# The ILP approach to the layered graph drawing

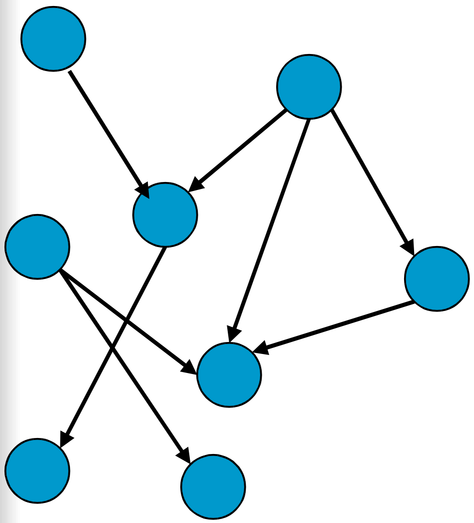
Ago Kuusik

**Veskisilla Teooriapäevad 1-3.10.2004**

# Outline

- Introduction
- Hierarchical drawing & Sugiyama algorithm
- Linear Programming (LP) and Integer Linear Programming (ILP)
- Multi-level crossing minimisation ILP
- Maximum level planar subgraph ILP

# Graphs



Set of **vertices** V
(real world: entities)

Set of **edges** E
::= **pairs** of vertices
(real world: relations)

**Directed graph**:
E ::= set of **ordered** pairs of vertices

# Graph drawing

- Started to grow in 1960s, aim – software understanding
- Now, used in number of areas, for example
  - Software engineering (call graphs, class diagrams, database schemas)
  - Social studies (relationship diagrams)
  - Chemistry (molecular structures)

# Graph drawing

Definition:

Given a graph $G=(V, E)$, represent the graph on a **plane**:

- Vertices – **closed shapes**
- Edges – **Jordan curves** between vertex shapes

(Jordan curve = a closed curve that does not intersect itself)

# Aesthetic criteria

- General
  - Min. number of edge crossings
  - Uniform edge direction (directed graph)
  - Min. number of edge bends
  - Min. area
- Application-specific
  - Specific vertex shapes (E-R diagram)
  - Specific vertex locations (class hierarchy)

# Hierachical drawing

Given: a directed acyclic graph

(A cyclic graph can be converted acyclic by reversing some edges; minimum feedback arc set is NP hard)

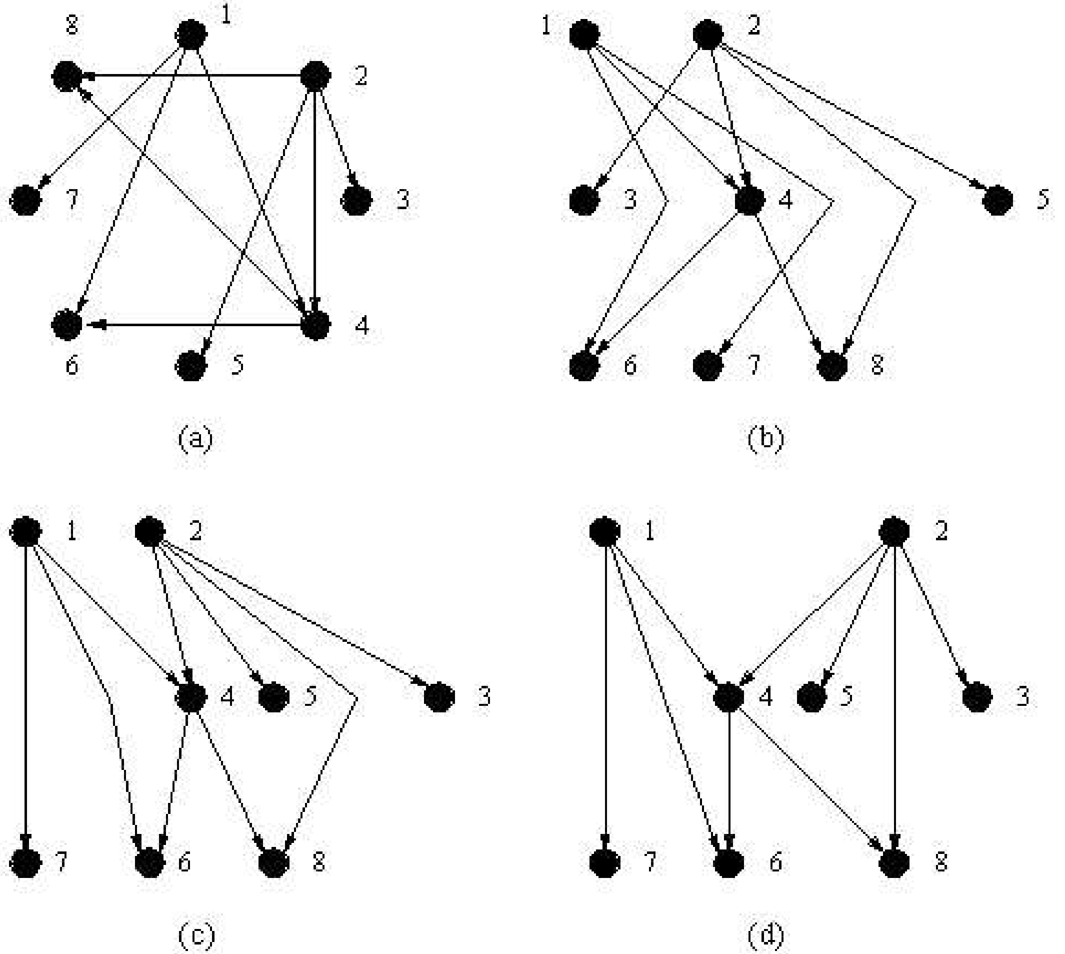
- Objective:
  - Uniform edge direction
  - Min. Number of edge crossings

# Sugiyama algorithm

- Published by Sugiyama, Tagawa, Toda 1981
- Vertices are placed on discrete layers
- Edges have uniform direction
- Edges connect vertices of adjacent layers
- Reduced edge crossings
- Overall balance of vertex locations

0. random layout

1. layering




2. sorting on layers

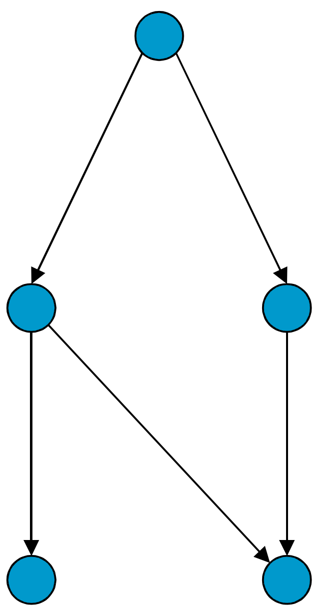
3. final positioning

# 1. Layering

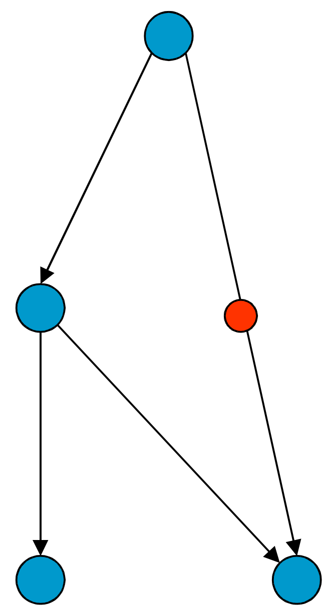
- Assign vertices to discrete layers so that the edges point to common direction
  - Longest path layering, shortest path layering: simple DFS algorithms
  - Coffman-Graham layering – constrains the width of the drawing
  - ILP approaches (Nikolov, 2002)

# Proper and non-proper layering

Proper



Non-Proper



# 2. Sorting of vertices on layers

- A combinatorial problem of re-ordering a set instead of a geometric placement problem
- Objective: min. edge crossings
- NP-hard even for 2 layers (Eades et al, 1986)
- Heuristics: barycenter, median, stochastic
- **ILP approach**

