Decision Diagrams for Constraint Programming

Willem-Jan van Hoeve
Tepper School of Business
Carnegie Mellon University
www.andrew.cmu.edu/user/vanhoeve/mdd/
Plan

What can MDDs do for Combinatorial Optimization?
- *Compact representation* of all solutions to a problem
- Limit on size gives *approximation*
- Control strength of approximation by size limit

MDDs for Constraint Programming and Scheduling
- MDD propagation natural generalization of domain propagation
- Orders of magnitude improvement possible

MDDs for Discrete Optimization
- MDD *relaxations* provide upper bounds
- MDD *restrictions* provide lower bounds
- New branch-and-bound scheme

Many Opportunities: integrated methods, theory, applications,...
Decision Diagrams

- Binary Decision Diagrams were introduced to compactly represent Boolean functions [Lee, 1959], [Akers, 1978], [Bryant, 1986]
- BDD: merge isomorphic subtrees of a given binary decision tree
- MDDs are multi-valued decision diagrams (i.e., for arbitrary finite-domain variables)

\[ f(x_1, x_2, x_3) = (\neg x_1 \land \neg x_2 \land \neg x_3) \lor (x_1 \land x_2) \lor (x_2 \land x_3) \]
**Brief background**

- Original application areas: circuit design, verification
- Usually *reduced ordered* BDDs/MDDs are applied
  - fixed variable ordering
  - minimal exact representation
- Recent interest from optimization community
  - cut generation [Becker et al., 2005]
  - 0/1 vertex and facet enumeration [Behle & Eisenbrand, 2007]
  - post-optimality analysis [Hadzic & Hooker, 2006, 2007]
  - set bounds propagation [Hawkins, Lagoon, Stuckey, 2005]
- Interesting variant
  - relaxed MDDs
    [Andersen, Hadzic, Hooker & Tiedemann, CP 2007]
Exact MDDs for discrete optimization

(1) $x_1 + x_2 + x_3 \geq 1$
(2) $x_1 + x_4 + x_5 \geq 1$
(3) $x_2 + x_4 \geq 1$
Exact MDDs for discrete optimization

(1) \( x_1 + x_2 + x_3 \geq 1 \)
(2) \( x_1 + x_4 + x_5 \geq 1 \)
(3) \( x_2 + x_4 \geq 1 \)
Exact MDDs for discrete optimization

\[
\begin{align*}
&x_1 + x_2 + x_3 \geq 1 \\
&x_1 + x_4 + x_5 \geq 1 \\
&x_2 + x_4 \geq 1
\end{align*}
\]
Exact MDDs for discrete optimization

---: 0
--: 1

(1) \(x_1 + x_2 + x_3 \geq 1\)
(2) \(x_1 + x_4 + x_5 \geq 1\)
(3) \(x_2 + x_4 \geq 1\)
Exact MDDs for discrete optimization

(1) \( x_1 + x_2 + x_3 \geq 1 \)
(2) \( x_1 + x_4 + x_5 \geq 1 \)
(3) \( x_2 + x_4 \geq 1 \)

Each path corresponds to a solution

\( (1,0,1,1,0) \)
Approximate MDDs

- Exact MDDs can be of exponential size in general
- Can we **limit the size** of the MDD and still have a meaningful representation?
  - Yes, first proposed by Andersen et al. [2007]: Limit the *width* of the MDD (the maximum number of nodes on any layer)

- Limited-width MDDs: main focus of this tutorial
**Suppressed Decision Diagrams**

- Zero-suppressed BDD (0-BDD or ZDD)
  - arc skips layers for which variables will take value 0
- One-suppressed BDD (1-BDD)
  - arc skips layers for which variables will take value 1
- Zero/one-suppressed BDD (0/1-BDD)
  - arc skips layers for which variables will take value 0/1

- Similarly suppressed MDDs can be defined
- Will not be discussed in detail, but methodology can be extended
MDDs for Constraint Programming
Motivation

Constraint Programming applies
• systematic search and
• inference techniques
to solve combinatorial problems

