Cube and Conquer
Guiding CDCL SAT Solvers by Lookaheads

Marijn Heule\textsuperscript{1,2}, Oliver Kullmann\textsuperscript{3}, Siert Wieringa\textsuperscript{4} and Armin Biere\textsuperscript{1}

\textsuperscript{1}Johannes Kepler University, Linz, Austria
\textsuperscript{2}Delft University of Technology, Delft, The Netherlands
\textsuperscript{3}Swansea University, United Kingdom
http://cs.swan.ac.uk/~csoliver
\textsuperscript{4}Aalto University, Helsinki, Finland

Belgrade, February 3, 2012,
Fifth Workshop on Formal and Automated Theorem Proving and Applications
Best Paper Award at HVC 2011 [Heule et al., 2012]
Motivation

Context:
- Huge performance boost of CDCL solvers in the last decade
- CDCL solvers have become a crucial tool, e.g. in Formal Verification

Challenges:
- CDCL is not strong on small hard combinatorial problems
- CDCL is hard to parallelise effectively

CDCL: Conflict-Driven Clause Learning
Satisfiability problem

Satisfiability (SAT) problem:
- Given a formula in Conjunctive Normal Form, is there a truth assignment to the Boolean variables satisfying all clauses?
- clause: \((a \lor b \lor c)\) (“CNF-clause”)
- cube: \((d \land e \land f)\) (alternatively, think of it as a partial assignment).

Major complete SAT solver architectures:
- Conflict-Driven Clause Learning
- Lookahead.
Conflict-Driven Clause Learning solvers

Highlights:
- goal: find small effective conflict clauses
- decisions: assign variables that occur in recent conflicts
- strength: powerful on "easy" problems

Ideal CDCL situation:
- hit a conflict that can be generalised / analysed to a small clause
Conflict-Driven Clause Learning solvers

Highlights:
- goal: find small effective conflict clauses
- decisions: assign variables that occur in recent conflicts
- strength: powerful on "easy" problems

General CDCL situation:
- hit a conflict that can be generalised / analysed to a large clause
Lookahead solvers

Highlights:
- goal: construct a small binary search tree
- decisions: assign variables that cause a large reduction
- strength: powerful on small hard problems

Ideal lookahead situation:
- split the search space into two equally large but smaller parts
Lookahead solvers

Highlights:
  - goal: construct a small binary search tree
  - decisions: assign variables that cause a large reduction
  - strength: powerful on small hard problems

General lookahead situation:
  - the search space is split into a large and a small part
Best of both worlds: Combining Lookahead and CDCL
Best of both worlds: Cube and Conquer

Cube and Conquer

February 3, 2012, FATPA 9 / 27
Cube: key observation / contribution

Split until the (sub-)problems become easy:
- do not have a fixed cut off depth
- determine hardness by (total) number of assigned variables
- create many thousands or even millions of cubes.

General lookahead situation:
- the search space is split into a large and a small part
Cube: example

\[ F_1 := F \land (x_5 \land x_7 \land \neg x_8) \]
\[ F_2 := F \land (x_5 \land x_7 \land x_8 \land x_2) \]
\[ F_3 := F \land (x_5 \land \neg x_7 \land x_9) \]
\[ F_4 := F \land (x_5 \land \neg x_7 \land \neg x_9) \]

\[ F_5 := F \land (\neg x_5 \land \neg x_2 \land \neg x_3) \]
\[ F_6 := F \land (\neg x_5 \land x_2 \land x_8 \land x_9) \]
\[ F_7 := F \land (\neg x_5 \land x_2 \land x_8 \land \neg x_9) \]
Cube: pseudo-code (1)

Cube(CNF $F$, DNF $A$, CNF $C$, dec. lits. $\varphi_{\text{dec}}$, imp. lits. $\varphi_{\text{imp}}$) returns a pair $(\mathcal{A}, \mathcal{C})$, the list of cubes and the list of learned clauses

1. $\theta := 1.05 \cdot \theta$

2. $(F, \varphi_{\text{imp}}) := \text{lookahead\_simplify\_and\_learn}(F, \varphi_{\text{dec}}, \varphi_{\text{imp}})$

3. \textbf{if} $\varphi_{\text{dec}} \cup \varphi_{\text{imp}}$ falsify a clause in $F$ \textbf{or} $|\varphi_{\text{dec}}| > 20$ \textbf{then} $\theta := 0.7 \cdot \theta$