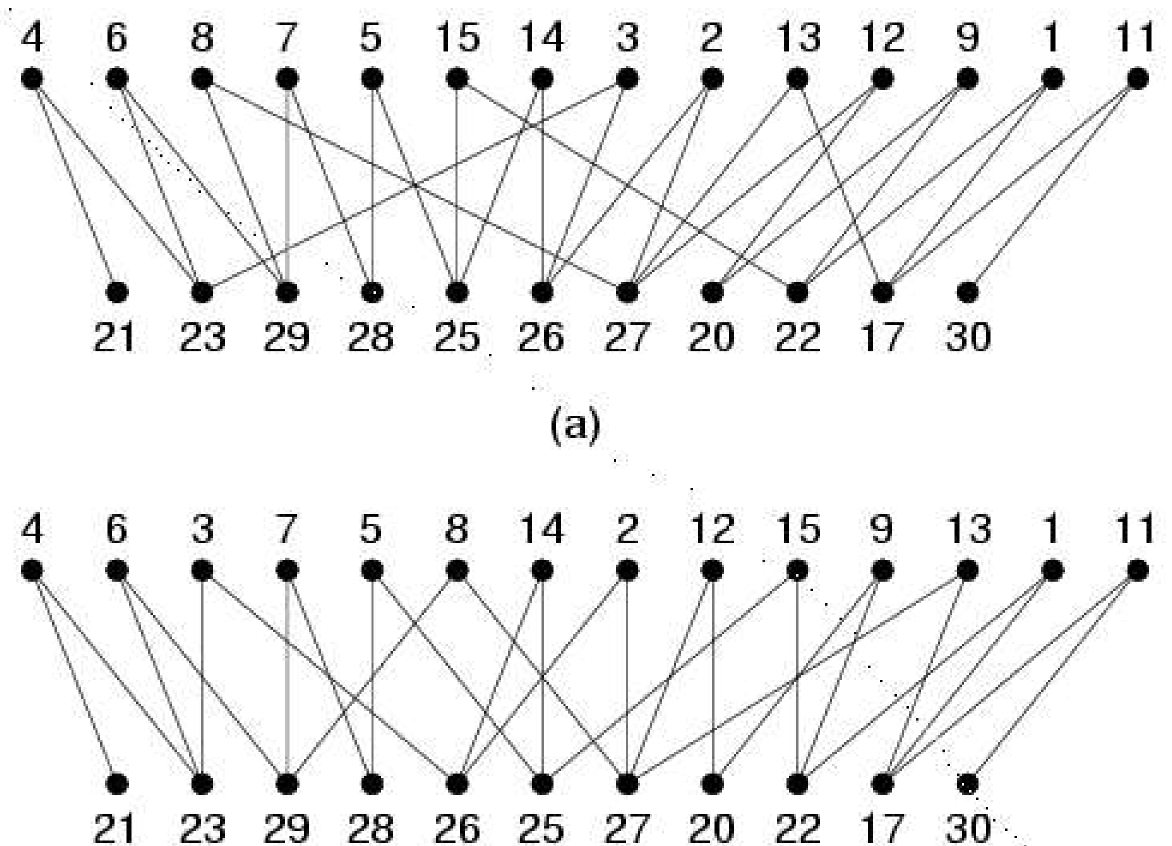
# 3. Positioning of vertices

- Objective: balanced positioning, reduction of edge bends
- Algorithms:
  - Linear Programming method (Sugiyama et al., 1981)
  - Pendulum heuristic (Sander, 1995)

# Our improvement to Sugiyama algorithm

- We aim to improve the 2nd step: **reordering of vertices**:
  - Find the optimal solution or a solution with guaranteed quality
  - Optimise accross all layers
  - Consider visualising the maximum level planar subgraph as an alternative to crossing minimisation

# Min. crossings vs max. planar subgraph



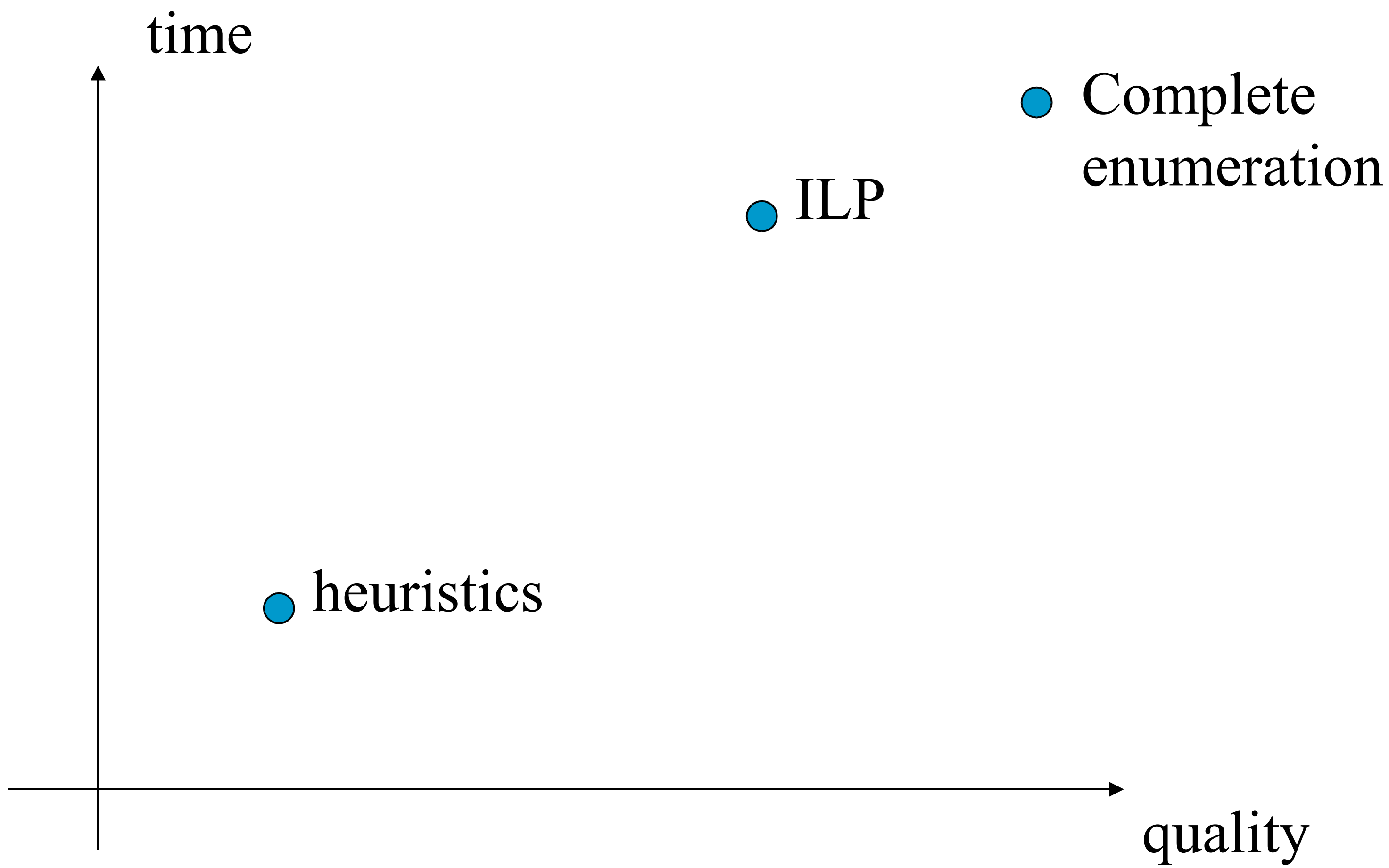

Max. planar subgraph

Min. crossings

# ILP - motivation

- Possible to get the optimum result faster than by complete enumeration
- Known precision of the solution, if a terminated run
- Applications where quality is important (like publishing)
- Generation of comparative results for proving heuristics
- Study of the problem from a different angle

# ILP vs other approaches



# Linear program - Example

A chemical factory has a line with 3 machines:

Unit price


$x_1$ 100 EEK/t

$x_2$ 200 EEK/t

| A | B | C |
|---|---|---|
| $x_1$: 16t/day | $x_1$: 4t/day | $x_1$: 6t/day |
| $x_2$: 4t/day | $x_2$: 6t/day | $x_2$: 6t/day |

Max. daily capacity (exclusively)

Find the daily amounts ($x_1$ and $x_2$) of products so that the total price is maximal.

# Linear program - formulation

$\max(100x_1 + 200x_2)$

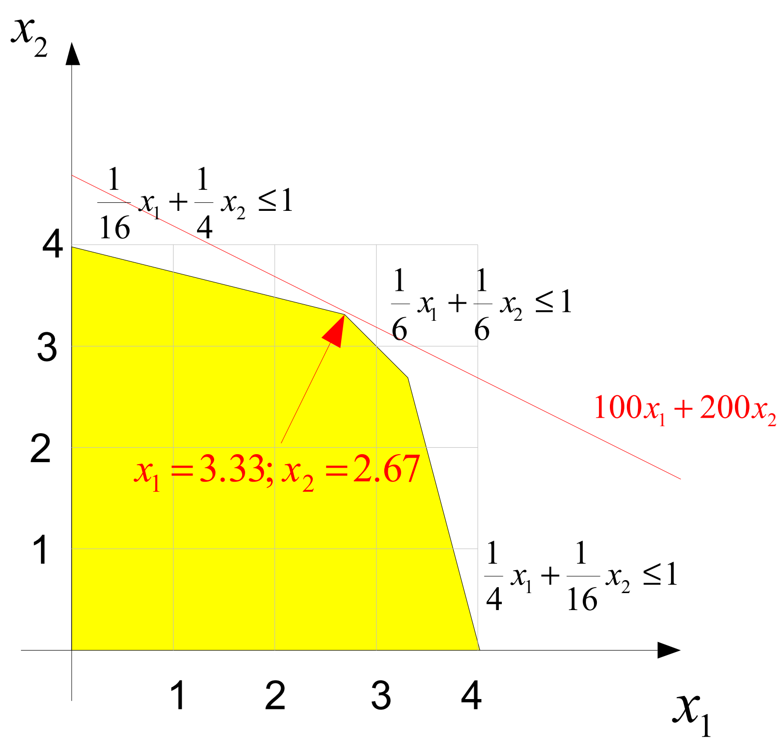
subject to:

(A) $\frac{1}{16}x_1 + \frac{1}{4}x_2 \le 1$

(B) $\frac{1}{4}x_1 + \frac{1}{16}x_2 \le 1$

(C) $\frac{1}{6}x_1 + \frac{1}{6}x_2 \le 1$

$x_1, x_2 \ge 0$



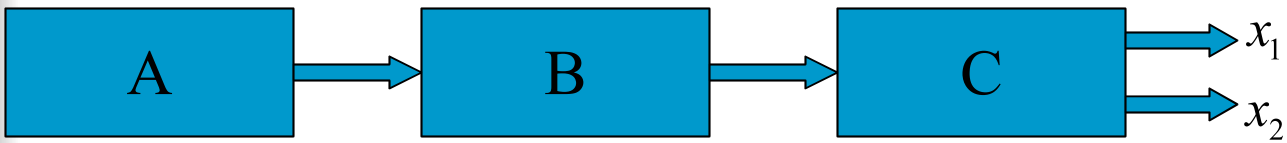
# LP solution methods

- Simplex method (Dantzig 1947)
  - basically a greedy search along the vertices of the polytope determined by constraints
  - polynomially unbounded
  - works well in practice
- Ellipsoid (Khachyan 1979) and interior point (Karmarkar 1984) methods
  - polynomially bounded

# Integer Linear program - Example

A car factory has a line with 3 machines:

Piece price




$x_1$ 100 kEEK

$x_2$ 200 kEEK

| A | B | C |
|---|---|---|
| $x_1$: 16p/day | $x_1$: 4p/day | $x_1$: 6p/day |
| $x_2$: 4p/day | $x_2$: 6p/day | $x_2$: 6p/day |

Max. daily capacity (exclusively)

How many ($x_1$ and $x_2$) of the different car models must be manufactured so that the total price will be maximal.

# Integer linear program - formulation

$$\max(100x_1 + 200x_2)$$

subject to:

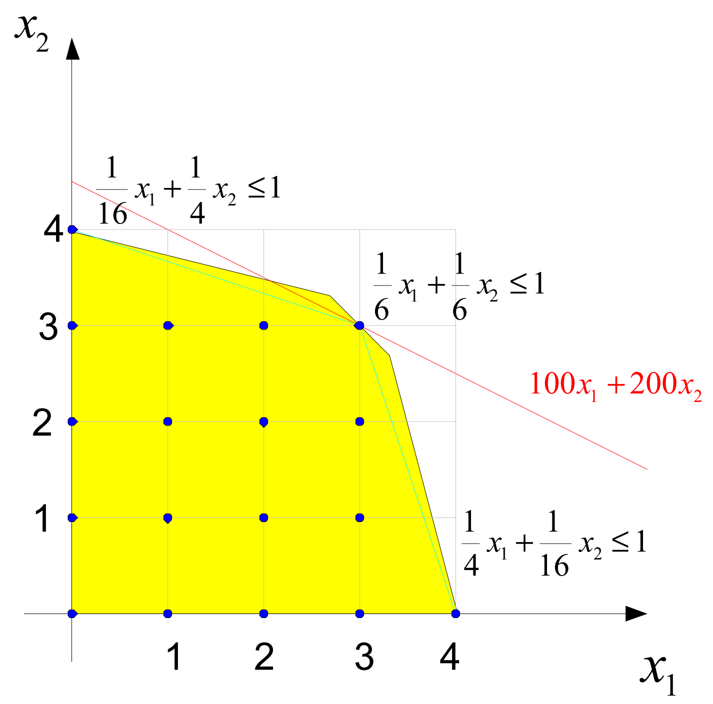
$$\text{(A)} \quad \frac{1}{16}x_1 + \frac{1}{4}x_2 \le 1$$

$$\text{(B)} \quad \frac{1}{4}x_1 + \frac{1}{16}x_2 \le 1$$

$$\text{(C)} \quad \frac{1}{6}x_1 + \frac{1}{6}x_2 \le 1$$

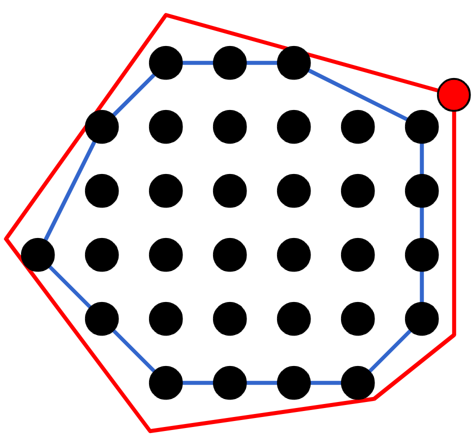
$$x_1, x_2 \ge 0$$

$$x_1, x_2 \text{ integral}$$



# ILP algorithms

- When the solution of a LP-relaxation is
  - <u>Integral</u> – we have found the optimal solution
  - <u>Fractional</u> – need to define a more constrained problem



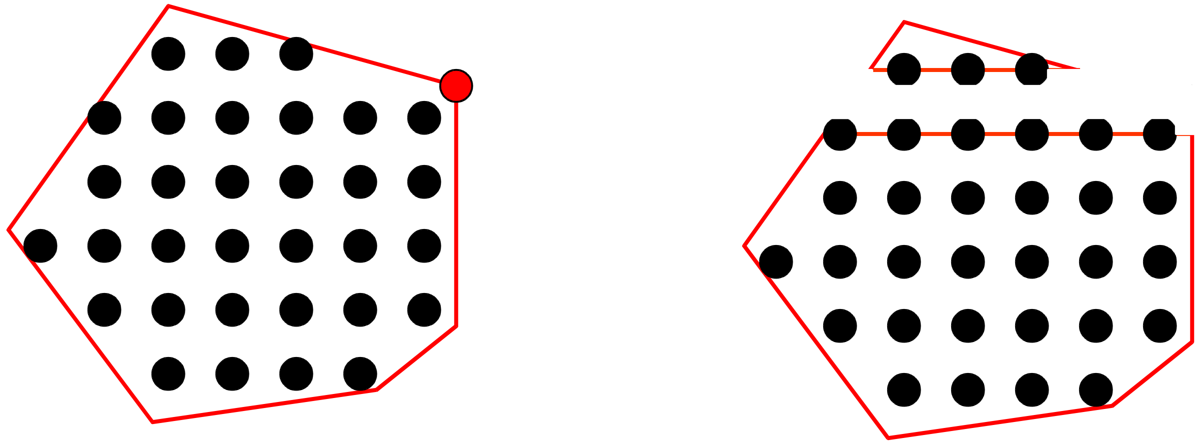
# Branch-and-bound algorithm

- Solve the initial LP, let *LB* = *cx*
- Select a fractional variable $x_i$=$t_i$ and create two subproblems:

$$\min\left\{ cx \mid Ax \le b, x_i \le \lfloor t_i \rfloor \right\} \quad \min\left\{ cx \mid Ax \le b, x_i \ge \lceil t_i \rceil \right\}$$

