MDD Propagation for Disjunctive Scheduling

INFORMS Optimization Society Conference 2012

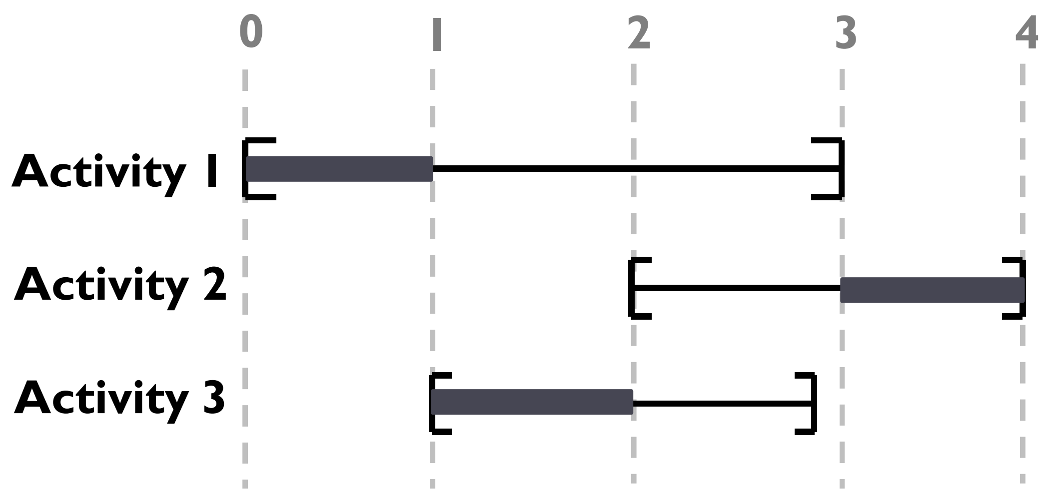
Andre Augusto Cire

Joint work with Dr. Willem-Jan van Hoeve

Tepper School of Business, Carnegie Mellon University

Disjunctive Scheduling

- Sequencing and scheduling of **activities** in 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
  - …

Current Literature

- Active research spread across communities
  - Operations Research
  - Artificial Intelligence

- Our focus: Constraint-based Scheduling

Constraint-Based Scheduling

- Constraints in a model capture richer structures, e.g.

disjunctive(s, p)

which enforces

$$\left(s_i + p_i \leq s_j\right) \vee \left(s_j + p_j \leq s_i\right), \text{ for all } i, j, \; i \neq j$$

- Specialized inference techniques for each constraint
- Separation between *model* and *solution approach*

Constraint-Based Scheduling

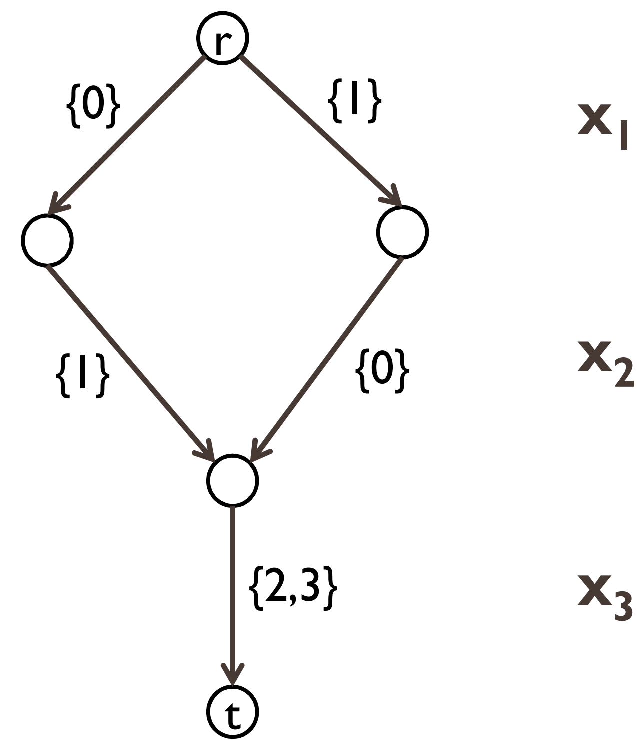
- Inference for disjunctive scheduling
  - Precedence relations
  - Time intervals that an activity can be processed
- Sophisticated techniques include:
  - Edge-Finding
  - Not-first / not-last rules