4. \textbf{if} $\varphi_{\text{dec}} \cup \varphi_{\text{imp}}$ falsify a clause in $F$ \textbf{then return} $(\mathcal{A}, \mathcal{C} \cup \{\neg \varphi_{\text{dec}}\})$

5. \textbf{if} cutoff heuristic is triggered \textbf{then return} $(\mathcal{A} \cup \{\varphi_{\text{dec}}\}, \mathcal{C})$

6. $l_{\text{dec}} := \text{lookahead\_decide}(F, \varphi_{\text{dec}}, \varphi_{\text{imp}})$

7. $(\mathcal{A}, \mathcal{C}) := \text{Cube}(F, \mathcal{A}, \mathcal{C}, \varphi_{\text{dec}} \cup \{l_{\text{dec}}\}, \varphi_{\text{imp}})$

8. \textbf{return} Cube($F, \mathcal{A}, \mathcal{C}, \varphi_{\text{dec}} \cup \{\neg l_{\text{dec}}\}, \varphi_{\text{imp}}$)
Cube: pseudo-code (2)

\begin{verbatim}
Cube (CNF F, DNF A, CNF C, dec. lits. \( \varphi_{\text{dec}} \), imp. lits. \( \varphi_{\text{imp}} \))
1 \( \theta := 1.05 \cdot \theta \)
2 \( \langle F, \varphi_{\text{imp}} \rangle := \text{lookahead_simplify_and_learn} (F, \varphi_{\text{dec}}, \varphi_{\text{imp}}) \)
3 \textbf{if } \varphi_{\text{dec}} \cup \varphi_{\text{imp}} \text{ falsify a clause in } F \text{ or } |\varphi_{\text{dec}}| > 20 \text{ then } \theta := 0.7 \cdot \theta \)
4 \textbf{if } \varphi_{\text{dec}} \cup \varphi_{\text{imp}} \text{ falsify a clause in } F \text{ then return } \langle A, C \cup \{\neg \varphi_{\text{dec}}\} \rangle \)
5 \textbf{if } |\varphi_{\text{dec}}| \cdot |\varphi_{\text{dec}} \cup \varphi_{\text{imp}}| > \theta \cdot |\text{vars}(F)| \text{ then return } \langle A \cup \{\varphi_{\text{dec}}\}, C \rangle \)
6 \( l_{\text{dec}} := \text{lookahead_decide} (F, \varphi_{\text{dec}}, \varphi_{\text{imp}}) \)
7 \( \langle A, C \rangle := \text{Cube} (F, A, C, \varphi_{\text{dec}} \cup \{l_{\text{dec}}\}, \varphi_{\text{imp}}) \)
8 \textbf{return } \text{Cube} (F, A, C, \varphi_{\text{dec}} \cup \{\neg l_{\text{dec}}\}, \varphi_{\text{imp}}) \)
\end{verbatim}
Conquer: describing cubes

How much information to send to the CDCL solver?

- Only the decisions

- The full assignment (including failed literals)

- The simplified formula (including local learnt clauses)
Conquer: ordering cubes

What is the optimal order to solve the cubes?

- Depth-first search (in lookahead order)

  1  2  3  4  5  6

  Solves cubes with increasing (approximated) search space

  4  1  5  3  6  2

  Solves cubes with decreasing (approximated) search space

  2  6  3  5  1  4
Conquer: pseudo-code

Conquer (CDCL solver $S$, CNF formula $F$, DNF of assumptions $A$)

1. $S$.Load ($F$)
2. while $A$ is not empty do
3. get a cube $c$ from $A$ and remove $c$ from $A$
4. if $S$.SolveWithAssumptions ($c$) = satisfiable then
5. return satisfiable
6. $S$.AnalyzeFinal ()
7. $S$.ResetClauseDeletionPolicy ()
8. return unsatisfiable
Conquer: parallel solving

Strategies to solve cubes in parallel:

1. cores solve different cubes in parallel
2. cores solve the same cube in parallel
3. start with (1) till no new cubes are available, continue with (2)

What to share between cores?

- nothing, so hardly communication required (only ask / receive cubes)
- sharing the AnalyzeFinal clauses (maybe only to master)
- sharing the short conflict clauses, units (maybe also binaries)
Conquer: parallel solving

Strategies to solve cubes in parallel:
1. cores solve different cubes in parallel
2. cores solve the same cube in parallel
3. start with (1) till no new cubes are available, continue with (2)

What to share between cores?
- nothing, so hardly communication required (only ask / receive cubes)
- sharing the *AnalyzeFinal* clauses (maybe only to master)
- sharing the short conflict clauses, units (maybe also binaries)
Results: two experiments

1\textsuperscript{st} experiment: single core on Van der Waerden numbers

- hard combinatorial problem in Ramsey Theory
- comparison with the best solver for each instance
- cube solver: 0Ksolver
- conquer solver: minisat-2.2.0
- describing the cubes (just) by the naked simplified formula (applying the partial assignments; without any local learning).