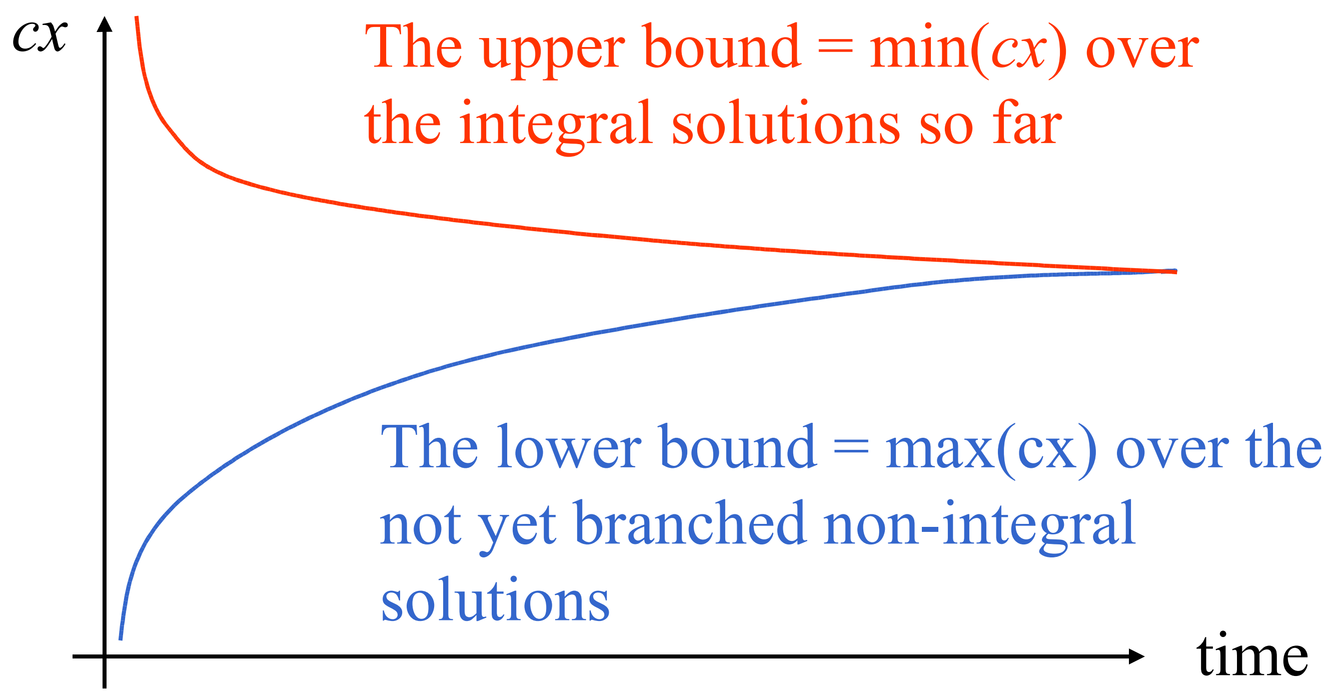
- For each subproblem, SOLVE, if
  - $cx > LB$ ↙ don’t explore this branch
  - $cx < LB$ ↙
    - Non-integral $x$: REPEAT with some $x_j$
    - Integral $x$:
      - $LB = cx$, if no more unexplored subproblems ↙ optimal solution
      - $x$ is infeasible ↙ don’t explore this branch

# Branch-and-bound



# The lower and upper bounds

Consider a minimisation problem:



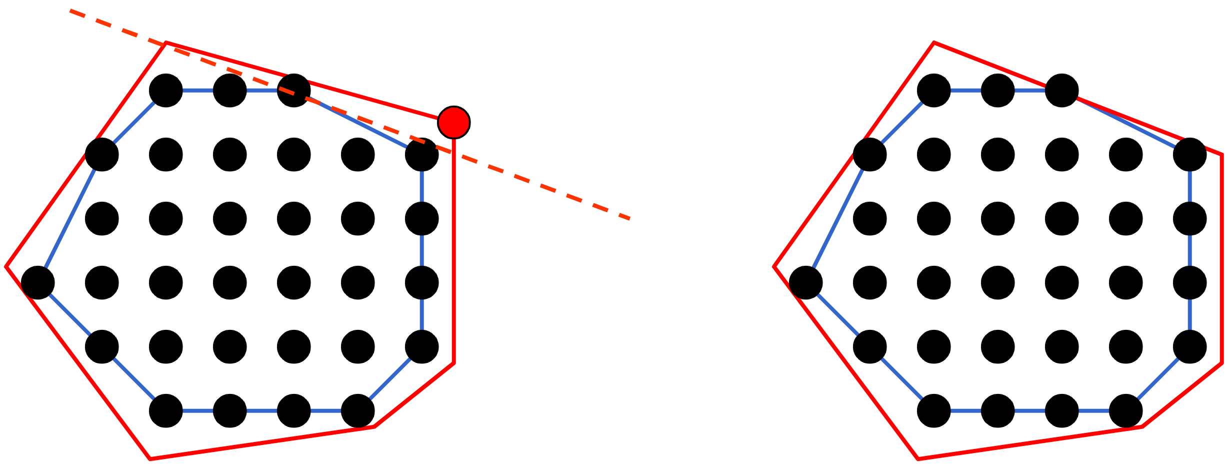
# Cutting plane algorithm

Do

1. Solve the LP-relaxation
2. If $x$ is not integral
    1. Add a constraint to LP-relaxation that separates $x$ from the polytope

While x not integral

# Cutting plane



# Branch and Cut

- Based on branch-and bound algorithm
- Each time a subproblem results in a non-integer solution *x*, try to find a <u>cutting plane</u> <u>separating</u> *x* from the polytope
- Use only <u>binding</u> constraints for a subproblem

# Polyhedral combinatorics

Binary vector: $x$

The set of all **valid** binary vectors: $S$

Weights vector: $c$

General combinatorial optimisation problem:

$$\min\{\, cx \text{ subject to } x \in S \}$$

Replace it with a linear optimisation problem:

$$\min\{\, cx \text{ subject to } Ax \leq b \}$$

$$S = P \cap \mathbf{Z}^n, \quad P = \{\, x \in \mathbf{R}^n : Ax \leq b \}$$

# Polyhedral combinatorics

- How to define $Ax \leq b$? This an art!
- The best inequalities are those that define a **facet** of the polytope.
- A facet-defining inequality holds as an equality for $|x|$ linearly independent solutions.

# How to derive an ILP formulation?

- Derive from the ILP formulation of some similar problem, e.g.:
  - Multi-level crossing minimisation ↗ linear ordering
  - Maximum level planar subgraph ↗ maximum planar subgraph
- Analyse smaller problem instances by software.

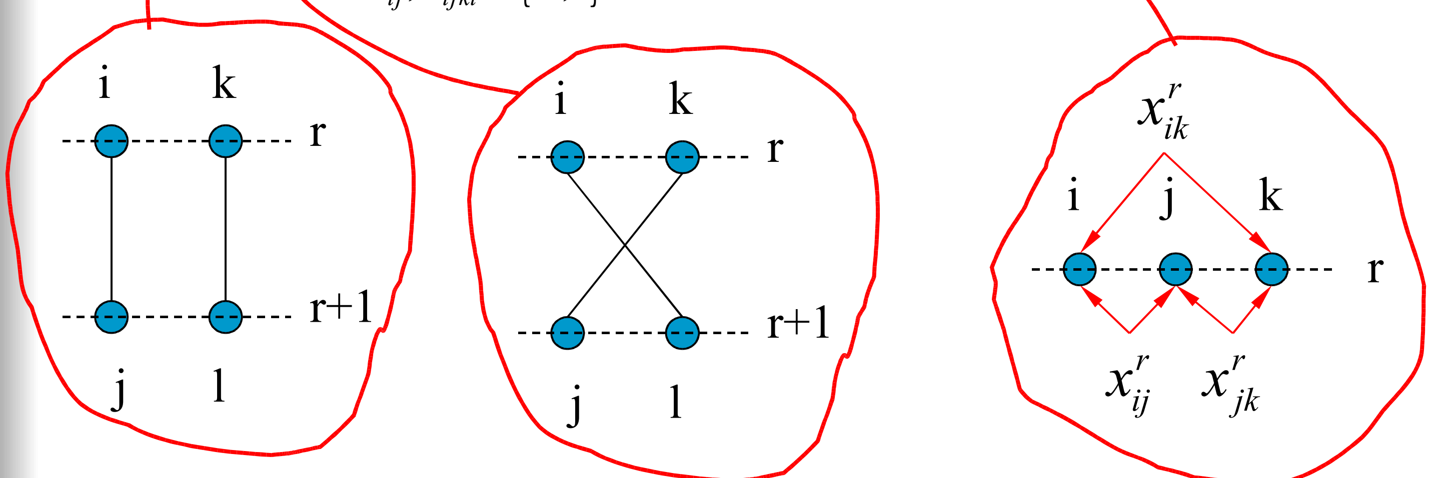
  http://www.informatik.uni-heidelberg.de/groups/comopt/software/PORTA/

  (PORTA derives from all enumerated solutions the facets of the polytope supported by these solutions)

# Multi-Level Crossing Minimisation ILP

Minimise $$\sum_{r=1}^{p-1} \sum_{(i,j)(k,l)\in E_r} c^r_{ijkl}$$

Subject to
$$-c^r_{ijkl} \le x^{r+1}_{jl} - x^r_{ik} \le c^r_{ijkl} \quad (i,j),(k,l)\in E_r,\ j<l$$
$$1-c^r_{ijkl} \le x^{r+1}_{lj} + x^r_{ik} \le 1+c^r_{ijkl} \quad (i,j),(k,l)\in E_r,\ j<l$$
$$0 \le x^r_{ij} + x^r_{jk} - x^r_{ik} \le 1 \quad i<j<k$$
$$x^r_{ij}, c^r_{ijkl} \in \{0,1\}$$



# The vertex-exchange graph

Consider the crossing constraints:

$$-c^r_{ijkl} \le x^{r+1}_{jl} - x^r_{ik} \le c^r_{ijkl}$$

$$1 - c^r_{ijkl} \le x^{r+1}_{lj} + x^r_{ik} \le 1 + c^r_{ijkl}$$

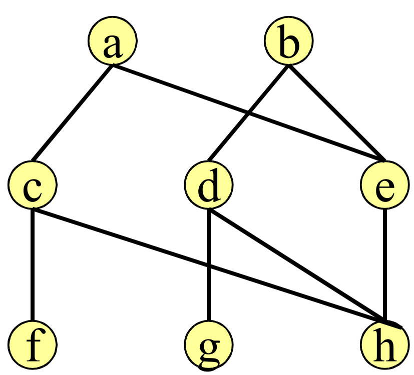
They establish relationships between linear ordering variables.

