Outline

- Disjunctive Scheduling
- MDD representation
- Filtering and precedence relations
- Experimental results
- Conclusion
Disjunctive Scheduling

- Sequencing and scheduling of activities on a resource

**Activities**
- Processing time: $p_i$
- Release time: $r_i$
- Deadline: $d_i$

**Resource**
- Nonpreemptive
- Process one activity at a time
Extensions

- Precedence relations between activities
- Sequence-dependent setup times

Variety of objective functions
- Makespan
- Sum of setup-times
- Tardiness / number of late jobs
- …
Constraint-Based Scheduling

- Inference for disjunctive scheduling
  - Precedence relations
  - Time intervals that an activity can be processed

- Sophisticated techniques include:
  - Edge-Finding
  - Not-first / not-last rules

Examples:
- $1 \ll 3$
- $s_3 \geq 3$
Constraint-Based Scheduling

- Extensible, flexible scheduling systems
  - Successful in many real-world applications

- Challenges arise in presence of
  - Sequence-dependent setup times
  - Complex objective functions

- New inference techniques based on Multivalued Decision Diagrams to tackle these challenges
Multivalued Decision Diagrams

\[ x_1 + x_2 \leq 1, \]
\[ x_1 \neq x_2, x_1 \neq x_3, x_2 \neq x_3, \]
\[ x_1, x_2, x_3 \in \{0,1,2,3\}. \]

- **Ordered Acyclic Digraph**
  - *Layers*: variables
  - *Arc labels*: variable assignments
- **Paths from** \( r \) **to** \( t \): feasible solutions
- **Compact** representation of the search tree for a problem.
Multivalued Decision Diagrams

- Consider any separable objective function, e.g.
  \[ f(x) = 2x_1 + 3x_2 + (x_3)^3 \]

- Appropriate arc weights: shortest path minimizes \( f(x) \)
Consider any separable objective function, e.g.

\[ f(x) = 2x_1 + 3^{x_2} + (x_3)^3 \]

Appropriate arc weights: shortest path minimizes \( f(x) \)
MDD for Disjunctive Scheduling

- Every solution can be written as a permutation $\pi$

$$\begin{array}{cccc}
\text{Act} & r_i & d_i & p_i \\
1 & 0 & 3 & 2 \\
2 & 4 & 9 & 2 \\
3 & 3 & 8 & 3 \\
\end{array}$$

Path $\{1\} - \{3\} - \{2\}$

$$\begin{align*}
0 & \leq \text{start}_1 \leq 1 \\
6 & \leq \text{start}_2 \leq 7 \\
3 & \leq \text{start}_3 \leq 5 \\
\end{align*}$$
Permutation Model

Our two main considerations:

- Compilation
  - How to translate a disjunctive instance to an MDD

- Inference techniques
  - Types of inference we can obtain from MDD
Theorem: Constructing the exact MDD for a Disjunctive Instance is an NP-Hard problem

Nevertheless, some interesting restrictions, e.g. (Balas [99]):

- TSP defined on a complete graph
- Given a fixed parameter $k$, we must satisfy

\[ i \ll j \quad \text{if} \quad j - i \geq k \quad \text{for cities } i, j \]

Corollary: The exact MDD for the TSP above has $O(n2^k)$ nodes
Compilation

- Even in restricted cases, MDDs can grow exponentially

- We are still interested in general cases for inference purposes

- Alternative: **Relaxed MDDs**
  - Limit on the width of the graph
  - *Filter and Refinement* [Andersen et al. CP2007], [Hoda et al. CP2010]
Filter and Refinement

- **Start with a relaxed MDD**
  - Contains all feasible paths

- **Filter infeasible arc values**
  - Top-down/Bottom-up passes
Filter and Refinement

- **Start with a relaxed MDD**
  - Contains all feasible paths

- **Filter infeasible arc values**
  - Top-down/Bottom-up passes

- **Refinement**
  - Add nodes to improve relaxation
  - Usually heuristics
Filter: Top-Down Example

- Filter based on a state information at each node

Example:
Filtering arc (u,v)
Filter: Top-Down Example

- **All-paths state:** $A_u$
  - Labels belonging to all paths from node $r$ to node $u$
  - $A_u = \{3\}$
  - Thus eliminate $\{3\}$ from $(u,v)$

- Introduced for *Alldifferent* constraint in [Andersen et al 2007])
**Filter: Top-Down Example**