2\textsuperscript{nd} experiment: multi core on challenging applications

- unsolved application instances from the SAT09 benchmarks
- comparison with the best parallel solvers
- cube solver: march
- conquer solver: lingeling.
Results: palindromic Van der Waerden numbers

- $k_1$: arithmetic progression of first set
- $k_2$: arithmetic progression of second set
- $n$: number of variables
- best solver: time of fastest sequential solver
- $D$: cut off depth.

<table>
<thead>
<tr>
<th>$k_1$</th>
<th>$k_2$</th>
<th>$n$</th>
<th>$#\text{cls}$</th>
<th>$?\text{?}$</th>
<th>best solver</th>
<th>$D$</th>
<th>$#\text{cubes}$</th>
<th>$\text{C&amp;C}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>25</td>
<td>294</td>
<td>45779</td>
<td>U</td>
<td>$\sim$ 13 days</td>
<td>45</td>
<td>9120</td>
<td>6.5 hours</td>
</tr>
<tr>
<td>3</td>
<td>25</td>
<td>304</td>
<td>49427</td>
<td>U</td>
<td>$\sim$ 13 days</td>
<td>45</td>
<td>13462</td>
<td>2 days</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>194</td>
<td>15544</td>
<td>U</td>
<td>$&gt; 14$ days</td>
<td>30</td>
<td>132131</td>
<td>2 days</td>
</tr>
<tr>
<td>4</td>
<td>12</td>
<td>198</td>
<td>15889</td>
<td>U</td>
<td>$&gt; 14$ days</td>
<td>34</td>
<td>147237</td>
<td>8 hours</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>157</td>
<td>9121</td>
<td>U</td>
<td>3.5 days</td>
<td>20</td>
<td>2248</td>
<td>5 hours</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
<td>162</td>
<td>9973</td>
<td>U</td>
<td>53 days</td>
<td>20</td>
<td>87667</td>
<td>40 hours</td>
</tr>
</tbody>
</table>

See [Ahmed et al., 2011, Kullmann, 2012].
Results: parallel SAT solving

Portfolio solvers:
- run multiple versions of the same solver (different seeds)
- share short conflict clauses such as units
- solver pLingeling (pLing), on a 12-core machine

Grid based SAT solving approach:
- run solvers with different cubes on a grid
- grid constraints: limited communication, possible delay and timeout
- solver PartitionTree (PTree) on a grid, up to 60 jobs in parallel
<table>
<thead>
<tr>
<th>Benchmark</th>
<th>?</th>
<th>#cubes</th>
<th>I</th>
<th>II</th>
<th>II</th>
<th>pLing</th>
<th>PTree</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9dlx_vliw_at_b_iq8</td>
<td>U</td>
<td>121</td>
<td>150</td>
<td></td>
<td></td>
<td>3256</td>
<td></td>
</tr>
<tr>
<td>9dlx_vliw_at_b_iq9</td>
<td>U</td>
<td>100</td>
<td>179</td>
<td></td>
<td></td>
<td>5164</td>
<td></td>
</tr>
<tr>
<td>AProVE07-25</td>
<td>U</td>
<td>84247</td>
<td>89</td>
<td>100340</td>
<td>8690</td>
<td></td>
<td>9967</td>
</tr>
<tr>
<td>dated-5-19-u</td>
<td>U</td>
<td>57716</td>
<td>418</td>
<td>3214</td>
<td>1451</td>
<td>4465</td>
<td>2522</td>
</tr>
<tr>
<td>eq.atree.braun.12</td>
<td>U</td>
<td>86541</td>
<td>85</td>
<td>3261</td>
<td>273</td>
<td></td>
<td>4691</td>
</tr>
<tr>
<td>eq.atree.braun.13</td>
<td>U</td>
<td>81313</td>
<td>77</td>
<td>18165</td>
<td>1517</td>
<td></td>
<td>9972</td>
</tr>
<tr>
<td>gss-24-s100</td>
<td>S</td>
<td>18237</td>
<td>48</td>
<td>4975</td>
<td>415</td>
<td>2930</td>
<td>3492</td>
</tr>
<tr>
<td>gss-26-s100</td>
<td>S</td>
<td>19455</td>
<td>57</td>
<td>37259</td>
<td>3108</td>
<td>18173</td>
<td>10347</td>
</tr>
<tr>
<td>gus-md5-14</td>
<td>U</td>
<td>60102</td>
<td>961</td>
<td></td>
<td></td>
<td></td>
<td>13890</td>
</tr>
<tr>
<td>ndhf_xits_09_UNS</td>
<td>U</td>
<td>37358</td>
<td>82</td>
<td>71096</td>
<td>12041</td>
<td></td>
<td>9583</td>
</tr>
<tr>
<td>rbcl_xits_09_UNK</td>
<td>U</td>
<td>54669</td>
<td>132</td>
<td>94911</td>
<td>11542</td>
<td></td>
<td>9819</td>
</tr>
<tr>
<td>rpoc_xits_09_UNS</td>
<td>U</td>
<td>30681</td>
<td>114</td>
<td>48028</td>
<td>8366</td>
<td></td>
<td>8635</td>
</tr>
<tr>
<td>sortnet-8-ipc5-h19</td>
<td>S</td>
<td>724</td>
<td>153</td>
<td>48668</td>
<td>4067</td>
<td>2700</td>
<td>4304</td>
</tr>
<tr>
<td>total-10-17-u</td>
<td>U</td>
<td>9192</td>
<td>288</td>
<td>5638</td>
<td>4517</td>
<td>3672</td>
<td>4447</td>
</tr>
<tr>
<td>total-5-15-u</td>
<td>U</td>
<td>14914</td>
<td>215</td>
<td></td>
<td></td>
<td></td>
<td>18670</td>
</tr>
</tbody>
</table>
## Results: hard application benchmarks