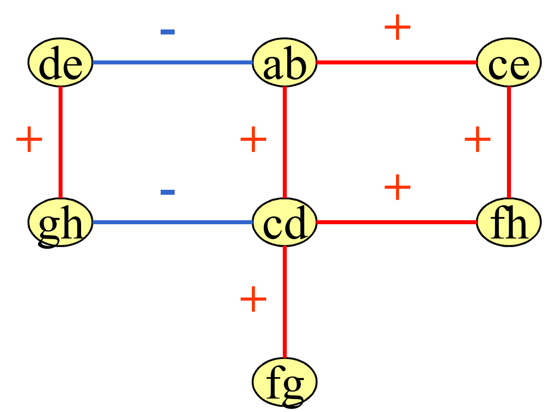
If $c^r_{ijkl} = 0$:

$$x^{r+1}_{jl} = x^r_{ik}$$

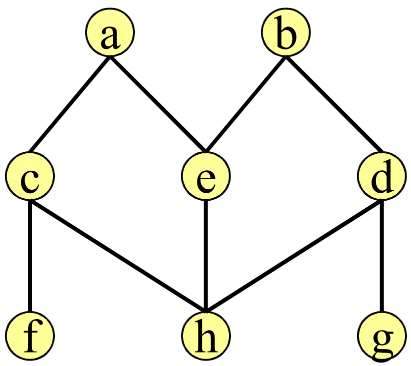
$$x^{r+1}_{lj} = 1 - x^r_{ik}$$

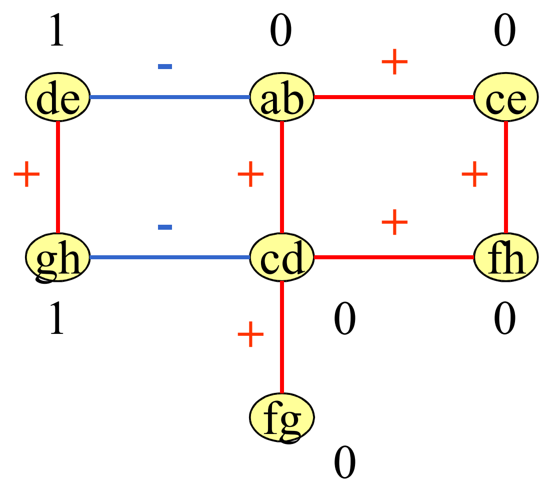
Objects & relationships ↙ Graph

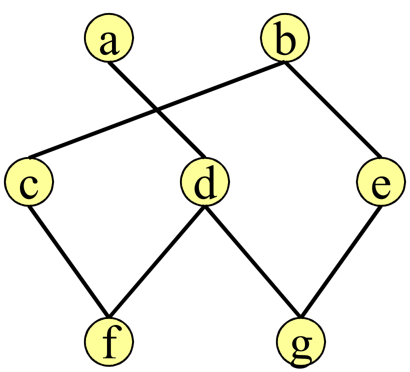


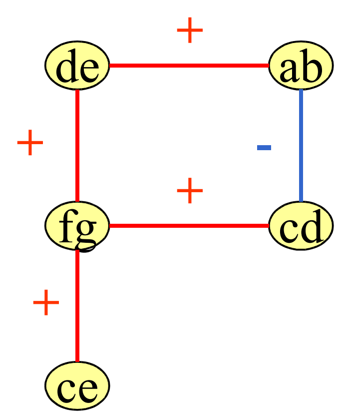



# Level planarity testing



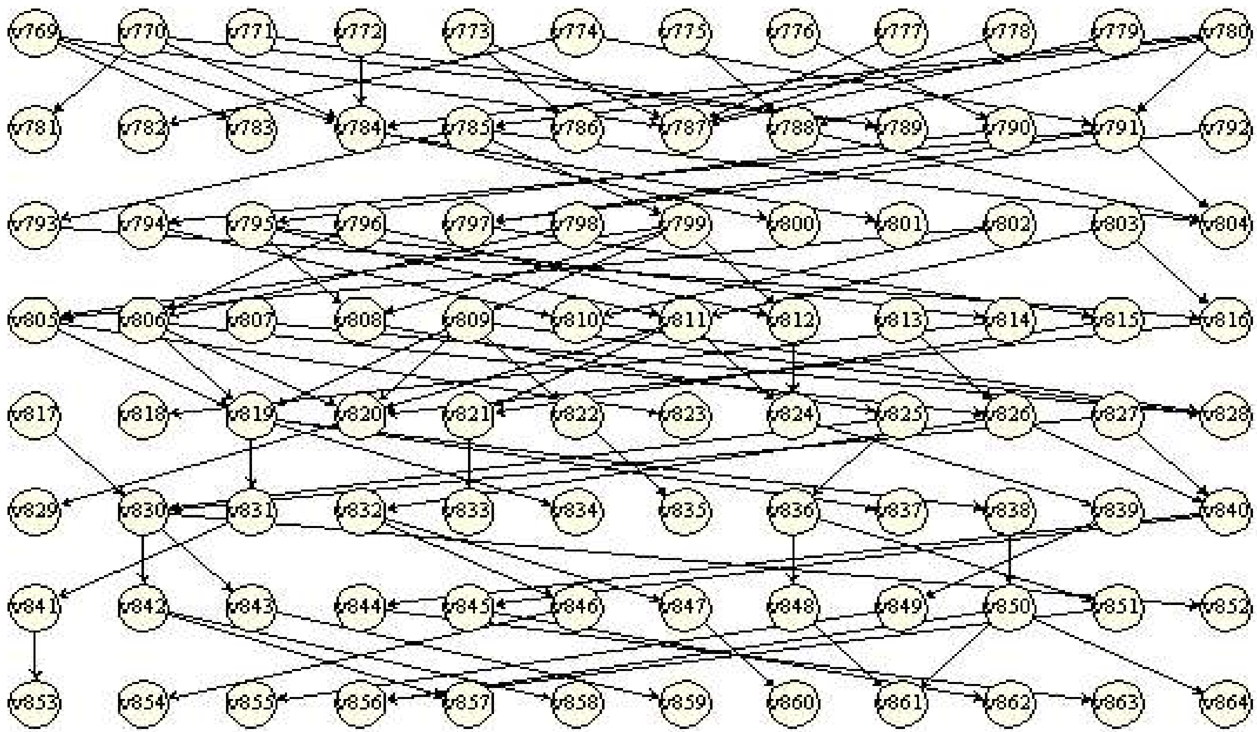



# Benefits of V-E graph

- $O(|V|^2)$ level planarity testing algorithm
  ($|E| < 2|V| - 4$)
- $O(|V|^3)$ layout algorithm for level planar graph
- V-E graph odd/labelled cycle inequalities to crossing minimisation ILP:

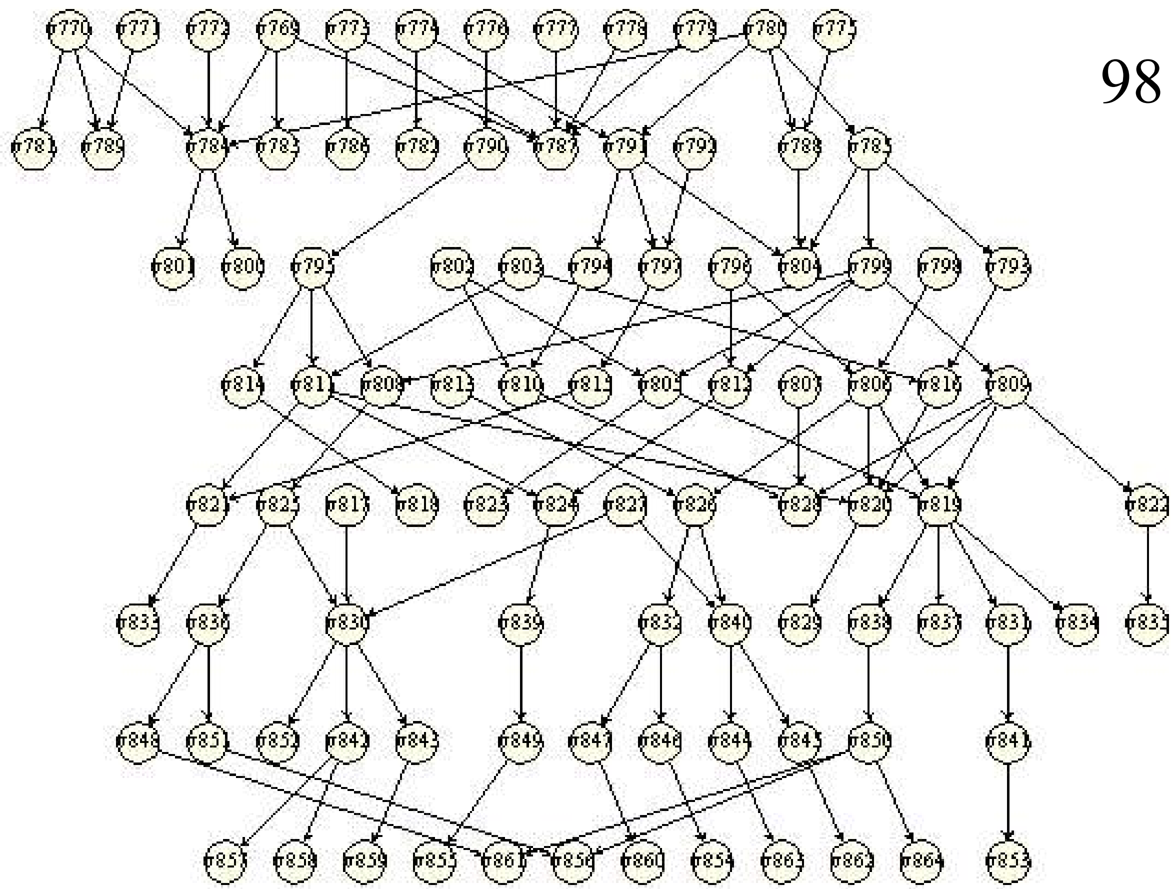
$$\sum_{c^r_{ijkl} \in C} c^r_{ijkl} \geq 1, \quad C \text{ - an odd-labelled fundamental cycle in V-E graph}$$