Inference mainly takes place through:
• Filtering provably inconsistent values from variable domains
• Propagating the updated domains to other constraints

\[
\begin{align*}
x_1 > x_2 \\
x_1 + x_2 &= x_3 \\
\text{alldifferent}(x_1, x_2, x_3, x_4)
\end{align*}
\]

\[
x_1 \in \{1, 2\}, \ x_2 \in \{0, 1, 2, 3\}, \ x_3 \in \{2, 3\}, \ x_4 \in \{0, 1\}
\]
Characterization of propagation

- Let $C(X)$ be a constraint on variables $X$. Let $D(x)$ denote the domain of possible values for $x$ in $X$.
- Constraint $C(X)$ is domain consistent if for each $x$ in $X$, each $v$ in $D(x)$ belongs to a solution to $C$.

\[
\begin{align*}
&x_1 \neq x_2 \\
&x_1 \neq x_3 \\
&x_2 \neq x_3
\end{align*}
\]

\[\text{alldifferent}(x_1, x_2, x_3)\]

- $x_1 \in \{2, 3\}$, $x_2 \in \{1, 2, 3\}$, $x_3 \in \{2, 3\}$

- *Establish domain consistency*: Remove all inconsistent values from the variable domains.
Illustrative example

\text{alldifferent}(x_1,x_2,x_3,x_4) \quad (1)
\begin{align*}
x_1 + x_2 + x_3 & \geq 9 \quad (2) \\
x_i & \in \{1,2,3,4\}
\end{align*}

List of all solutions to \text{alldifferent}: 

\begin{array}{cccc}
1 & 2 & 3 & 4 \\
1 & 2 & 4 & 3 \\
1 & 3 & 2 & 4 \\
\vdots \\
4 & 3 & 2 & 1
\end{array}

Suppose we could evaluate (2) on this list

projection: D(x_i) = \{1,2,3,4\}
**Illustrative example**

\[\text{alldifferent}(x_1, x_2, x_3, x_4) \quad (1)\]

\[x_1 + x_2 + x_3 \geq 9 \quad (2)\]

\[x_i \in \{1, 2, 3, 4\}\]

List of all solutions to \textit{alldifferent}: 

<table>
<thead>
<tr>
<th>( x_1 )</th>
<th>( x_2 )</th>
<th>( x_3 )</th>
<th>( x_4 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>✔️ 2</td>
<td>3</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>✔️ 2</td>
<td>4</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>✔️ 3</td>
<td>2</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>✔️ 4</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

...  


Suppose we could evaluate (2) on this list

projection: \( D(x_4) = \{1\} \)
Illustrative example (cont’d)

\[
\text{alldifferent}(x_1,x_2,x_3,x_4) \quad (1)
\]
\[
x_1 + x_2 + x_3 \geq 9 \quad (2)
\]
\[
x_i \in \{1,2,3,4\}
\]

List of all solutions: \text{use MDDs}

<table>
<thead>
<tr>
<th>x_1</th>
<th>x_2</th>
<th>x_3</th>
<th>x_4</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>3</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>2</td>
<td>4</td>
<td>3</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>
Motivation for MDD propagation

• All structural relationships among variables are projected onto the domains
• Potential solution space implicitly defined by Cartesian product of variable domains (very coarse relaxation)

We can communicate more information between constraint using MDDs [Andersen et al. 2007]

• Explicit representation of more refined potential solution space
• Limited width defines relaxed MDD
• Strength is controlled by the imposed width
MDD-based Constraint Programming

• Maintain limited-width MDD
  – Serves as relaxation
  – Typically start with width 1 (initial variable domains)
  – Dynamically adjust MDD, based on constraints

• Constraint Propagation
  – **Edge filtering**: Remove provably inconsistent edges (those that do not participate in any solution)
  – **Node refinement**: Split nodes to separate edge information

• Search
  – As in classical CP, but may now be guided by MDD
Domain consistency generalizes naturally to MDDs:

- Let \( C(X) \) be a constraint on variables \( X \) and let \( M \) be an MDD on \( X \)
- Constraint \( C \) is **MDD consistent** if for each arc in \( M \), there is at least one path in \( M \) that represents a solution to \( C \)

