Exact Computation of Crossing Numbers

Markus Chimani
Friedrich-Schiller-University Jena, Germany

NP-hard Problems…. so what?

Unless P=NP:

Solving NP-hard problems requires exponential time **in general**

Traditional algorithmics

What can we achieve in polynomial time?

→ Heuristics, Approximations, Fixed-parameter-tractability (FPT)

Alternative Approach

Ist the worst-case exponential time **really that bad**?

→ Consider algorithms that give **exact solutions** that are **usually** sufficiently **fast**.

But how?

Often successful: **Mathematical Programming** techniques.

Some Success Stories of Math.Prog.

Travelling Salesman Problem

Given *N* cities and distances in between them.
Find the shortest round trip through all of them.

Applications: routing, soldering, robotics…

1954: Dantzig, Fulkerson, Johnson
  49 cities (US state capitals) ✓
  manually!

Pioneering Math.Prog.: Cuts, Branch-and-Cut,…

Now: Exact algorithms work even for
  large scale instances

Sweden	24.978	✓
VLSI	85.900	✓
World TSP	1.904.711	<0.05%

George B. Dantzig
TSP 25.000 cities
(by Robert Bosch)

Some Success Stories of Math.Prog.

Various success stories in many different fields of optimizations

Large Instances

e.g. **TSP…**

Fast

e.g. ***k*-Cardinality Tree**

- Given a weighted graph. Find the cheapest subtree with *k* edges.
- *Applications:* network design, oil-field leasing,…
- Exact algorithms solve all established benchmark sets;
  for small&medium graphs even **faster than the best inexact approaches**

Influence on other CS fields

e.g. **Primal-Dual Approximation Algorithms**

- based on math.prog. formulations and polyhedral studies

Linear Programming and Beyond

Linear Program (LP)

- Set of variables
- Linear objective function
- Set of linear constraints

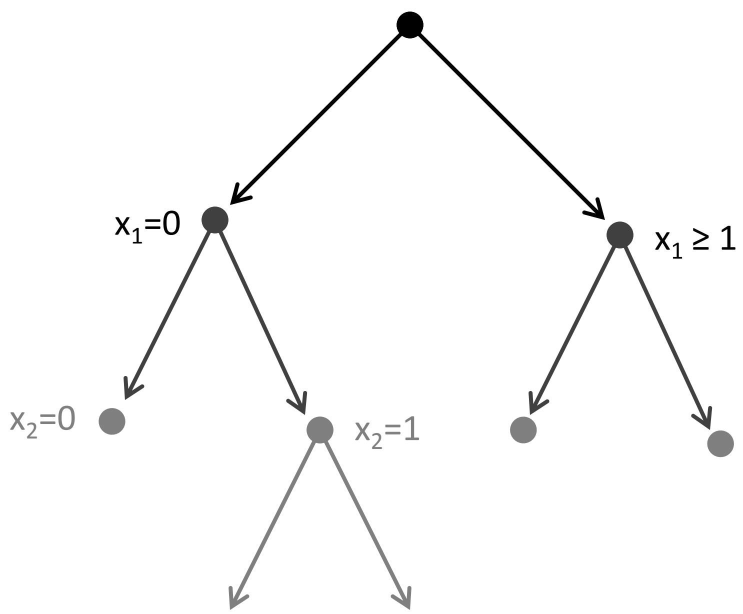
Example:

$$\begin{aligned} \min \quad & x_1 + 2\cdot x_2 - 5\cdot x_3 \\ s.t. \quad & x_1 + 4\cdot x_2 \geq 7 \\ & x_3 - x_1 \leq 5 \\ & x_1, x_2, x_3 \geq 0 \end{aligned}$$

LPs can be solved in polynomial time!

Integer Linear Program (ILP)

- Linear program
- require integrality for (some) variables
- NP-hard: Branch-and-Bound



Crossing Number as an ILP

Given: Graph $G=(V,E)$

Variables: For each pair of (non-adjacent) edges $e,f \in E$:

$$x_{\{e,f\}} = \begin{cases} 1 & \text{if } e \text{ is crossed by } f, \\ 0 & \text{else} \end{cases}$$

Objective function: $\min \sum_{e,f} x_{\{e,f\}}$

Current optimum solution: **all zero**
→ Ensure that crossings occur when necessary
→ Enforce that the solution gives a feasible solution → **planarization**

