Model checking with edge-valued decision diagrams

Pierre Roux\textsuperscript{1}  
Radu I. Siminiceanu\textsuperscript{2}

NASA Formal Methods Symposium  
April 15, 2010

\textsuperscript{1}ENS Lyon, France (pierre.roux@ens-lyon.org)  
\textsuperscript{2}NIA (radu@nianet.org)
1 Decision Diagrams

2 EVMDDs

3 Implementation
The State of Symbolic Model Checking
Research

Evolution and Impact of Decision Diagrams

- Late 80s - early 90s: the wow factor, BDDs are (re)discovered
- Late 90s - early 00s: real progress
  - Extensions, generalizations (MTBDDs, BMDs, EVMDDs, etc)
  - New techniques (saturation, BMC, CEGAR, interpolation)
- Since then ...
  - Interest has shifted to other areas (SAT/SMT solving)
  - There are even rumors out there that symbolic MC has entered a "Brezhnevian era" (stagnation)
  - Fact or fiction?
Stagnation: fact or fiction?

- A little bit of both
- New ideas exist, but are disparate
- Examples of untapped resources:
  - Edge-valued decision diagrams (EVMDD)
  - Identity-reduced decision diagrams
  - Hashing, caching, garbage collection
  - Guided search heuristics

Our (declared) goal

Represent in one formalism (some of) the best techniques available at the moment across a spectrum of existing tools
Encoding of functions

The advent of symbolic MC: **compact** representation of

- boolean functions $f : \{0, 1\}^n \to \{0, 1\}$
- sets $\{x \in \{0, 1\}^n \mid f(x) = 1\}$

Evolution:

- Truth table: $2^n$ entries
- Binary Decision Diagram (BDD): merge common subtrees
  
  **still exponential size in worst case, often better in practice**

<table>
<thead>
<tr>
<th>$a$</th>
<th>$b$</th>
<th>$c$</th>
<th>$f(a, b, c)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Model checking with edge-valued decision diagrams
Integer/arithmetic functions

- \( f : \{0, 1\}^n \rightarrow \mathbb{Z} \)
- Extend BDD to **Multi-Terminal BDD (MTBDD)**

```
<table>
<thead>
<tr>
<th>a</th>
<th>0</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>b</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>
```

*Figure: \( f : (a, b) \mapsto 2a + b \)*

- Inefficient if \( \text{Img}(f) \) is large: less chances to share subtrees

**Examples of other forms of DDs:**
- Multiway DDs (MDD): \( f : \{0, \ldots, k_1\} \times \cdots \times \{0, \ldots, k_n\} \rightarrow \{0, 1\} \)
- Binary Moment Diagrams (BMD):
  - work well for multipliers, but not much else
Edge Valued MDDs (EVMDDs)

- EVBDDs introduced in 1992, but not sufficiently exploited
  ⇒ (Reed-Müller spectrum !?!)

- From MTBDDs to EVMDDs:
  merge all terminals (0) and assign (integer) values to edges

- Value of $f$: composition of edge-values (e.g. addition, $+$) along the path from root to terminal node
EVMDD characteristics

- EVMDD encoding is smaller than MTBDDs (# nodes)
  ⇒ proved in this paper
- Size can be linear instead of exponential (e.g. linear functions)
- Composition ⇒ a generic algorithm for all binary operators:
  for \( f, g \) encoded by EVMDDs of size \(|f|\) and \(|g|\)
  \( f \otimes g \) computed in \( O(|f| |g| |\text{Img}(f)| |\text{Img}(g)|) \)
- The algorithm has **exactly the same complexity**
  as its equivalent for MTBDDs, hence
  **no gain** in (worst-case) time complexity
- Is there room for improvement?
EV$^+$MDD algorithms

Yes, for following operations:

- **Addition:**
  \[ f + g \text{ computed in } O(|f| \cdot |g|) \]
  (actually better with QEV$^+$MDDs)

- **Relational operators:**
  \[ f \sqcap c \text{ computed in } O(c \cdot |f|) \]
  \[ f \sqcap g \text{ computed in } O(|f| \cdot |g|) \]

- **Multiplication:**
  \[ f \times g \text{ computed in } O(|f|^2 \cdot |g|^2 \cdot |f \times g|) \]
  - exponential in worst case
  - much better in many “practical” cases

- **Remainder and Euclidean division by constant:**
  \[ f/c \text{ and } f\%c \text{ computed in } O(c \cdot |f|) \]
An EVMDD-based Model Checker