Equivalent to domain consistency for MDD of width 1
Specific MDD propagation algorithms

- Linear equalities and inequalities  
  [Hadzic et al., 2008]
  [Hoda et al., 2010]

- *Alldifferent* constraints  
  [Andersen et al., 2007]

- *Element* constraints  
  [Hoda et al., 2010]

- *Among* constraints  
  [Hoda et al., 2010]

- Disjunctive scheduling constraints  
  [Hoda et al., 2010]
  [Cire & v.H., 2011, 2013]

- *Sequence* constraints (combination of *Amongs*)  
  [Bergman et al., 2014]

- Generic re-application of existing domain filtering algorithm for any constraint type  
  [Hoda et al., 2010]
• For a given constraint type we maintain specific ‘state information’ at each node in the MDD

• Computed from incoming arcs (both from top and from bottom)

• State information is basis for MDD filtering and for MDD refinement
First example: Among constraints

- Given a set of variables $X$, and a set of values $S$, a lower bound $l$ and upper bound $u$,

$$Among(X, S, l, u) := l \leq \sum_{x \in X} (x \in S) \leq u$$

“among the variables in $X$, at least $l$ and at most $u$ take a value from the set $S$”

- Applications in, e.g., sequencing and scheduling
Example: Nurse Rostering

• Set of nurses N, who can work evening, day, or night shift, or can have a day off: \{e,d,n,o\}

• Rules: Each nurse works
  – at most 2 night shifts out of every 8 consecutive days,
  – at least 22 work shifts out of every 30 consecutive days,

• Planning horizon: 80 days

• For each day \(i\), we have fixed demand \(D_{i,s}\) for each shift \(s = e,d,n\).

• Goal: Find a solution that meets demand and satisfies all constraints.
CP model

- Variables $x_{i,n}$: shift of nurse $n$ on day $i$
  
  $$D(x_{i,n}) = \{e, d, n, o\}$$

- Constraints

  meet demand: $\sum_n (x_{i,n}=s) \geq D_{i,s}$ for all $i, s$

  at most $2/8$: $\text{Among}\{x_{i,n}, \ldots, x_{i+7,n}\}, \{n\}, 0, 2$
  
  for all $i=1, \ldots, 73$, and $n$

  at least $22/30$: $\text{Among}\{x_{i,n}, \ldots, x_{i+29,n}\}, \{e, d, n\}, 22, 30$
  
  for all $i=1, \ldots, 51$, and $n$
First example: Among constraints

- Given a set of variables $X$, and a set of values $S$, a lower bound $l$ and upper bound $u$,

$$\text{Among}(X, S, l, u) := l \leq \sum_{x \in X} (\ x \in S) \leq u$$

“among the variables in $X$, at least $l$ and at most $u$ take a value from the set $S$”

- Applications in, e.g., sequencing and scheduling
- WLOG assume here that $X$ are binary and $S = \{1\}$
- Let’s develop an MDD propagation algorithm
Example MDD for Among

Exact MDD for Among(\{x_1, x_2, x_3, x_4\},\{1\},2,2)

State information: path length from top and from bottom
Goal: Given an MDD and an *Among* constraint, remove all inconsistent edges from the MDD (establish MDD-consistency)

Approach:

- Compute path lengths from the root and from the sink to each node in the MDD
- Remove edges that are not on a path with length between lower and upper bound

[Hoda et al., CP 2010]
Example

State: T U
path lengths from top (T) and from bottom (U)

What happens if we only maintain bounds instead of all path lengths?

Among({x₁,x₂,x₃,x₄},{1},2,2)
MDD Filtering for Among

Goal: Given an MDD and an Among constraint, remove all inconsistent edges from the MDD (establish MDD-consistency)

Approach:

- Compute path lengths from the root and from the sink to each node in the MDD
- Remove edges that are not on a path with length between lower and upper bound

- Complete (MDD-consistent) version
  - Maintain all path lengths; quadratic time

- Partial version (may not remove all inconsistent edges)
  - Maintain and check bounds (longest and shortest paths); linear time
Node refinement for Among

For each layer in MDD, we first apply edge filter, and then try to refine:

- consider incoming edges for each node
- split the node if there exist incoming edges that are not equivalent (w.r.t. path length)
- in other words, need to identify equivalence classes

Example:

- We will propagate $\text{Among}({x_1,x_2,x_3,x_4},{1},2,2)$ through a BDD of maximum width 3
Example

Among\(\{x_1,x_2,x_3,x_4\},\{1\},2,2\)
Example

Among({x₁,x₂,x₃,x₄},{1},2,2)
Example

Among($\{x_1, x_2, x_3, x_4\}, \{1\}, 2, 2$)
Example

\[ \text{Among}({x_1, x_2, x_3, x_4}, \{1\}, 2, 2) \]
Experiments

• Multiple among constraints
  – 50 binary variables total
  – 5 variables per among constraint, indices chosen from normal distribution with uniform-random mean in [1..50] and stdev 2.5, modulo 50 (i.e., somewhat consecutive)
  – Classes: 5 to 200 among constraints (step 5), 100 instances per class

• Nurse rostering instances (horizon $n$ days)
  – Work 4-5 days per week
  – Max A days every B days
  – Min C days every D days
  – Three problem classes

• Compare width 1 (traditional domains) with increasing widths
Multiple Amongs: Backtracks

width 1 vs 4

width 1 vs 16
Multiple Amongs: Running Time

- width 1 vs 4
- width 1 vs 16
Nurse rostering problems

<table>
<thead>
<tr>
<th>Size</th>
<th>Width 1 BT</th>
<th>Width 1 CPU</th>
<th>Width 4 BT</th>
<th>Width 4 CPU</th>
<th>Width 32 BT</th>
<th>Width 32 CPU</th>
</tr>
</thead>
<tbody>
<tr>
<td>Class 1 40</td>
<td>61,225</td>
<td>55.63</td>
<td>8,138</td>
<td>12.64</td>
<td>3</td>
<td>0.09</td>
</tr>
<tr>
<td>80</td>
<td>175,175</td>
<td>442.29</td>
<td>5,025</td>
<td>44.63</td>
<td>11</td>
<td>0.72</td>
</tr>
<tr>
<td>Class 2 40</td>
<td>179,743</td>
<td>173.45</td>
<td>17,923</td>
<td>32.59</td>
<td>4</td>
<td>0.07</td>
</tr>
<tr>
<td>80</td>
<td>179,743</td>
<td>459.01</td>
<td>8,747</td>
<td>80.62</td>
<td>2</td>
<td>0.32</td>
</tr>
<tr>
<td>Class 3 40</td>
<td>91,141</td>
<td>84.43</td>
<td>5,148</td>
<td>9.11</td>
<td>7</td>
<td>0.18</td>
</tr>
<tr>
<td>80</td>
<td>882,640</td>
<td>2,391.01</td>
<td>33,379</td>
<td>235.17</td>
<td>55</td>
<td>3.27</td>
</tr>
</tbody>
</table>
Exercises

1. Consider the constraint $x \neq y$ for two finite-domain variables $x$ and $y$. Assume that $x$ and $y$ belong to a set $X$ of variables for which we are given a relaxed MDD. Design an MDD propagator for $x \neq y$.

2. Consider the following CSP:

   $x_1 \in \{0,1\}$, $x_2 \in \{0,1,2\}$, $x_3 \in \{1,2\}$

   $x_1 \neq x_2$, $x_2 \neq x_3$, $x_1 \neq x_3$

   Apply filtering and refinement (using the propagator from Exercise 1), starting from a width-1 MDD, until the MDD represents all solutions to the CSP.
More exercises

3. Design an MDD propagator for the constraint

\[ \sum_{i=1}^{n} c_i x_i \leq b \]

where \( c_i, b \) are constants and \( x_i \) are finite-domain variables. Can we establish MDD consistency in polynomial time for this constraint (for an arbitrary MDD defined on \( x_i \))?

4. Suppose we have a system of linear constraints:

\[ \sum_{i=1}^{n} c_{ij} x_i \leq b_j \quad \text{for } j = 1, \ldots, m \]

How would you use the propagator from exercise 3 to handle this system?