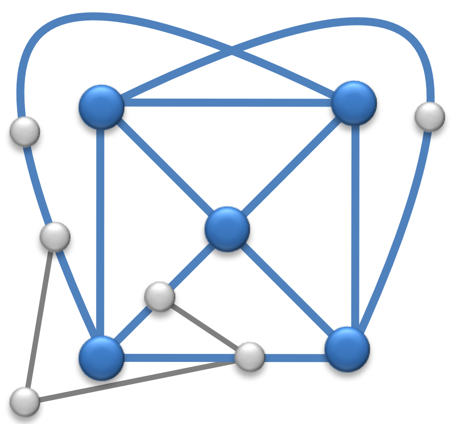
Kuratowski-Constraints

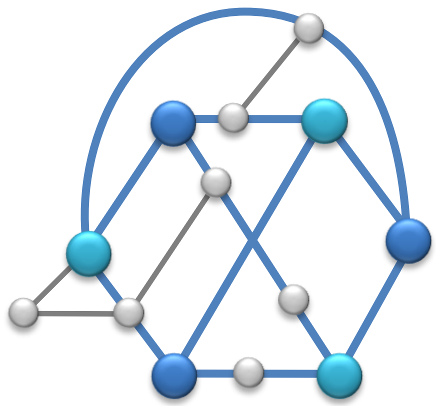
Kuratowski's Theorem (1930):

G is **planar** (= ***G*** can be drawn in the plane without crossings)

$\Leftrightarrow$ ***G*** contains no **Kuratowski subdivisions**

Kuratowski subdivision $\Leftrightarrow$ Subdivision of a K_5 or $K_{3,3}$





Planarity testing: linear-time algorithms [Hopcroft, Tarjan 74]

Crossing Number as an ILP

Given: Graph $G=(V,E)$

Variables: For each pair of (non-adjacent) edges $e,f \in E$:

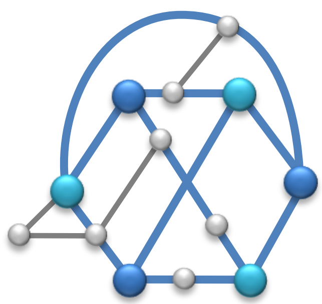
$$x_{\{e,f\}} = \begin{cases} 1 & \text{if } e \text{ is crossed by } f, \\ 0 & \text{else} \end{cases}$$

Objective function: $\min \sum_{e,f} x_{\{e,f\}}$

Kuratowski constraints: For each Kuratowski subdivision K in G:

$$\sum_{\{e,f\} \in C(K)} x_{\{e,f\}} \geq 1$$

where $C(K)$ are the edge pairs belonging to non-adjacent Kuratowski paths of K

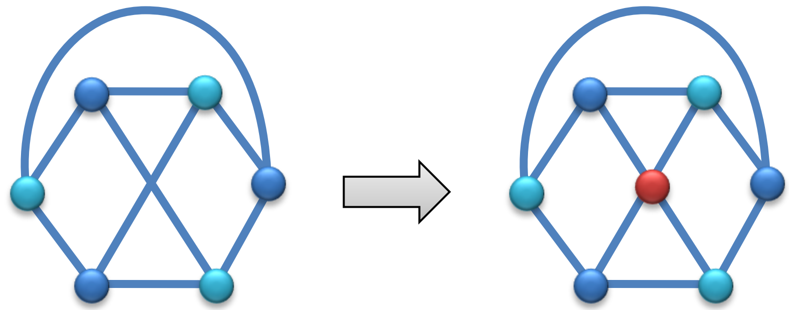


Now only feasible solutions??

Problems

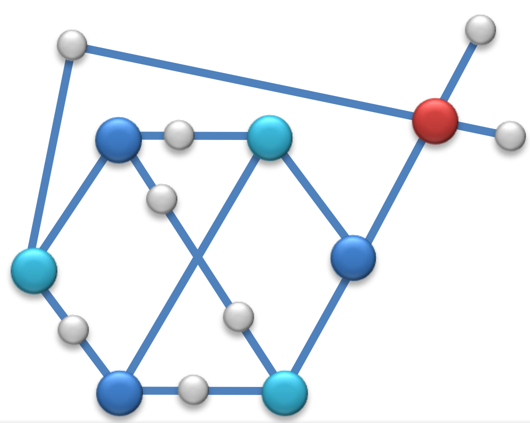
If the solution is feasible, it should induce a **planarization**:

Substituting crossings with dummy nodes (degree 4) should yield a planar graph.



Problem 1

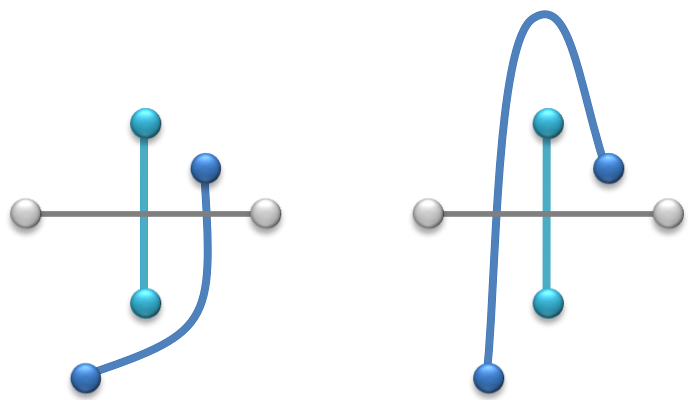
The chosen crossings may not lead to a feasible solution, i.e., further crossings may be necessary, arising from “hidden“ Kuratowskis.