# Example (random layout)



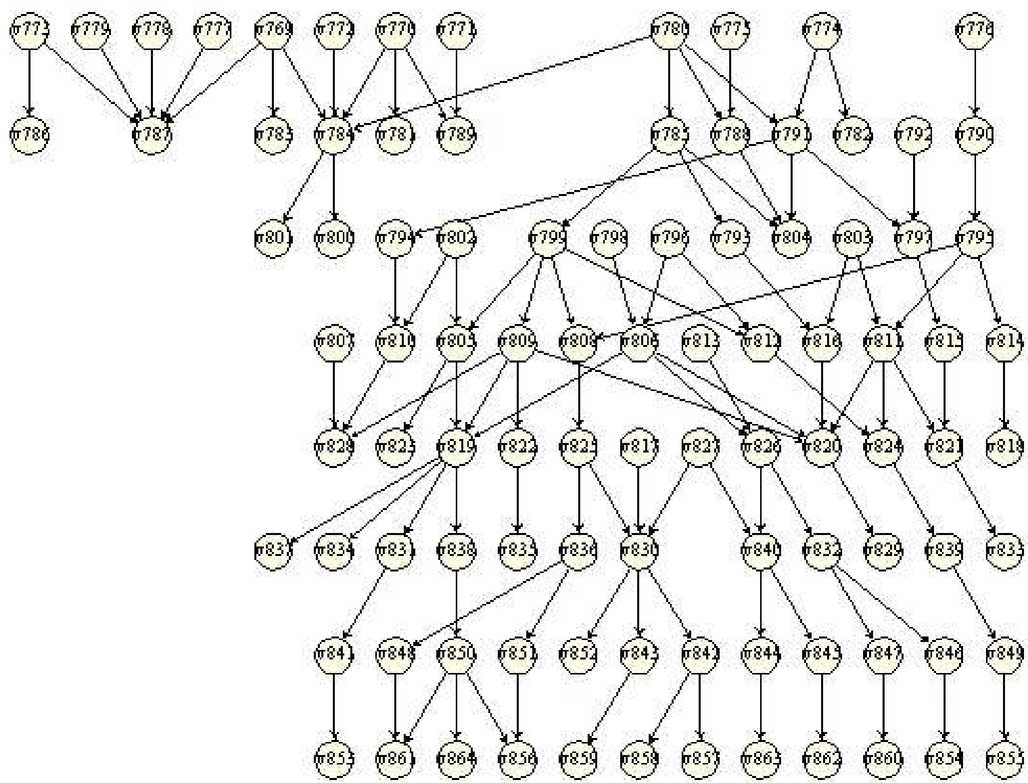

96 vertices, 110 edges

# Example (Sugiyama layout)



98 crossings

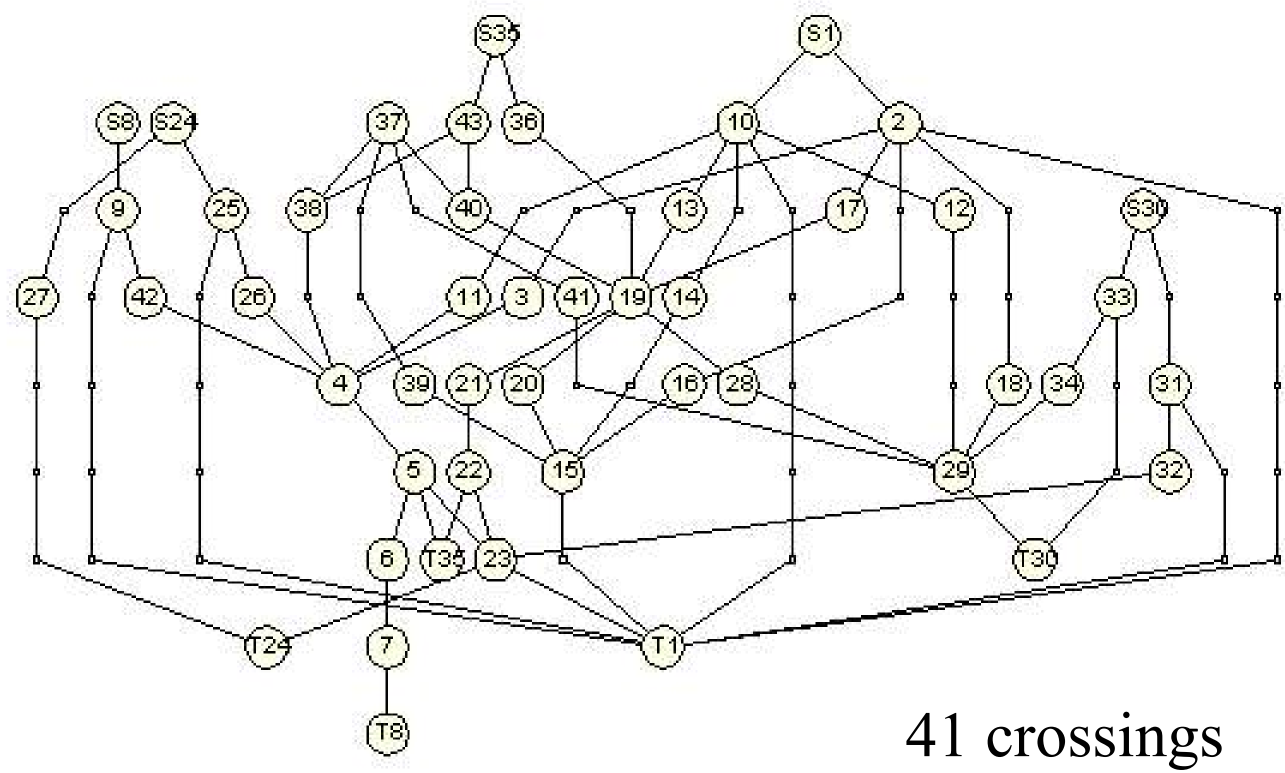
# Example (ILP layout)