Constraint-Based Scheduling

- Extensible, flexible scheduling systems
  - Successful in many real-world applications
- Well-known deficiencies
  - Sequence-dependent setup times
  - Complex objective functions
- New inference techniques based on Multivalued Decision Diagrams to tackle these deficiencies

Multivalued Decision Diagrams

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



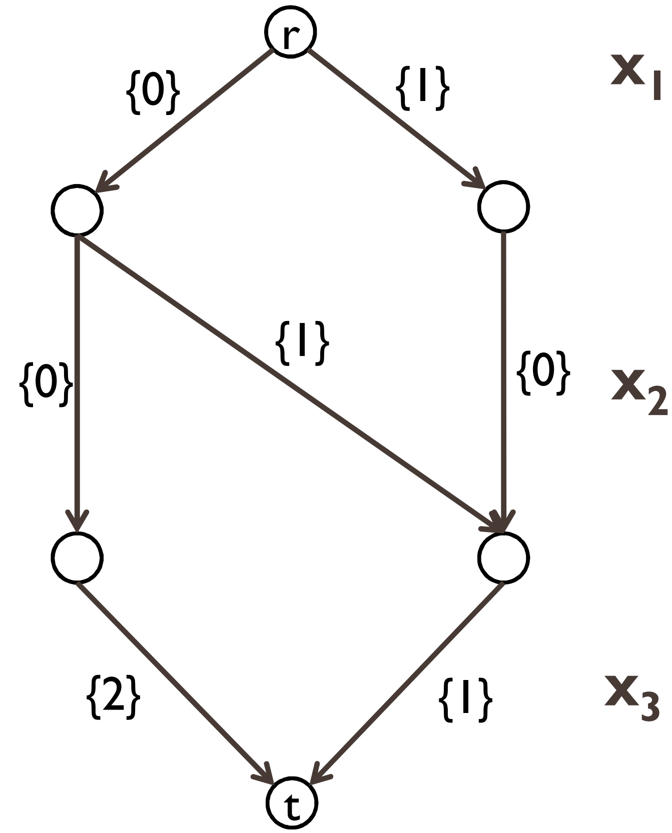
- Ordered Acyclic Digraph
  - *Layers:* variables
  - *Arc labels:* variable assignments
- Paths from **r** to **t**: feasible solutions
- Compact representation of the search tree for a problem.

Multivalued Decision Diagrams

- Consider any separable objective function, e.g.

$$f(x) = 2x_1 + 3^{x_2} + (x_3)^3$$

- Appropriate arc weights: shortest path minimizes f(x)

