Introduction

- Combining two frameworks
  - AND/OR Search Spaces
  - Multi-valued Decision Diagrams (MDDs)
- Both are more compact ways to represent problems.
- Their combination yields an even more compact representation.
- Decision Diagrams are known to allow online speed queries.
Introduction

$F(ACF)$
$F(ABE)$
$F(BC)$
$F(BD)$
$F(CD)$
$F(ABC)$

OFFLINE

ONCE

ONLINE

Compiled structure

Evaluate answer
Polytime? Constant time?
Ordered Binary Decision Diagram

B = \{0,1\} \quad f : B^3 \rightarrow B

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>f(ABC)</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>0</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>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</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>

Table

Decision tree

1) Merge nodes with identical children

2) Remove redundant nodes

Ordering enables efficient operations
Decision Diagrams

\[ f = (A \lor E) \land (B \lor F) \]

\[ g = (C \lor G) \land (D \lor H) \]

\[ f \land g = \]

\[ \text{OBDD} \]

\[ f \land g = \]

\[ \text{OBDD} \]
AOMDDs

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>f(ABC)</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>0</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>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</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>

Point dead-ends to terminal node “0”

Point goods to terminal node “1”

Minimal AND/OR graph

Decision Diagram
Removing Redundancy

Group OR node together with its AND children into a meta-node

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>f(ABC)</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>0</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>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</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>

OBDD (pseudo tree is a chain)
AOBDD
Apply Operator
Apply Operator

\[
\begin{align*}
A_1 A_2 A_3 A_4 A_5 B_1 B_2 B_3 B_4 B_5 B_6 B_7 A_1 B_1 A_2 B_2 A_4 B_4 B_6 = 0
\end{align*}
\]
Apply Operator

A1 \odot B1 = A1B1

A1 = 01
B1 = 01
A1B1 = 01

A2 \odot B2 = A2B2

A2 = 01
B2 = 01
A2B2 = 01
Apply Operator
Apply Operator

\[ A_1 \otimes \ldots \otimes B_1 \otimes \ldots \otimes B_6 = \ldots \otimes A_4 \otimes B_4 \otimes B_5 \]
And/Or Multi-Valued Decision Diagrams

- AOMDDs are:
  - AND/OR search graphs
  - canonical representations, given a pseudo tree

- Defined by two rules:
  - All isomorphic subgraphs are merged
  - There are no redundant (meta) nodes
Example: \((f \lor h) \land (a \lor \neg h) \land (a \# b \# g) \land (f \lor g) \land (b \lor f) \land (a \lor e) \land (c \lor e) \land (c \# d) \land (b \lor c)\)
Example (continued)
AOBDD vs. OBDD

AOBDD
18 nonterminals
47 arcs

OBDD
27 nonterminals
54 arcs

primal graph
Constraint Optimization - AND/OR Tree
AND/OR Context Minimal Graph

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>f₁(ABC)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>∞</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>D</th>
<th>f₂(ABD)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>∞</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>2</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>5</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>6</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>5</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>B</th>
<th>D</th>
<th>E</th>
<th>f₃(BDE)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>∞</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>4</td>
</tr>
</tbody>
</table>

f₁(ABC) f₂(ABD) f₃(BDE)
AOMDD – Compilation by Search

redundant
AOMDD – Compilation by Search
AOMDD – Compilation by Search

isomorphic
AOMDD – Compilation by Search

![Diagram of AOMDD compilation process with nodes A, B, C, D, E and edges showing the compilation process. Nodes are labeled with values indicating states or decisions.]
AOMDD for Constraint Optimization

AOMDD for all solutions

AOMDD for two best solutions

A=0
B=1
C=1
D=0
E=1
cost = 5

A=0
B=1
C=1
D=1
E=1
cost = 8

A=1
B=0
C=1
D=0
E=1
cost = 11

A=1
B=0
C=1
D=1
E=1
cost = 11

A=1
B=1
C=1
D=0
E=1
cost = 11

A=1
B=1
C=1
D=1
E=1
cost = 11
Complexity of Compilation

- The size of the AOMDD is $O(n \, k^{w^*})$

- The compilation time is also bounded by $O(n \, k^{w^*})$

$k = \text{domain size}$

$n = \text{number of variables}$

$w^* = \text{treewidth}$
Semantic Treewidth

- Given a network, there may exist a sparser equivalent network.
- Challenges the idea of using induced width to measure the difficulty of the problem.
- AOMDD sizes are much smaller than the bound.
Semantic Treewidth

- With respect to a pseudo tree, this is the smallest treewidth over all equivalent networks that can have that pseudo tree.
- With respect to the network, this is the smallest semantic treewidth over all pseudo trees that can express the set of solutions.
- Instead of the induced width bounding AOMDD size, we can use semantic treewidth.
## Semantic Treewidth

### Equality constraint network results

<table>
<thead>
<tr>
<th>eq(n,10)</th>
<th>graph</th>
<th>c</th>
<th>w*</th>
<th>h</th>
<th>time</th>
<th>#aomdd</th>
</tr>
</thead>
<tbody>
<tr>
<td>n=10</td>
<td>chain</td>
<td>9</td>
<td>1</td>
<td>5</td>
<td>0.0240</td>
<td>91</td>
</tr>
<tr>
<td></td>
<td>complete</td>
<td>45</td>
<td>9</td>
<td>9</td>
<td>0.0660</td>
<td>91</td>
</tr>
<tr>
<td>n=50</td>
<td>chain</td>
<td>49</td>
<td>1</td>
<td>25</td>
<td>0.1420</td>
<td>491</td>
</tr>
<tr>
<td></td>
<td>complete</td>
<td>1225</td>
<td>49</td>
<td>49</td>
<td>1.1130</td>
<td>491</td>
</tr>
<tr>
<td>n=100</td>
<td>chain</td>
<td>99</td>
<td>1</td>
<td>50</td>
<td>0.3120</td>
<td>991</td>
</tr>
<tr>
<td></td>
<td>complete</td>
<td>4950</td>
<td>99</td>
<td>99</td>
<td>5.5900</td>
<td>991</td>
</tr>
</tbody>
</table>
Constraint Propagation

- We can also prune the search space during compilation without removing possible solutions.
- In Bayesian networks, prune a subtree if the weight of the assignment is 0.
Experiments

- What about the pseudo-tree height parameter?
- Problems: WCSP instances
  - ISCAS 89 Circuits
  - SPOT5 Satellites
  - Mastermind
  - CELAR6 Radio Frequencies
- Time bound for compilation: 3 hours
Experiments

- Compilation was for finding the optimal solution
- Used AOBB with static mini-bucket heuristics (i-bound = 10)
- Tried different implementations of MinFill
  - Existing implementation in the compiler
  - daoopt (gets lower h because it considers it too)
  - CVO
Experiments

- BnB pruning makes the size unpredictable as a function of the parameters
- Need to modify the routine for solution counting so the entire AOMDD is actually compiled
Experiments (BN)

- Reproduce and extend BN results in JAIR 2008 paper
- UAI 2006 Bayesian network benchmarks
  - Domain sizes of 2
  - Evidence on 30 random variables (to simplify the networks slightly)
  - Many elements with “0” support
- Compile with constraint propagation
To do

- Perform experiments to compare optimization vs. full compilation
  - Need to extend code for WCSPs
- Another way to deal with unpredictability of w/h vs. size in optimization?
  - Compute many orderings with equal w/h and average the search space/AOMDD sizes.
Future Work?

- Try regressing curves that depend on $w$, $h$, or both
- Evaluation of bottom-up version of compilation (Robert’s algorithm in CP 2006)