- **Some-paths state:** \( S_u \)
  - Labels belonging to some path from node \( r \) to node \( u \)
  - \( S_u = \{1,2,3\} \)
  - Identification of Hall sets
  - Thus eliminate \( \{1,2,3\} \) from \( (u,v) \)

- Introduced for Alldifferent constraint in [Andersen et al 2007])
Filter: Top-Down Example

- **Earliest Completion Time:** $E_u$
  - Minimum completion time of all paths from root to node $u$

- **Similarly: Latest Completion Time**
Filter: Top-Down Example

<table>
<thead>
<tr>
<th>Act</th>
<th>$r_i$</th>
<th>$d_i$</th>
<th>$p_i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0</td>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>2</td>
<td>3</td>
<td>7</td>
<td>3</td>
</tr>
<tr>
<td>3</td>
<td>1</td>
<td>8</td>
<td>3</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
<td>6</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>2</td>
<td>10</td>
<td>3</td>
</tr>
</tbody>
</table>

- $E_u = 7$
- Eliminate 4 from $(u,v)$
MDDs and Precedence Relations

**Theorem:** Given the exact MDD $M$, we can deduce all implied precedences in polynomial time in the size of $M$

- For a node $v$,
  - $A_{v}^{\downarrow}$: all-paths from root to $v$
  - $A_{v}^{\uparrow}$: all-paths from terminal to $v$

- Precedence relation $i \ll j$ holds if and only if $(j \not\in A_{u}^{\downarrow})$ or $(i \not\in A_{u}^{\uparrow})$ for all nodes $u$ in $M$

- Same technique applies to relaxed MDD
Communicate Precedence Relations

1. Provide precedences inferred from the MDD to CP
   - Update time variables
   - Other inference techniques may utilize them

2. We can filter the relaxed MDD using precedence relations inferred from other (CP) techniques
   - Precedences deduced by this method might not be dominated by other techniques, even for small widths.
Experimental Results

- Implemented in *Ilog CP Optimizer (CPO)*
  - State-of-the-art constraint based scheduling solver
  - Uses a portfolio of inference techniques and LP relaxation

- Two versions considered
  - Standalone MDD
  - *Ilog CPO + MDD* (but *partial* integration!)

- Instances from TSP with Time Windows
  - minimize sum of setup times / minimize makespan
<table>
<thead>
<tr>
<th>Instance</th>
<th>CPO</th>
<th>MDD</th>
<th>CPO+MDD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backtracks</td>
<td>Time</td>
<td>Backtracks</td>
<td>Time</td>
</tr>
<tr>
<td>n20w100.002</td>
<td>1,382,397</td>
<td>95.71</td>
<td>190,101</td>
</tr>
<tr>
<td>n20w60.004</td>
<td>151,301</td>
<td>15.41</td>
<td>85,245</td>
</tr>
<tr>
<td>n20w80.001</td>
<td>19,060</td>
<td>1.31</td>
<td>5,076</td>
</tr>
<tr>
<td>n20w80.005</td>
<td>61,823</td>
<td>5.46</td>
<td>22,369</td>
</tr>
<tr>
<td>n40w40.001</td>
<td>210,682</td>
<td>26.53</td>
<td>22,367</td>
</tr>
<tr>
<td>n40w40.003</td>
<td>152,855</td>
<td>14.71</td>
<td>27,483</td>
</tr>
<tr>
<td>n40w40.004</td>
<td>480,970</td>
<td>50.81</td>
<td>28,334</td>
</tr>
<tr>
<td>n60w20.001</td>
<td>908,606</td>
<td>199.26</td>
<td>31,182</td>
</tr>
<tr>
<td>n60w20.002</td>
<td>84,074</td>
<td>14.13</td>
<td>1,657</td>
</tr>
<tr>
<td>n60w20.003</td>
<td>22,296,012</td>
<td>+∞</td>
<td>134,755</td>
</tr>
<tr>
<td>n60w20.004</td>
<td>2,685,255</td>
<td>408.34</td>
<td>5,855</td>
</tr>
<tr>
<td>n60w20.005</td>
<td>19,520</td>
<td>9.32</td>
<td>2,580</td>
</tr>
</tbody>
</table>

minimize sum of setup times  
MDDs have maximum width 16
Combined CP+MDD

85 instances from Dumas and Ascheuer (AFG)
MDDs have maximum width 16
Dynamic search strategy
Conclusions

- **The Permutation Model**
  - Natural MDD representation
  - Strong relation to precedence graph
  - High-level communication between MDD and other inference mechanisms

- **Practical perspective**
  - Easy to implement in current constraint solvers
  - Observed orders of magnitude improvement