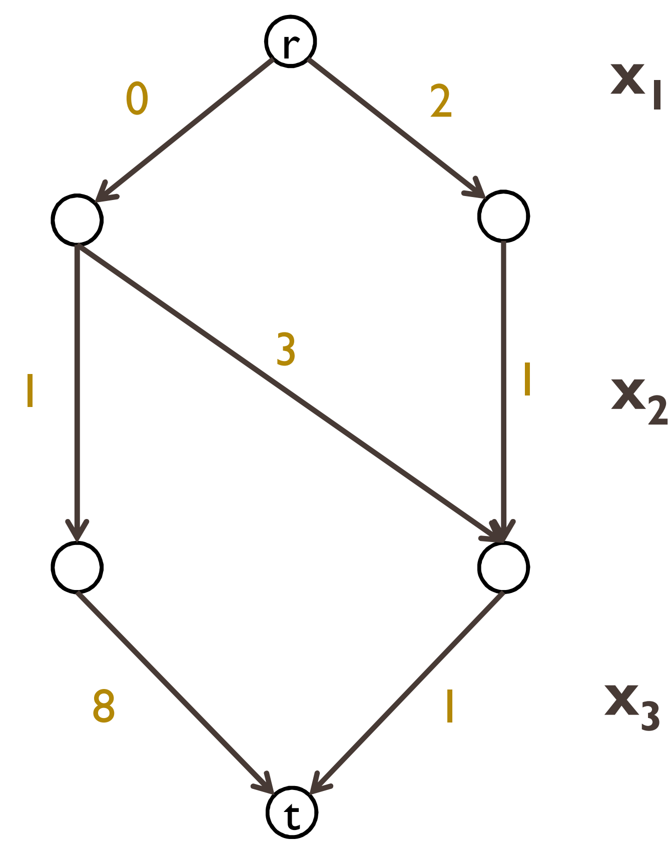


Multivalued Decision Diagrams

- 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)



Disjunctive Scheduling

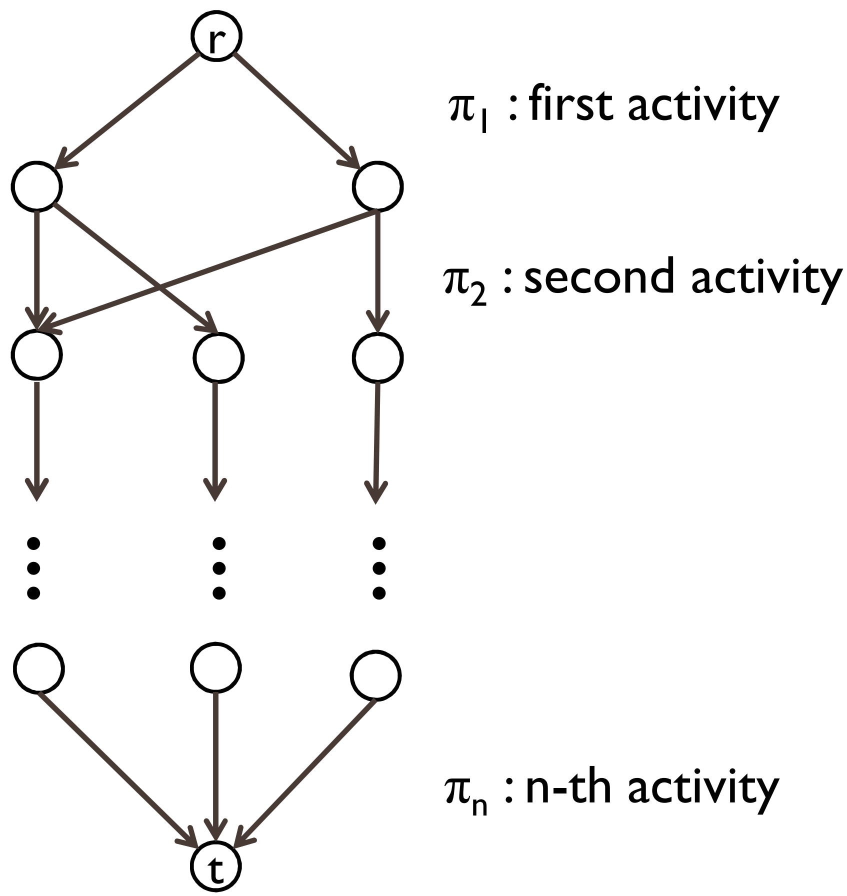
- Natural representation as MDDs
- Every solution can be written as a permutation **π**

