Exercise 1: Solution

Assume wlog that x is ordered higher than y in the MDD. For each node u in the MDD, we maintain the state information:

- $S_u^{\downarrow}$: the set of all values that x takes on some r-u path
- $S_u^{\uparrow}$: the set of all values that y takes on some u-t paths

  (these need to be maintained only for nodes between the layers representing x and y)

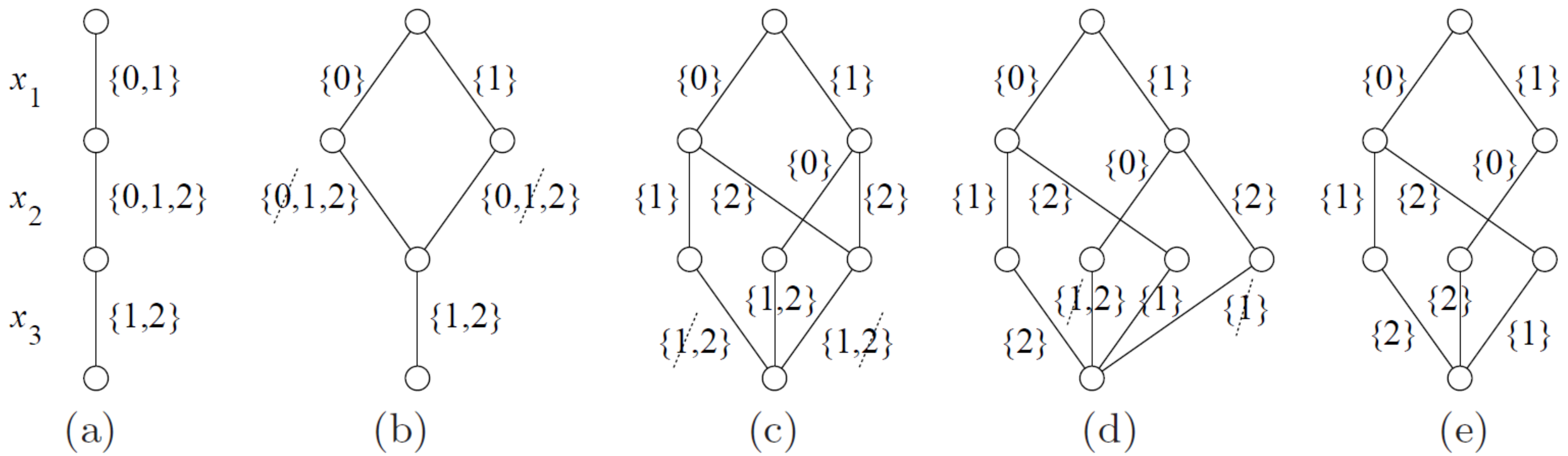
- $S_u^{\downarrow}$ is computed as the union of $S_v^{\downarrow}$ for all arcs (v,u) in the MDD (likewise for $S_u^{\uparrow}$)
- For each node u in the layer representing y for which $S_u^{\downarrow}$ is a singleton {e}, we delete any arc out of u with label e.
- Similarly for the layer representing x, we do this if $S_u^{\uparrow}$ is a singleton.

(See [Hoda et al, CP 2010])

Exercise 2 : Solution

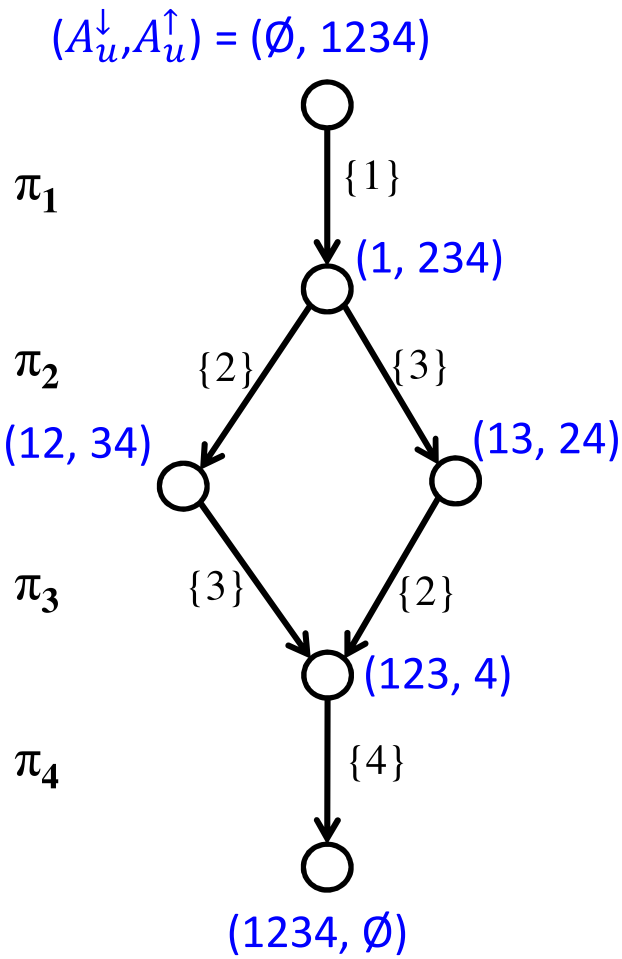
$x_1 \in \{0,1\}$, $x_2 \in \{0,1,2\}$, $x_3 \in \{1,2\}$