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>?</th>
<th>#cubes</th>
<th>I total</th>
<th>II total</th>
<th>II 12-core</th>
<th>pLing 12-core</th>
<th>PTree 60-core</th>
</tr>
</thead>
<tbody>
<tr>
<td>9dlx_vliw_at_b_iq8</td>
<td>U</td>
<td>121</td>
<td>150</td>
<td>—</td>
<td>—</td>
<td>3256</td>
<td>—</td>
</tr>
<tr>
<td>9dlx_vliw_at_b_iq9</td>
<td>U</td>
<td>100</td>
<td>179</td>
<td>—</td>
<td>—</td>
<td>5164</td>
<td>—</td>
</tr>
<tr>
<td>AProVE07-25</td>
<td>U</td>
<td>84247</td>
<td>89 100340</td>
<td>8690</td>
<td>—</td>
<td>9967</td>
<td>9992</td>
</tr>
<tr>
<td>dated-5-19-u</td>
<td>U</td>
<td>57716</td>
<td>418 3214</td>
<td>1451</td>
<td>4465</td>
<td>2522</td>
<td>4691</td>
</tr>
<tr>
<td>eq.atree.braun.12</td>
<td>U</td>
<td>86541</td>
<td>85 3261</td>
<td>273</td>
<td>—</td>
<td>4691</td>
<td>9972</td>
</tr>
<tr>
<td>eq.atree.braun.13</td>
<td>U</td>
<td>81313</td>
<td>77 18165</td>
<td>1517</td>
<td>—</td>
<td>9972</td>
<td>10347</td>
</tr>
<tr>
<td>gss-24-s100</td>
<td>S</td>
<td>18237</td>
<td>48 4975</td>
<td>415</td>
<td>2930</td>
<td>3492</td>
<td></td>
</tr>
<tr>
<td>gss-26-s100</td>
<td>S</td>
<td>19455</td>
<td>57 37259</td>
<td>3108</td>
<td>18173</td>
<td>10347</td>
<td></td>
</tr>
<tr>
<td>gus-md5-14</td>
<td>U</td>
<td>60102</td>
<td>961</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>13890</td>
</tr>
<tr>
<td>ndhf_xits_09_UNS</td>
<td>U</td>
<td>37358</td>
<td>82 71096</td>
<td>12041</td>
<td>—</td>
<td>9583</td>
<td></td>
</tr>
<tr>
<td>rbcl_xits_09_UNK</td>
<td>U</td>
<td>54669</td>
<td>132 94911</td>
<td>11542</td>
<td>—</td>
<td>9819</td>
<td></td>
</tr>
<tr>
<td>rpoc_xits_09_UNS</td>
<td>U</td>
<td>30681</td>
<td>114 48028</td>
<td>8366</td>
<td>—</td>
<td>8635</td>
<td></td>
</tr>
<tr>
<td>sortnet-8-ipc5-h19</td>
<td>S</td>
<td>724</td>
<td>153 48668</td>
<td>4067</td>
<td>2700</td>
<td>4304</td>
<td></td>
</tr>
<tr>
<td>total-10-17-u</td>
<td>U</td>
<td>9192</td>
<td>288 5638</td>
<td>4517</td>
<td>3672</td>
<td>4447</td>
<td></td>
</tr>
<tr>
<td>total-5-15-u</td>
<td>U</td>
<td>14914</td>
<td>215</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>18670</td>
</tr>
</tbody>
</table>