  $\pi_1, \pi_2, \pi_3, \ldots, \pi_n$: activity sequencing in the machine

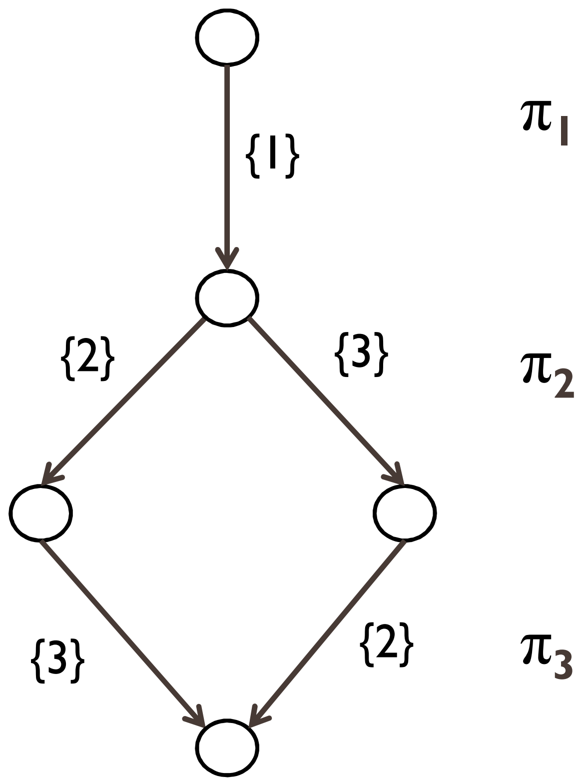
- Schedule is *implied* by a sequence, e.g.:

$$start_{\pi_i} \geq start_{\pi_{i-1}} + p_{\pi_{i-1}} \qquad i = 2, \ldots, n$$

Permutation Model



Example



Act	r_i	d_i	p_i
1	0	**3**	2
2	**4**	9	2
3	**3**	8	3

Path {1} – {3} – {2}

$0 \leq start_1 \leq 1$
$6 \leq start_2 \leq 7$
$3 \leq start_3 \leq 5$

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

Compilation

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

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

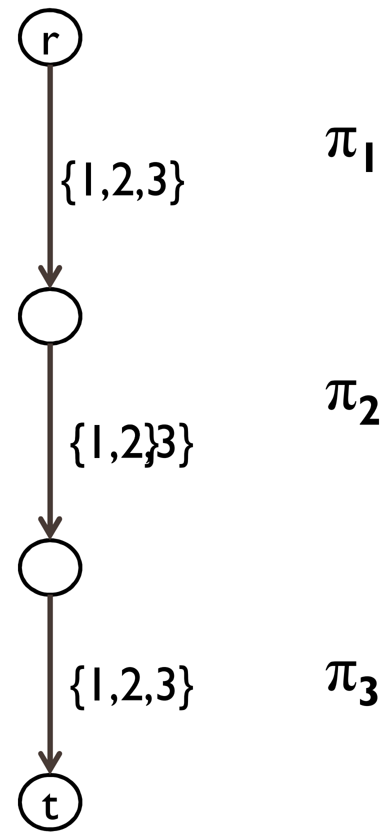
Corollary: *The exact MDD for the TSP above has $O(n2^k)$ nodes*

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 2007, Hoda et al 2010]

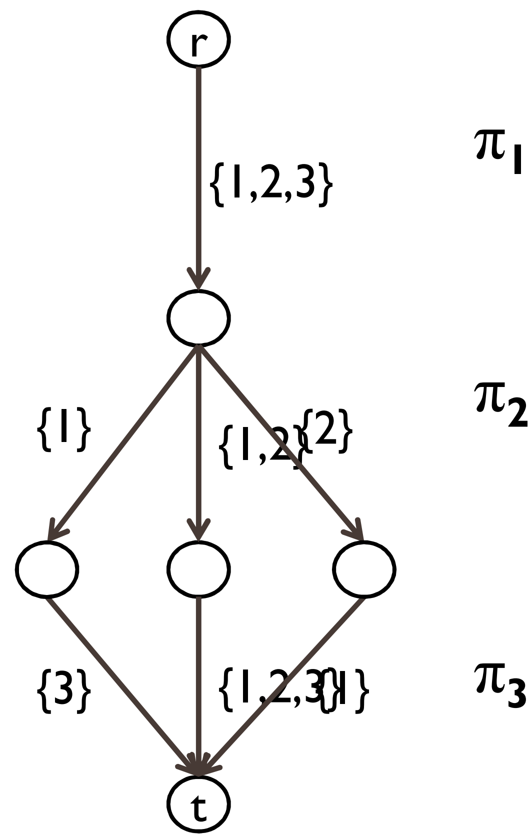
Filter and Refinement

- Start with a *relaxed* MDD
  - Contains all feasible paths
- Filter infeasible arc values
  - Top-down/Bottom-up passes