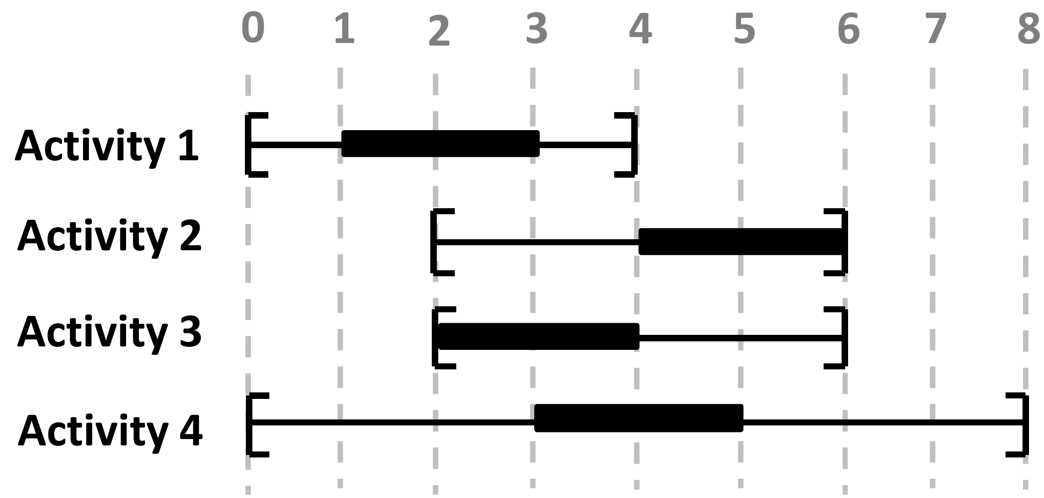
$x_1 \neq x_2$, $x_2 \neq x_3$, $x_1 \neq x_3$

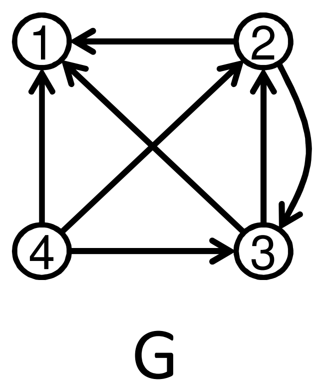


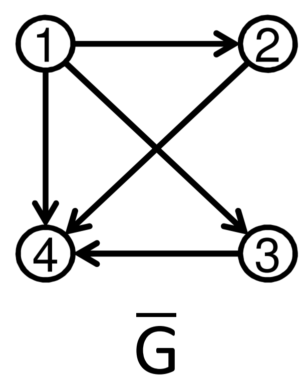
Refining and filtering an MDD of width one (a) for $x_1 \neq x_2$ (b), $x_2 \neq x_3$ (c), and $x_1 \neq x_3$ (d), yielding the MDD in (e). Dashed lines mark filtered values.

Exercise 3: Solution









$1 \ll 2$
$1 \ll 3$
$1 \ll 4$
$2 \ll 4$
$3 \ll 4$

Exercise 4: Solution

a) For ‘makespan’, the earliest completion time of the terminal is equivalent to the minimum makespan, by construction. The optimal permutation can be recursively retrieved from t up to r.

For ‘sum of setup times’, we let the cost of edge with label j be the minimum setup time t_{ij} for all incoming edges with label i. A shortest r-t path then corresponds to an optimal solution.