We have developed an EVMDD library featuring:

- EVMDDs for arithmetic expressions
- (Regular) MDDs for boolean expressions
- Identity-reduced encoding of transition relations
- Saturation-based state space construction
- Unsophisticated (i.e. fast) garbage collector (mark & sweep)

Some stats:

- 7 kLOC of ANSI C : library
- 4 kLOC : model checking front-end

Available at http://research.nianet.org/~radu/evmdd/
## Results

Building state space vs CUDD (BFS) and SMART (saturation)

<table>
<thead>
<tr>
<th>Model</th>
<th>Model size</th>
<th>Reachable states</th>
<th>CUDD (sec)</th>
<th>SMART (sec)</th>
<th>EVMDD (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dining philosophers</td>
<td>100</td>
<td>$4 \times 10^{62}$</td>
<td>11.42</td>
<td>1.49</td>
<td>0.03</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>$2 \times 10^{125}$</td>
<td>3054.69</td>
<td>3.03</td>
<td>0.07</td>
</tr>
<tr>
<td></td>
<td>15000</td>
<td>$2 \times 10^{9404}$</td>
<td>—</td>
<td>—</td>
<td>195.29</td>
</tr>
<tr>
<td>Round robin</td>
<td>40</td>
<td>$9 \times 10^{13}$</td>
<td>4.44</td>
<td>0.44</td>
<td>0.08</td>
</tr>
<tr>
<td>mutual exclusion</td>
<td>100</td>
<td>$2 \times 10^{32}$</td>
<td>—</td>
<td>2.84</td>
<td>1.17</td>
</tr>
<tr>
<td>protocol</td>
<td>200</td>
<td>$7 \times 10^{62}$</td>
<td>—</td>
<td>20.02</td>
<td>9.14</td>
</tr>
<tr>
<td>Slotted ring</td>
<td>10</td>
<td>$8 \times 10^{9}$</td>
<td>1.16</td>
<td>0.19</td>
<td>0.01</td>
</tr>
<tr>
<td>protocol</td>
<td>20</td>
<td>$2 \times 10^{20}$</td>
<td>—</td>
<td>0.71</td>
<td>0.04</td>
</tr>
<tr>
<td></td>
<td>200</td>
<td>$8 \times 10^{211}$</td>
<td>—</td>
<td>412.27</td>
<td>25.97</td>
</tr>
</tbody>
</table>

On Intel Core 2, 1.2GHz, 1.5GB mem ("—" means “> 1h”).
## Results

### Building state space vs CUDD (BFS) and SMART (saturation)

<table>
<thead>
<tr>
<th>Model</th>
<th>Model size</th>
<th>Reachable states</th>
<th>CUDD (sec)</th>
<th>SMART (sec)</th>
<th>EVMDD (sec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kanban assembly line</td>
<td>15</td>
<td>$4 \times 10^{10}$</td>
<td>80.43</td>
<td>3.41</td>
<td>0.01</td>
</tr>
<tr>
<td></td>
<td>20</td>
<td>$8 \times 10^{11}$</td>
<td>2071.58</td>
<td>8.23</td>
<td>0.02</td>
</tr>
<tr>
<td></td>
<td>400</td>
<td>$6 \times 10^{25}$</td>
<td>—</td>
<td>—</td>
<td>74.89</td>
</tr>
<tr>
<td>Knights problem</td>
<td>5</td>
<td>$6 \times 10^{7}$</td>
<td>1024.42</td>
<td>5.29</td>
<td>0.27</td>
</tr>
<tr>
<td></td>
<td>7</td>
<td>$1 \times 10^{15}$</td>
<td>—</td>
<td>167.41</td>
<td>3.46</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>$8 \times 10^{24}$</td>
<td>—</td>
<td>—</td>
<td>32.20</td>
</tr>
<tr>
<td>Randomized leader election</td>
<td>6</td>
<td>$2 \times 10^{6}$</td>
<td>4.22</td>
<td>8.42</td>
<td>0.86</td>
</tr>
<tr>
<td></td>
<td>9</td>
<td>$5 \times 10^{9}$</td>
<td>—</td>
<td>954.81</td>
<td>18.89</td>
</tr>
<tr>
<td></td>
<td>11</td>
<td>$9 \times 10^{11}$</td>
<td>—</td>
<td>—</td>
<td>109.25</td>
</tr>
</tbody>
</table>

On Intel Core 2, 1.2GHz, 1.5GB mem (“—” means “> 1h”).

Model checking with edge-valued decision diagrams
Questions

?