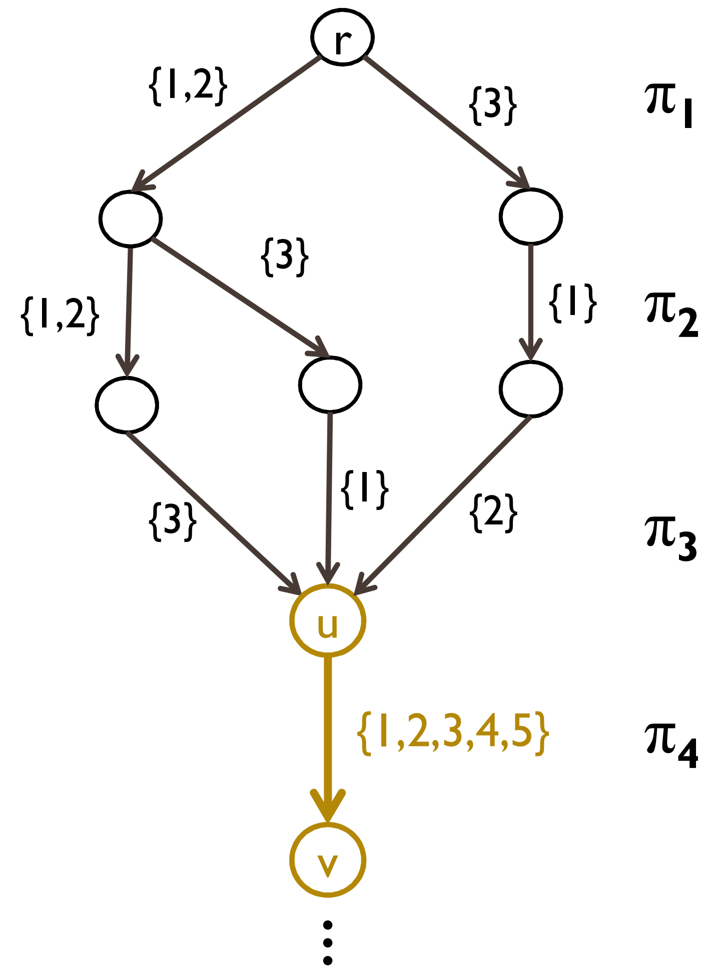
Filter and Refinement

- 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



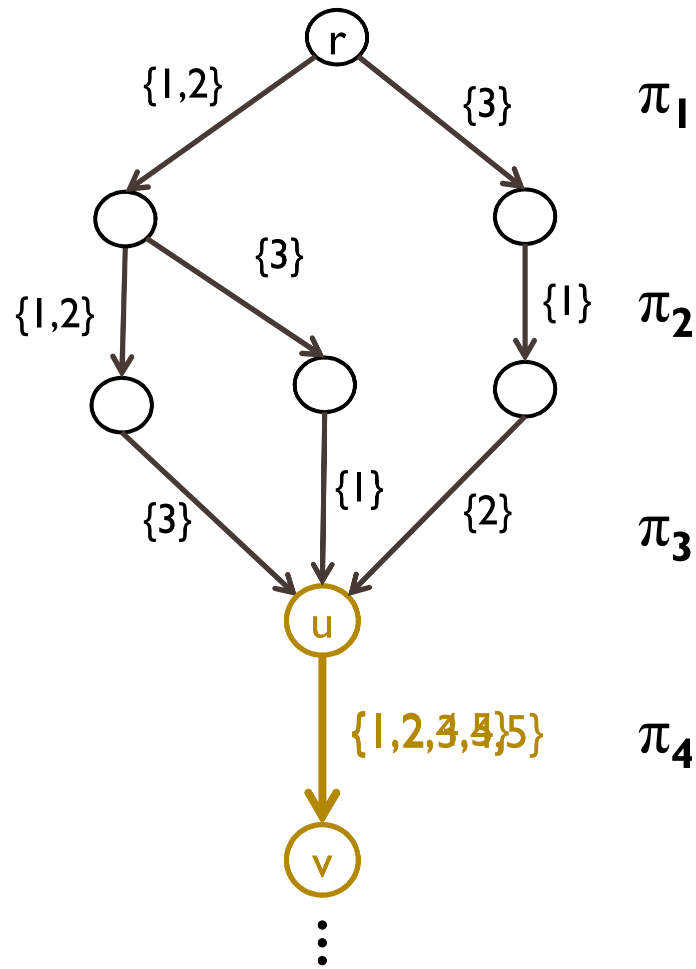
Filter: Top-Down Example

- Filter based on a *state* information at each node
- Ideal states
  - Compact