O Kullmann (Swansea University)  
Cube and Conquer  
February 3, 2012, FATPA  
22 / 27
## Results: hard application benchmarks

<table>
<thead>
<tr>
<th>Benchmark</th>
<th>?</th>
<th>#cubes</th>
<th>I total</th>
<th>II total</th>
<th>II 12-core</th>
<th>pLing 12-core</th>
<th>PTree 60-core</th>
</tr>
</thead>
<tbody>
<tr>
<td>9dlx_vliw_at_b_iq8</td>
<td>U</td>
<td>121</td>
<td>150</td>
<td>—</td>
<td>—</td>
<td>3256</td>
<td>—</td>
</tr>
<tr>
<td>9dlx_vliw_at_b_iq9</td>
<td>U</td>
<td>100</td>
<td>179</td>
<td>—</td>
<td>—</td>
<td>5164</td>
<td>—</td>
</tr>
<tr>
<td>AProVE07-25</td>
<td>U</td>
<td>84247</td>
<td>89</td>
<td>100340</td>
<td>8690</td>
<td>—</td>
<td>9967</td>
</tr>
<tr>
<td>dated-5-19-u</td>
<td>U</td>
<td>57716</td>
<td>418</td>
<td>3214</td>
<td>1451</td>
<td>4465</td>
<td>2522</td>
</tr>
<tr>
<td>eq.atree.braun.12</td>
<td>U</td>
<td>86541</td>
<td>85</td>
<td>3261</td>
<td>273</td>
<td>—</td>
<td>4691</td>
</tr>
<tr>
<td>eq.atree.braun.13</td>
<td>U</td>
<td>81313</td>
<td>77</td>
<td>18165</td>
<td>1517</td>
<td>—</td>
<td>9972</td>
</tr>
<tr>
<td>gss-24-s100</td>
<td>S</td>
<td>18237</td>
<td>48</td>
<td>4975</td>
<td>415</td>
<td>2930</td>
<td>3492</td>
</tr>
<tr>
<td>gss-26-s100</td>
<td>S</td>
<td>19455</td>
<td>57</td>
<td>37259</td>
<td>3108</td>
<td>18173</td>
<td>10347</td>
</tr>
<tr>
<td>gus-md5-14</td>
<td>U</td>
<td>60102</td>
<td>961</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>13890</td>
</tr>
<tr>
<td>ndhf_xits_09_UNS</td>
<td>U</td>
<td>37358</td>
<td>82</td>
<td>71096</td>
<td>12041</td>
<td>—</td>
<td>9583</td>
</tr>
<tr>
<td>rbcl_xits_09_UNK</td>
<td>U</td>
<td>54669</td>
<td>132</td>
<td>94911</td>
<td>11542</td>
<td>—</td>
<td>9819</td>
</tr>
<tr>
<td>rpoc_xits_09_UNS</td>
<td>U</td>
<td>30681</td>
<td>114</td>
<td>48028</td>
<td>8366</td>
<td>—</td>
<td>8635</td>
</tr>
<tr>
<td>sortnet-8-ipc5-h19</td>
<td>S</td>
<td>724</td>
<td>153</td>
<td>48668</td>
<td>4067</td>
<td>2700</td>
<td>4304</td>
</tr>
<tr>
<td>total-10-17-u</td>
<td>U</td>
<td>9192</td>
<td>288</td>
<td>5638</td>
<td>4517</td>
<td>3672</td>
<td>4447</td>
</tr>
<tr>
<td>total-5-15-u</td>
<td>U</td>
<td>14914</td>
<td>215</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>18670</td>
</tr>
</tbody>
</table>
Conclusions

Cube and Conquer:
- effectively combining lookahead and CDCL
- many thousands or even million of cubes
- natural to parallelise

Future work, online scheduling:
- adjust heuristics based on AnalyzeFinal
- communication between solvers
- all-in CDCL

Future work, theoretical foundations:
- create a proof-theoretic framework for understanding “tree-like versus dag-like resolution” and their interaction
- better understanding of cdcl-proof-systems in this context.


End