Problems

Problem 2

How to order the crossings/dummy-nodes if multiple crossings per edge?



Realizability problem:

Given edge pairs that cross (our x-variables).
Do they describe a feasible solution?

$\rightarrow$ **NP-complete!** [Kratochvíl 91]

$\rightarrow$ The ILP has to encode the order of the crossings…

Subdivision-based Exact Cr.Min. (SECM)

Observation

Realizability would be trivial if at most one crossing per edge:

Replace crossings by dummy nodes (no problem with order), test planarity

Such a restriction would give "wrong" crossing number on original graph

$\rightarrow$ Subdivide each edge into ℓ edge segments

$\rightarrow \ell$ = upper bound of the crossing number (primal heuristic)

Drawback

- Before: $O(|E|^2)$ variables
- Now: $O(|E|^4)$ variables, since
  $\exists G$: with an edge requiring $\Omega(|E|)$ crossings

Crossing Number as an ILP

Variables: For each pair of (non-adjacent) edges $e,f \in E$:

$$x_{\{e,f\}} = \begin{cases} 1 & \text{if } e \text{ is crossed by } f, \\ 0 & \text{else} \end{cases}$$

Objective function: $\min \sum_{e,f} x_{\{e,f\}}$

Observation: each edge pair crosses at most once

Crossing Number as an ILP

Variables:

For each pair of (non-adjacent) edges $e,f \in E$:

$$x_{\{e,f\}} = \begin{cases} 1 & \text{if } e \text{ is crossed by } f, \\ 0 & \text{else} \end{cases}$$

Objective function:

$$\min \sum_{e,f} x_{\{e,f\}}$$

Consider any orientation of G:

each edge has a direction

Crossing Number as an ILP

Variables:

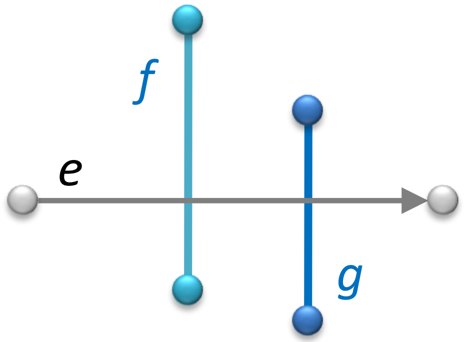
For each pair of (non-adjacent) edges $e,f \in E$:

$$x_{\{e,f\}} = \begin{cases} 1 & \text{if } e \text{ is crossed by } f, \\ 0 & \text{else} \end{cases}$$

Objective function:

$$\min \sum_{e,f} x_{\{e,f\}}$$

Consider any orientation of G:
each edge has a direction



Further variables:

For each ordered triple of edges $e,f,g \in E$:

$$y_{e,f,g} = \begin{cases} 1 & \text{if } e \text{ is crossed by } f \text{ before } g \\ 0 & \text{else} \end{cases}$$

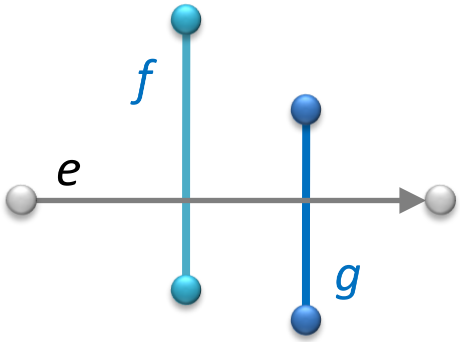
Ordering-based Exact Cr.Min. (OECM)

For each pair of edges $e,f \in E$: $x_{\{e,f\}} = \begin{cases} 1 & \text{if } e \text{ is crossed by } f, \\ 0 & \text{else} \end{cases}$

For each ordered triple of edges $e,f,g \in E$: $y_{e,f,g} = \begin{cases} 1 & \text{if } e \text{ is crossed by } f \text{ before } g, \\ 0 & \text{else} \end{cases}$

Linear order (LO) constraints:

- $x_{\{e,f\}} \geq y_{e,f,g}$, $x_{\{e,g\}} \geq y_{e,f,g}$
- $x_{\{e,f\}} + x_{\{e,g\}} \leq 1 + y_{e,f,g} + y_{e,g,f}$
- $y_{e,f,g} + y_{e,g,f} \leq 1$
- $y_{e,f,g} + y_{e,g,h} + y_{e,h,f} \leq 2$ (cyclic-order)