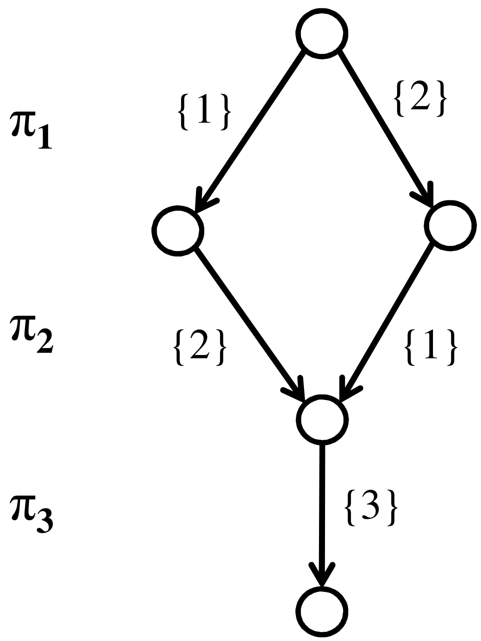
b) Similarly, for ‘tardiness’ a natural choice for the cost of an edge *a* is max{ 0, ect_a - δ_a }. However, this provides a lower bound, since the true cost of an edge depends on the total ordering. (Hence, in case the MDD is a single path, it does reflect the total tardiness.) See next slide for an example.

See [Cire & v.H., 2013] page 6/7

Exercise 4: Solution

Act	r_i	δ_i	p_i
1	0	10	5
2	2	5	3
3	0	10	2

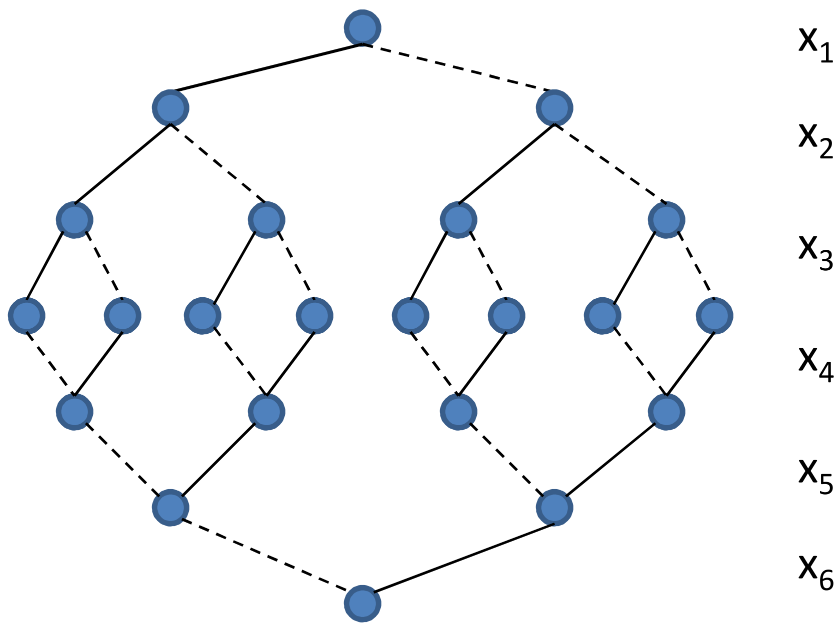
precedences: $1 \ll 3$
$2 \ll 3$



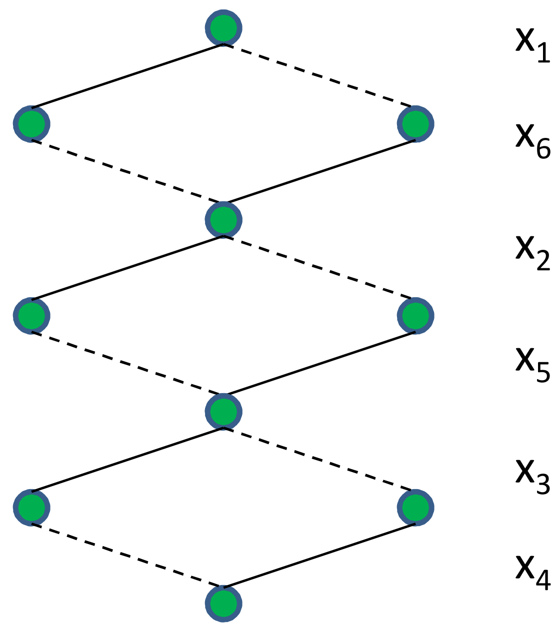
Shortest path {2} – {1} – {3} has cost 0 when edge cost is based on earliest completion time. Yet, the solution it represents has total tardiness 2.

Exercise 5: Solution

$4x_1 + 2x_2 + x_3 + x_4 + 2x_5 + 4x_6 = 7$



$4x_1 + 4x_6 + 2x_2 + 2x_5 + x_3 + x_4 = 7$



Exercise 6: Solution

min $3x_1 + 2x_2 + x_3 + 4x_4 + 2x_5$

s.t.

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

BDD construction choices:

- lexicographic ordering
- node deletion (instead of merging) rule: delete maximum shortest path, unless all constraints are covered by node
- filter 0-edge for last covering variable

Notes:

- An exact 0-BBD of width 3 exists
- For node *merging*, need to take maximum cost of merged nodes