Fast Normalization in the HOL Theorem Prover

Joe Hurd
joe.hurd@cl.cam.ac.uk

University of Cambridge
The Problem with CNF

Sometimes converting terms to CNF makes their size explode:

\[
\text{CNF} \left( (a_0 \land a_1 \land a_2 \land a_3) \lor (b_0 \land b_1 \land b_2 \land b_3) \lor (c_0 \land c_1 \land c_2 \land c_3) \lor (d_0 \land d_1 \land d_2 \land d_3) \right) =
\]

\[
(a_3 \lor b_3 \lor c_3 \lor d_0) \land (a_2 \lor b_3 \lor c_3 \lor d_0) \land
(a_1 \lor b_3 \lor c_3 \lor d_0) \land (a_0 \lor b_3 \lor c_3 \lor d_0) \land
\ldots \text{992 more atoms} \ldots
\]

\[
(a_0 \lor b_3 \lor c_3 \lor d_3) \land (a_1 \lor b_3 \lor c_3 \lor d_3) \land
(a_2 \lor b_3 \lor c_3 \lor d_3) \land (a_3 \lor b_3 \lor c_3 \lor d_3)
\]

Disastrous if we’re converting to CNF for a SAT solver
Definitional CNF guarantees the size of normalized terms will be linear in the size of original terms:

\[
\text{DEF\_CNF}\left( (a_0 \land a_1 \land a_2 \land a_3) \lor (b_0 \land b_1 \land b_2 \land b_3) \lor (c_0 \land c_1 \land c_2 \land c_3) \lor (d_0 \land d_1 \land d_2 \land d_3) \right) =
\]

\[
\exists v_0, v_1, v_2, v_3, v_4, v_5, v_6, v_7, v_8, v_9, v_{10}, v_{11}.
(v_{11} \lor \neg d_0 \lor \neg v_{10}) \land (v_{10} \lor \neg v_{11}) \land (d_0 \lor \neg v_{11}) \land
(v_{10} \lor \neg d_1 \lor \neg v_9) \land (v_9 \lor \neg v_{10}) \land (d_1 \lor \neg v_{10}) \land
\ldots 59 \text{ more atoms} \ldots
\]

\[
(v_0 \lor \neg v_1) \land (a_1 \lor \neg v_1) \land (v_0 \lor \neg a_2 \lor \neg a_3) \land
(a_3 \lor \neg v_0) \land (a_2 \lor \neg v_0) \land (v_2 \lor v_5 \lor v_8 \lor v_{11})
\]
Given an input term $t$, it’s easy to generate the definitional CNF normalized term $t'$. This allows a fast oracle implementation of normalization into definitional CNF:

\[
\{ORACLE\_SAYS\} \vdash t \iff t'
\]

We’d prefer a fully-expansive HOL proof that $t$ and $t'$ are logically equivalent:

\[
\vdash t \iff t'
\]

Unfortunately, the naive algorithm to derive this theorem is (at least) quadratic in the size of $t$. 
Fast Definitional CNF

- Can use a technique invented by Harrison to perform BDD operations by fully-expansive proof.
- Runs in (nearly) linear time, by making extensive use of proforma theorems.
- Essential part of this: instead of introducing many new boolean variables, we use one variable vector:

\[
\text{FAST}_\text{DEF}_\text{CNF} (\cdots )
\]

\[
= \exists v : \mathbb{N} \rightarrow \mathbb{B}.
\]

\[
(v(11) \lor \neg d_0 \lor \neg v(10)) \land (v(10) \lor \neg v(11)) \land
\]

\[
(d_0 \lor \neg v(11)) \land (v(10) \lor \neg d_1 \lor \neg v(9)) \land \cdots
\]
Performance Results

We compare the different methods on the $ADD_4$ term (from hardware verification) containing 1111 atoms.

<table>
<thead>
<tr>
<th>Operation on the $ADD_4$ Term</th>
<th>Time (s)</th>
<th>Infs.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Definitional NNF</td>
<td>10.610</td>
<td>7677</td>
</tr>
<tr>
<td>Oracle definitional CNF</td>
<td>0.390</td>
<td>0</td>
</tr>
<tr>
<td>Naive definitional CNF</td>
<td>122.620</td>
<td>12034</td>
</tr>
<tr>
<td>Fast definitional CNF</td>
<td>28.800</td>
<td>238258</td>
</tr>
<tr>
<td>Applying the $zCHAFF$ solver</td>
<td>4.680</td>
<td>0</td>
</tr>
</tbody>
</table>

Observe that the fast method uses more HOL inference steps than the naive method, but takes much less time.