<td>steiner45</td>
<td>39.3</td>
<td>307.1</td>
<td>229.8</td>
</tr>
<tr>
<td>m200…50</td>
<td>123.0</td>
<td>66.0</td>
<td>202.0</td>
</tr>
</tbody>
</table>

### Average Nodes

<table>
<thead>
<tr>
<th>benchmark</th>
<th>cplex</th>
<th>eclipse_lpr</th>
<th>eclipse_cp</th>
</tr>
</thead>
<tbody>
<tr>
<td>exam.pi</td>
<td>1917</td>
<td>2523</td>
<td>12</td>
</tr>
<tr>
<td>ex5.pi</td>
<td>524</td>
<td>18</td>
<td>5</td>
</tr>
<tr>
<td>max1024</td>
<td>148</td>
<td>257</td>
<td>22</td>
</tr>
<tr>
<td>prom2</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>bench1.pi</td>
<td>68</td>
<td>17</td>
<td>7</td>
</tr>
<tr>
<td>steiner45</td>
<td>74547</td>
<td>20648</td>
<td>5998</td>
</tr>
<tr>
<td>m200…50</td>
<td>31051</td>
<td>1511</td>
<td>1180</td>
</tr>
</tbody>
</table>
Improving Logic Minimization

Comparing with Heuristic Solvers

• More time, but better results

<table>
<thead>
<tr>
<th>benchmark</th>
<th>espresso</th>
<th></th>
<th>latte</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>cover</td>
<td>total</td>
<td>cover</td>
</tr>
<tr>
<td></td>
<td></td>
<td>12hrs*</td>
<td>139.24</td>
</tr>
<tr>
<td>ex5</td>
<td>–</td>
<td>12hrs*</td>
<td>129.58</td>
</tr>
<tr>
<td>max1024</td>
<td>–</td>
<td>12hrs*</td>
<td>329.11</td>
</tr>
<tr>
<td>prom2</td>
<td>–</td>
<td>12hrs*</td>
<td>12.97</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>benchmark</th>
<th>orig</th>
<th>espresso-heur</th>
<th>latte</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>minterm</td>
<td>literal</td>
<td>minterm</td>
</tr>
<tr>
<td>ex5</td>
<td>256</td>
<td>9668</td>
<td>74</td>
</tr>
<tr>
<td>max1024</td>
<td>1024</td>
<td>13472</td>
<td>274</td>
</tr>
<tr>
<td>prom2</td>
<td>287</td>
<td>5610</td>
<td>287</td>
</tr>
</tbody>
</table>
Our Contributions

1. Branch-and-Bound MinCostSat Solver – eclipse
2. Local-Search MinCostSat Solver -- eclipse-stoc
3. Branch-and-Bound MaxSat Solver – qtmax
4. Local-Search MaxSatPg Solver – wpack

Eclipse-stoc

MinCostSat: search for feasible solution with least cost

- Feasibility phase
  - How to find feasible solutions?
- Optimality phase
  - How to find feasible solutions with the least cost?
Local Search

• Which local search SAT solver to use?
  – There is no “best” local search SAT solver

• Benchmark profiling with variable immunity
  – the percentage of assigned variables along a random path of a branching tree
  – average over many branching paths

• Low variable immunity → UnitWalk
• High variable immunity → WalkSAT
Our Contributions

1. Branch-and-Bound MinCostSat Solver – eclipse
2. Local-Search MinCostSat Solver – eclipse-stoc
3. Branch-and-Bound MaxSat Solver – qtmax
4. Local-Search MaxSatPg Solver – wpack
MaxSat Problem

- Find variable assignment that maximize the number of clauses that can be satisfied at the same time
- Can transform into MinCostSat with *slack* variables
- Assign cost 0 to original variables and cost 1 to the slack variables
- Goal is to minimize the number of slack variable that are set to 1.

MaxSat Problem

- Maximum number of satisfiable clauses is 3
  \[
  \begin{align*}
  x_1 \lor x_2 & \quad x_1 \lor x_2 \lor S_1 \\
  \overline{x_1} \lor x_2 & \quad \overline{x_1} \lor x_2 \lor S_2 \\
  x_1 \lor \overline{x_2} & \quad x_1 \lor \overline{x_2} \lor S_3 \\
  \overline{x_1} \lor \overline{x_2} & \quad \overline{x_1} \lor \overline{x_2} \lor S_4
  \end{align*}
  \]

Minimize: \( S_1 + S_2 + S_3 + S_4 \)

- MinCostSat instance has optimum of 1
MaxSat Solver - qtmax

1. Lower-bounding
2. Search pruning
3. Branching variable selection
4. Efficient implementation

MaxSat vs MinCostSat vs ILP

Comparison of three state-of-the-art solvers:
1. qtmax (native MaxSat)
2. eclipse (MinCostSat)
3. cplex (ILP)
MaxSat vs MinCostSat vs ILP

Conclusions

ILP

MinCostSAT

Slack

Native

Non-Native

Covering

Non-Covering

MaxSat

Partional MaxSat

Group Partial MaxSat

Set Covering

Binate Covering
Future Research Goals

• Improved lower-bounding procedure
• Study Non-covering problems
• Complete Sat Solver using Local Search
Improved lower-bounding procedure

- Incremental lower-bounding
- Switching Strategy
  - MIS
  - Linear Programming Relaxation
  - Cutting Planes

Future Research Goals

- eclipse-bf: non-covering MinCostSat Problem

<table>
<thead>
<tr>
<th>benchmark</th>
<th>PI</th>
<th>b solo opt</th>
<th>node</th>
<th>time</th>
<th>eclipse-bf opt</th>
<th>node</th>
<th>time</th>
</tr>
</thead>
<tbody>
<tr>
<td>c432_F37gat@1</td>
<td>36</td>
<td>≤ 14</td>
<td>–</td>
<td>3000*</td>
<td>9</td>
<td>3488985</td>
<td>176.7</td>
</tr>
<tr>
<td>misex3_Fb@1</td>
<td>14</td>
<td>8</td>
<td>1119</td>
<td>28.1</td>
<td>8</td>
<td>138990</td>
<td>43.28</td>
</tr>
<tr>
<td>c1998_F469@0</td>
<td>33</td>
<td>11</td>
<td>66314</td>
<td>2854.9</td>
<td>11</td>
<td>1467317</td>
<td>479.3</td>
</tr>
<tr>
<td>c6288_F69gat@1</td>
<td>32</td>
<td>6</td>
<td>4882</td>
<td>433.24</td>
<td>6</td>
<td>1614324</td>
<td>1258.7</td>
</tr>
<tr>
<td>c3540_F20@1</td>
<td>50</td>
<td>6</td>
<td>4151</td>
<td>326.8</td>
<td>6</td>
<td>234591</td>
<td>1520.8</td>
</tr>
</tbody>
</table>

* b solo times out at 3000 seconds.