  - Markovian property
- Example:
  Filtering arc (u,v)



Filter: Top-Down Example

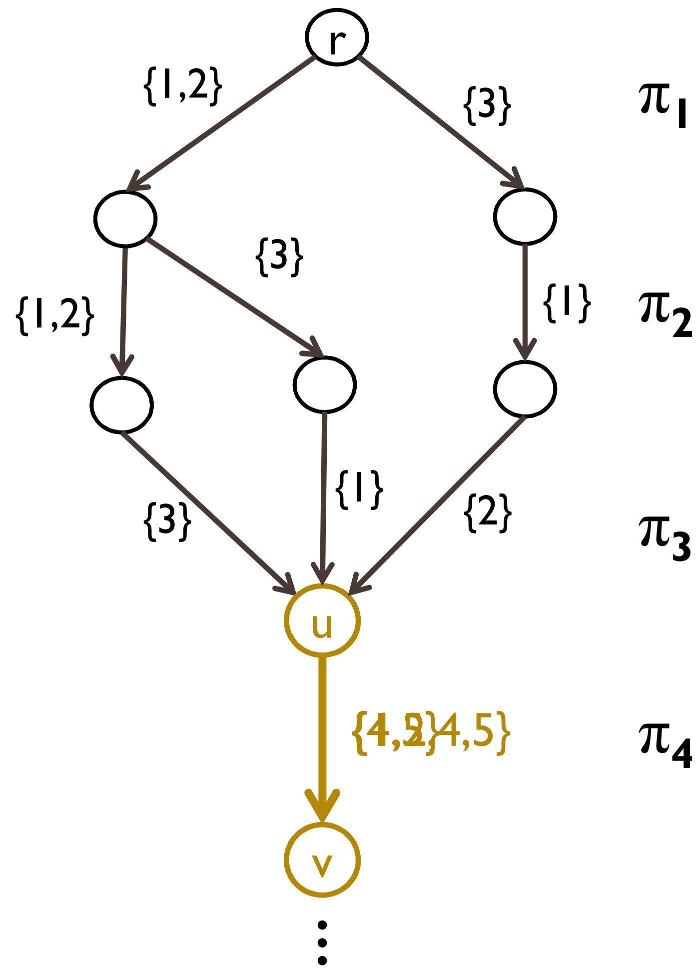
- *All-paths state:* A_u
  - *Labels belonging to all paths from node r to node u*
  - $A_u = \{3\}$
  - *Thus eliminate {3} from (u,v)*