31 crossings

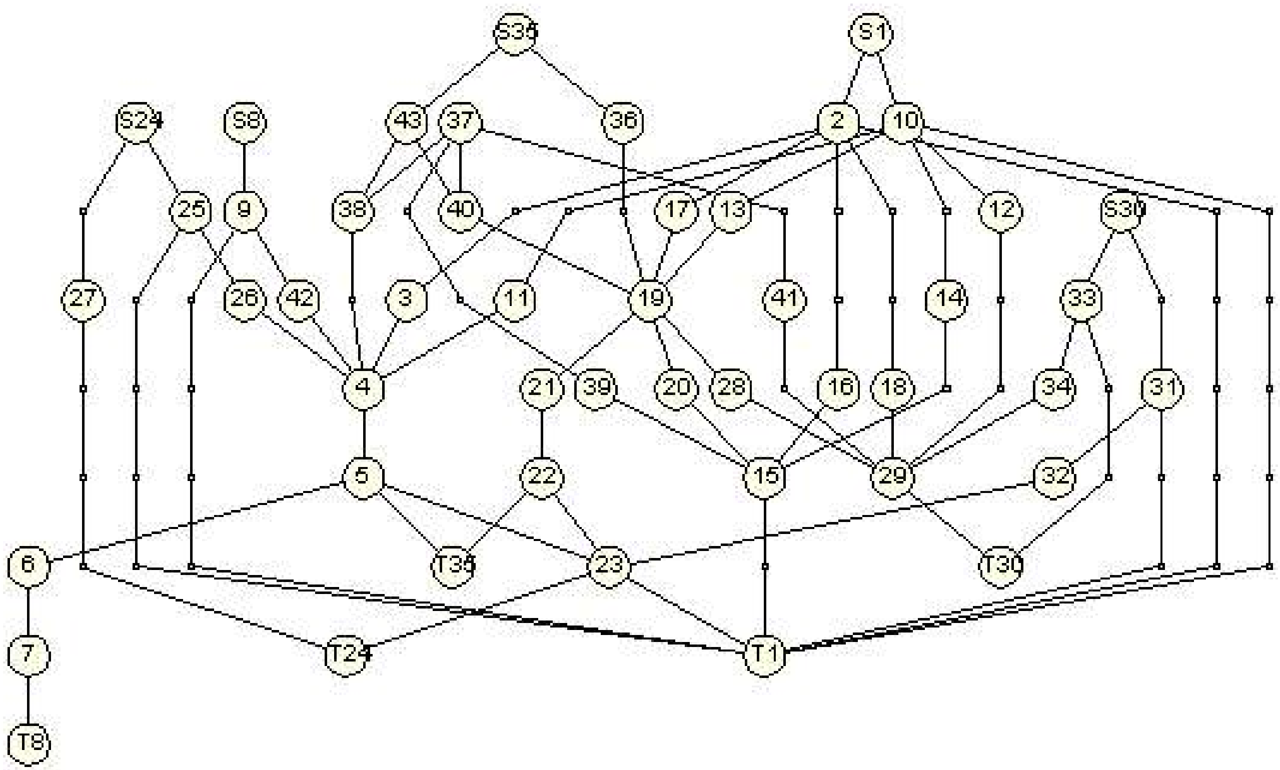
88s on 300MHz DEC AlphaStation

# Example (Gansner *et al.*, ‘93)




41 crossings

# Example (ILP)




38 crossings, 143s

# Independence system

- A set *E*; ***I*** is a set of subsets of *E*,
- $F_1 \subseteq F_2 \subseteq E$ and $F_2 \in$ ***I*** ⇒ $F_1 \in$ ***I***
- ⇒ (*E*,***I***) is an *independence system*
- If $C \subseteq E$, $C \notin$ ***I*** and for every $F \subset$ C, $F \in$ ***I***
  ⇒ *C* is a *circuit*

Determine the maximum weighted member of ***I***:

$$\max c^T x$$

subject to:

$$a_{i1}x_1 + a_{i2}x_2 + \ldots + a_{i|E|}x_{|E|} = |C_i| - 1, \quad i = 1, 2 \ldots, | \quad |$$

$$a_{ij} = \begin{cases} 0, e_j \notin C_i \\ 1, e_j \in C_i \end{cases}$$

# Maximum level planar subgraph ILP

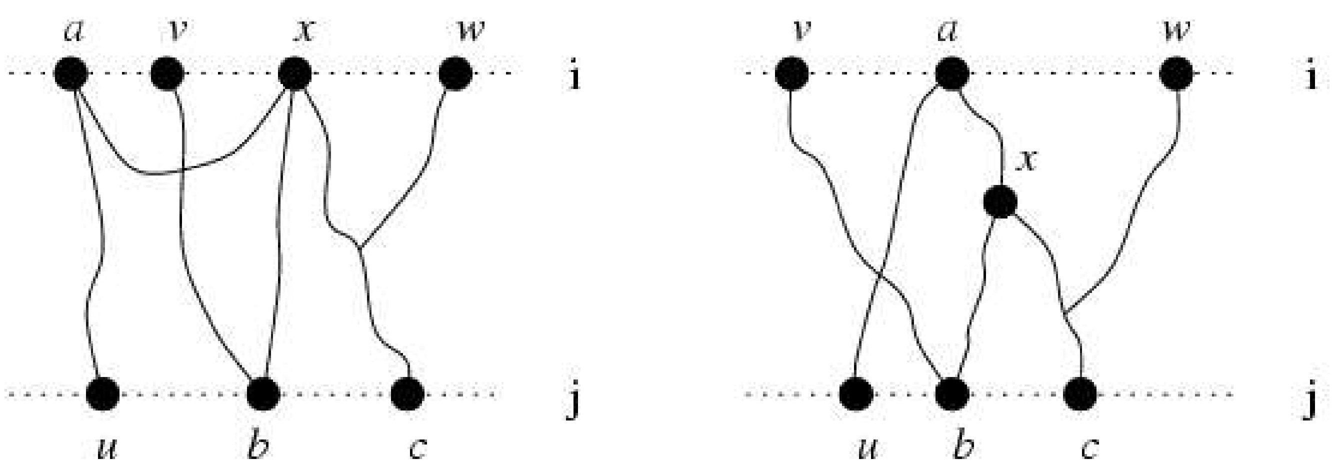
Independent sets – level planar subgraphs
Circuits –minimal level non-planar subgraphs

Given a level graph $G=(V,E)$:

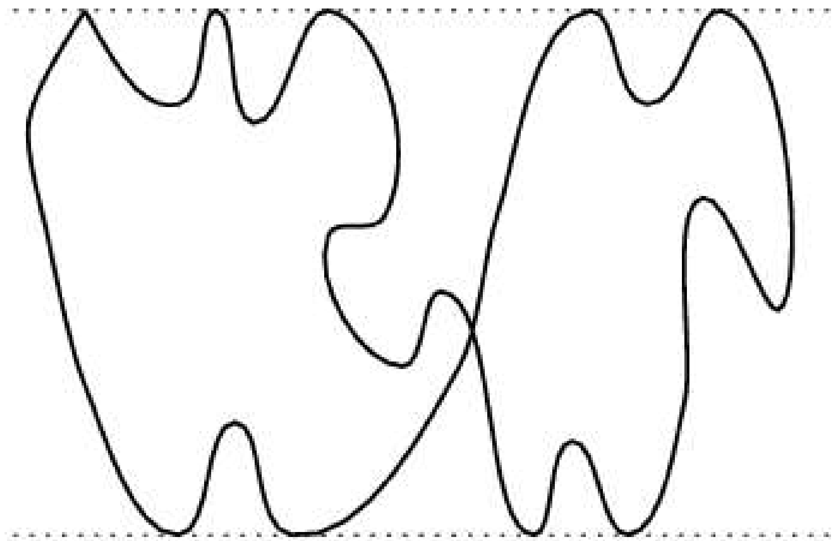
Maximise $\sum x_e$ $\quad E_p \subset E, \quad e \in E, \quad x_e = \begin{cases} 1, & e \in E_p \\ 0, & e \notin E_p \end{cases}$

Subject to: $\sum_{e \in S} x_e \leq |S| - 1$ $\quad S$ is a MLNP subgraph

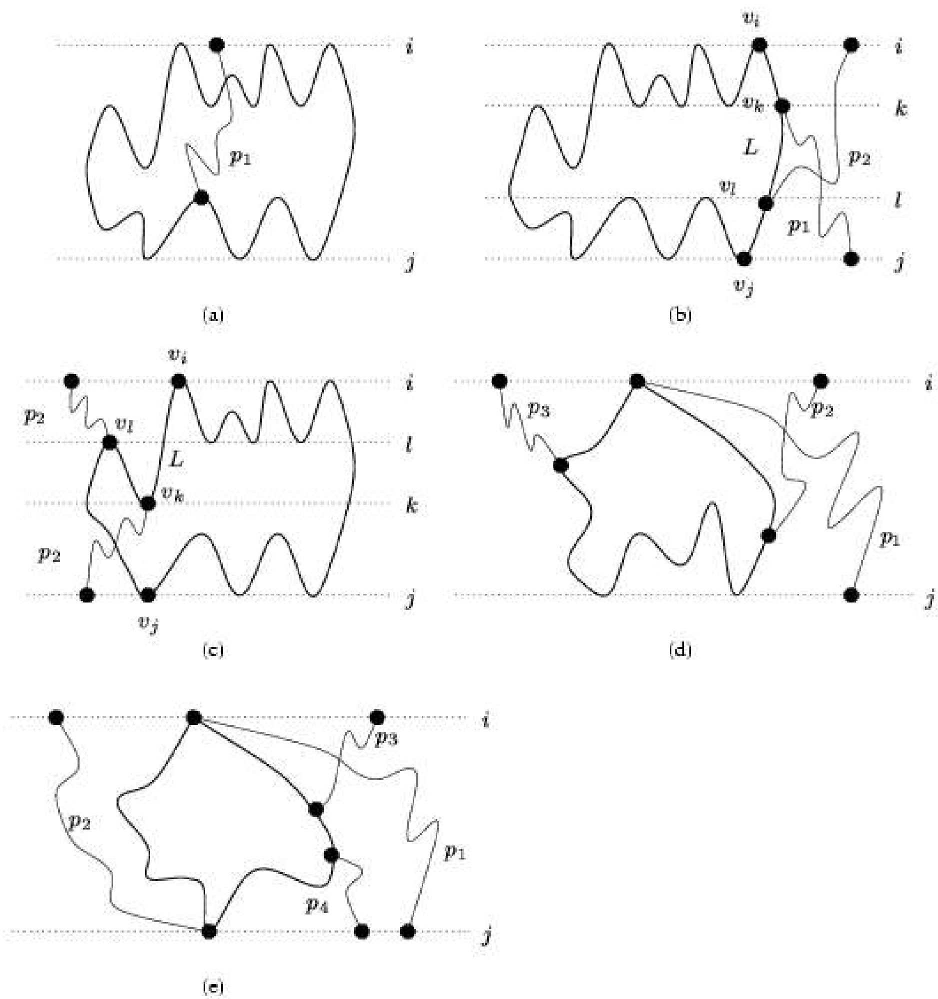
# MLNP subgraphs - trees



# MLNP subgraphs – LNP cycles



# MLNP subgraphs – LP cycles + paths



# Separation algorithm

**Algorithm 6** $NaiveSeparation(V, E)$

$E' \leftarrow E$
**for all** $e \in E'$ **do**
    $E' \leftarrow E' \setminus \{e\}$
    **if** $LevelPlanar(V, E')$ **then**
        $E' \leftarrow E' \cup \{e\}$
    **end if**
**end for**
return $E'$

# Using odd-labelled cycles of V-E graph

**Algorithm 7** $InformedSeparation(V, E, \mathcal{V}, \mathcal{E})$

find an odd-labelled cycle $\mathcal{C} \subseteq \mathcal{E}$

$C \leftarrow \{e \mid e, f \in E,\ \langle e, f \rangle \in \mathcal{C}\}$

return $NaiveSeparation(V, C)$

Use the edges of the original graph
that induced an odd-labelled cycles
in the vertex-exchange graph

# Primal heuristic

**Algorithm 9** *ImprovedPrimal*$(V, E)$

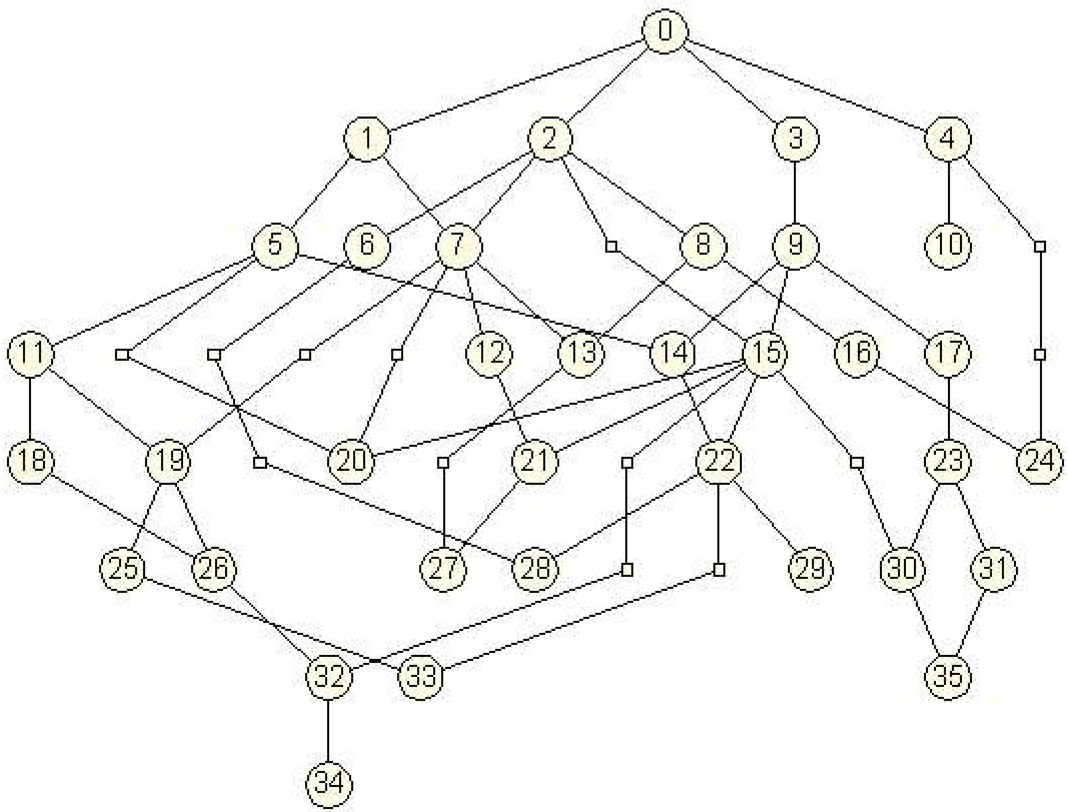
sort $E$ by increasing $x_e$<br>
$E_p \leftarrow E$<br>
$E_{np} \leftarrow \emptyset$<br>
**for all** $e \in E_p$ **do**<br>
    $E_p \leftarrow E_p \setminus \{e\}$<br>
    $E_{np} \leftarrow E_{np} \cup \{e\}$<br>
    **if** *LevelPlanar*$(V, E_p)$ **then**<br>
        break<br>
    **end if**<br>
**end for**<br>
**for all** $e \in E_{np}$ **do**<br>
    $E_p \leftarrow E_p \cup \{e\}$<br>
    **if** not *LevelPlanar*$(V, E_p)$ **then**<br>
        $E_p \leftarrow E_p \setminus \{e\}$<br>
    **end if**<br>
**end for**<br>
return $E_p$

- is executed each time a subproblem is solved.
- The result vector *x* is employed as weight function for a greedy heuristic

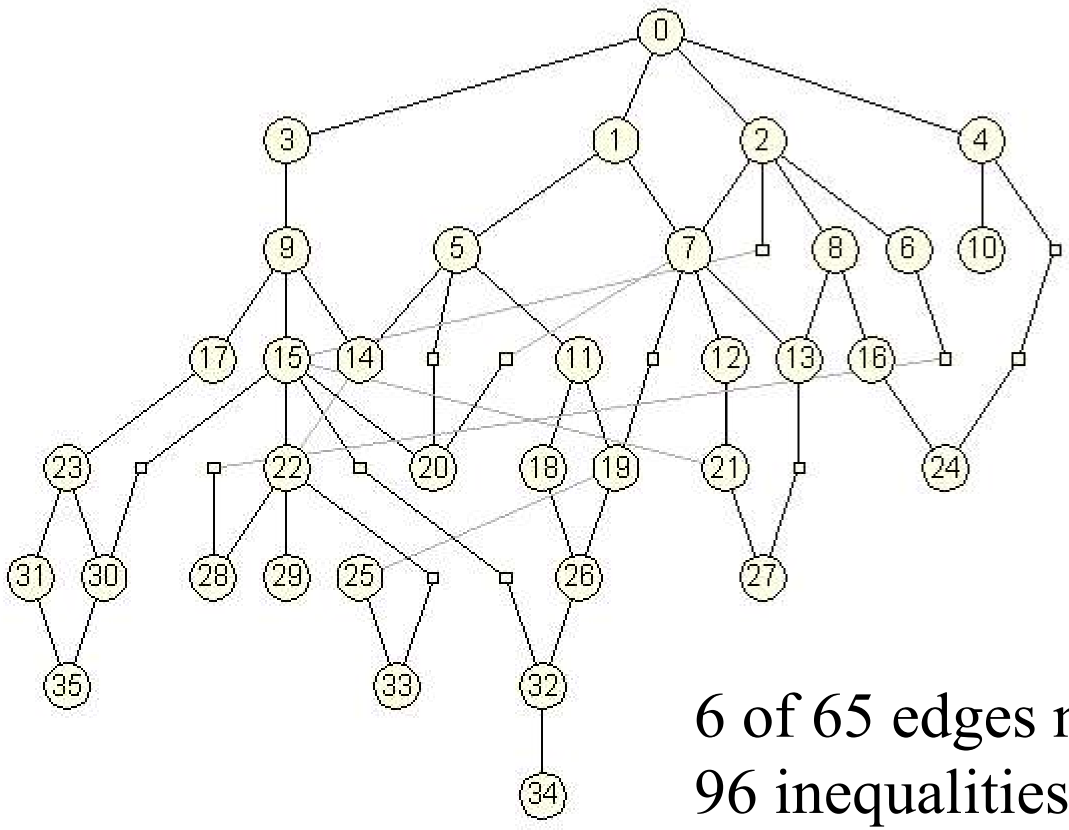
<u>2 stages:</u>

4. Removal
5. Adding attempts

# MLP subgraph example - original



# MLP subgraph example – B&C




6 of 65 edges non-planar
96 inequalities
24s

# Conclusions

<u>Results</u>

- MLNP subgraphs
- Vertex exchange graph
- Improved crossings minimisation ILP
- Max. planar subgraph ILP

<u>Open problems</u>

- Specific algorithms for detecting MLNP subgraphs?
- Can we estimate the crossing number by V-E graph?
- Efficiency of ILP-s:
  - employ O(n) level planarity testing
  - b&c crossing min ILP

Thank you for attention!