- 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)*

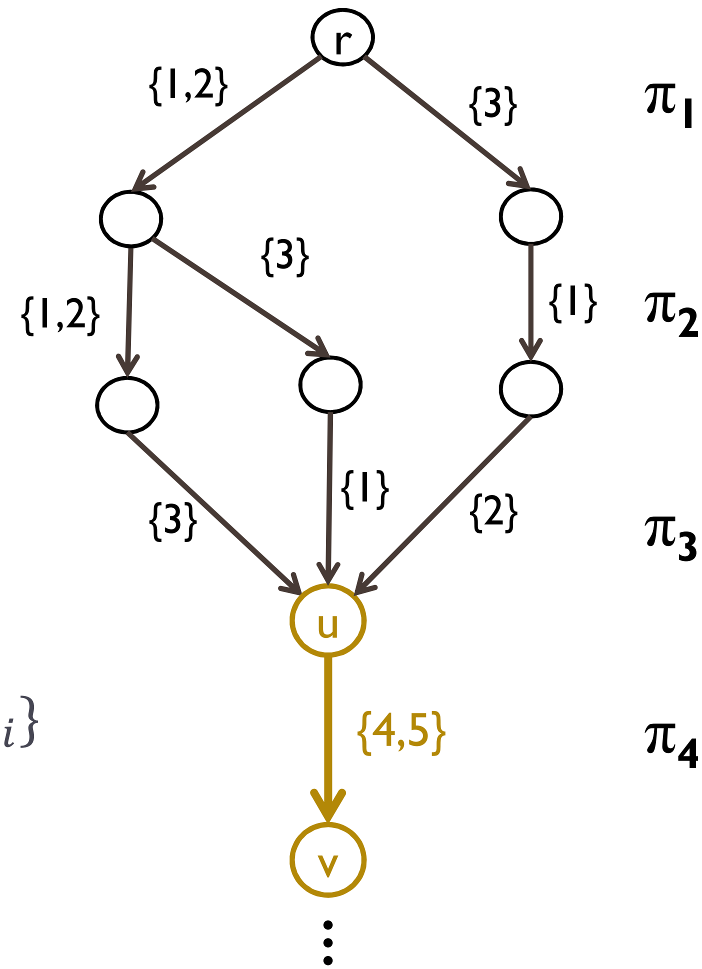


- Introduced for *Alldifferent* constraint in [Andersen et al 2007])

Filter: Top-Down Example

- *Earliest Completion Time:* E_u
  - *Minimum completion time of all paths from root to node u*
  - *Eliminate {i} from (u,v) if*

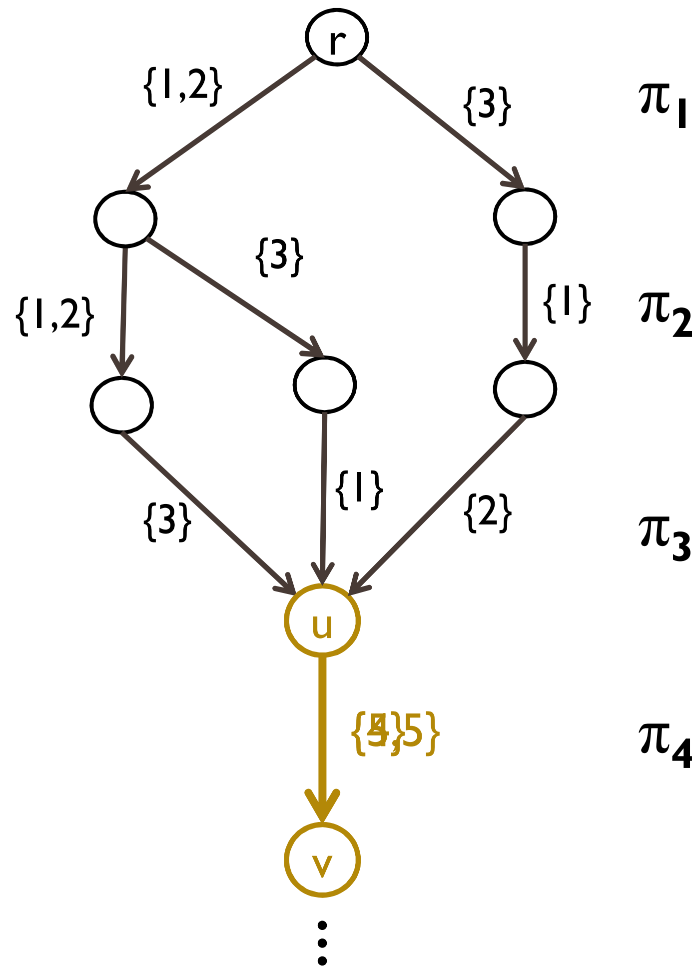
$$d_i < \max\{r_i, E_u\} + p_i + \min_{j \in \delta^-(i)} \{setup_{j,i}\}$$



Filter: Top-Down Example

Act	r_i	d_i	p_i
1	0	3	2
2	3	7	3
3	1	8	3
4	**5**	**6**	**1**
5	2	10	3

- $E_u = 7$
- *Eliminate 4 from (u,v)*



MDDs and the Precedence Graph

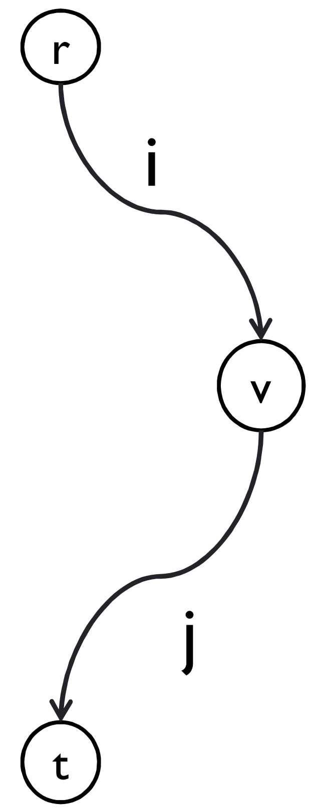
- Assume we have the exact MDD for a given instance

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

- *There exists a solution such that*

$$i \ll j$$

iff $i \in A_v^{\downarrow}$ *and* $j \in A_v^{\uparrow}$ *for some v*



MDDs and the Precedence Graph

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

- The "some path" states S_u are a relaxation of A_u
  - Theorem above is directly applied to a relaxed MDD
- A *Precedence Store* can be used to communicate information between traditional inference techniques and the relaxed MDD

MDDs and the Precedence Graph

1. We can deduce precedences from the relaxed MDD
   - Update time variables
   - Provide precedences to other inference techniques

2. We can *filter* the relaxed MDD using precedence relations inferred from other 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 scheduler solver
  - Uses a portfolio of inference techniques
  - Linear Relaxation

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

- Tests on many variations on disjunctive problems
  - Focus here on TSP with Time Windows

Instances Dumas – Standalone MDD

	CPO		MDD Width 16	
Instance	Backtracks	Time	Backtracks	Time
n20w100.002	1,382,397	95.71	190,101	76.41
n20w60.004	151,301	15.41	85,245	26.65
n20w80.001	19,060	1.31	5,076	1.15
n20w80.005	61,823	5.46	22,369	8.76
n40w40.001	210,682	26.53	22,367	7.33
n40w40.003	152,855	14.71	27,483	20.92
n40w40.004	480,970	50.81	28,334	10.34
n60w20.001	908,606	199.26	31,182	10.1
n60w20.002	84,074	14.13	1,657	0.14
n60w20.003	22,296,012	$+\infty$	134,755	105.85
n60w20.004	2,685,255	408.34	5,855	3.78
n60w20.005	19,520	9.32	2,580	0.33

Instances Dumas – CPO+MDD

	CPO		CPO+MDD Width 16	
Instance	Backtracks	Time	Backtracks	Time
n20w100.002	1,382,397	95.71	131,039	59.58
n20w60.004	151,301	15.41	21,743	7.81
n20w80.001	19,060	1.31	1,073	0.2
n20w80.005	61,823	5.46	7,638	3
n40w40.001	210,682	26.53	6,142	2.91
n40w40.003	152,855	14.71	800	0.14
n40w40.004	480,970	50.81	5,986	3.64
n60w20.001	908,606	199.26	17,637	7.46
n60w20.002	84,074	14.13	728	0.12
n60w20.003	22,296,012	+∞	55,311	39.43
n60w20.004	2,685,255	408.34	1,567	0.94
n60w20.005	19,520	9.32	1,039	0.08

Conclusions

- The Permutation Model
  - 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

Thank you!