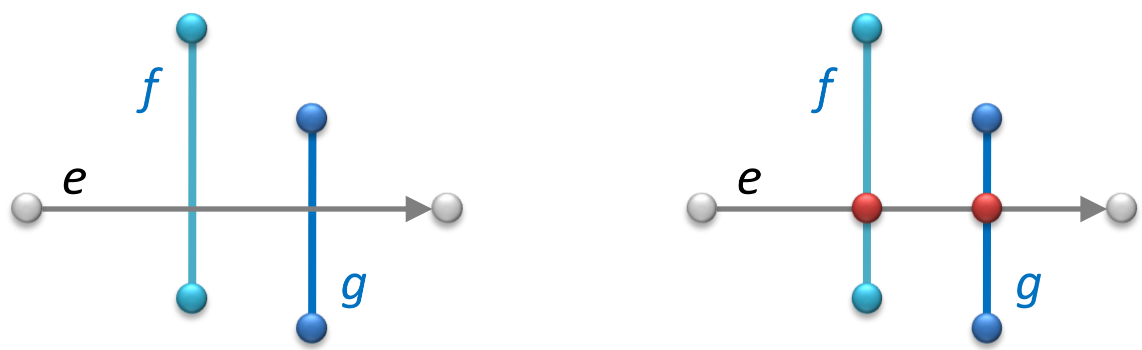
solution LO-feasible
= it satisfies LO-constraints

Ordering-based Exact Cr.Min. (OECM)

[+] any optimal solution can be uniquely described by the variables

[–] variables may describe infeasible solutions

[+] any integer LO-feasible solution (x',y') allows a unique
partial planarization $G[x',y']$:
G, plus dummy nodes for the crossings decribed by (x',y')



An integer LO-feasible solution is **feasible** $\Leftrightarrow$ $G[x',y']$ is **planar**

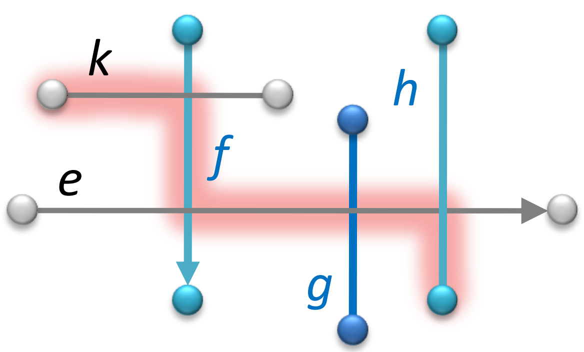
Ordering-based Exact Cr.Min. (OECM)

An integer LO-feasible solution is **feasible** $\Leftrightarrow$ $G[x',y']$ is **planar**

$G[x',y']$ is non-planar $\Leftrightarrow$ $\exists$ Kuratowski subdivision K

Crossing Shadow ($\mathcal{X}_K[x',y']$, $\mathcal{Y}_K[x',y']$):

- minimal description of crossing configuration allowing K
- $\mathcal{X}_K[x',y']$… set of edge pairs $\{e,f\}$:
  e is crossed by f; e and f are involved in a single crossing
- $\mathcal{Y}_K[x',y']$… set of ordered edge triples (e,f,g):
  e is crossed by f directly before g

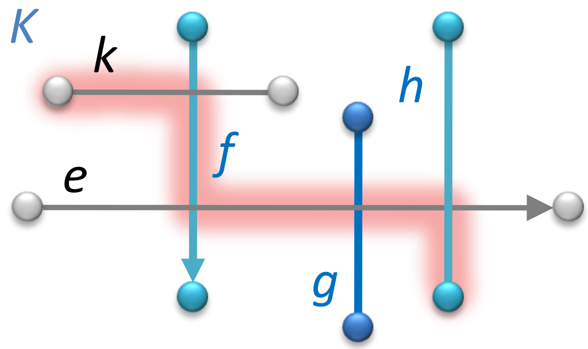


(f,k,e) (e,f,h)

Ordering-based Exact Cr.Min. (OECM)

An integer LO-feasible solution is **feasible** $\Leftrightarrow$ ***G*[*x'*,*y'*]** is **planar**

G[*x'*,*y'*] is non-planar $\Leftrightarrow$ $\exists$ Kuratowski subdivision *K*



Crossing Shadow $(\mathcal{X}_K[x',y'], \mathcal{Y}_K[x',y'])$

Kuratowski constraints:

$\forall$ integer LO-feasible solutions (x',y'), $\forall$ Kuratowski subdivisions K in $G[x',y']$:

$$\sum_{\{e,f\} \in C(K)} x_{\{e,f\}} \geq 1 - \sum_{\{e,f\} \in \mathcal{X}_K[x',y']} (1 - x_{\{e,f\}}) - \sum_{(e,f,g) \in \mathcal{Y}_K[x',y']} (1 - y_{e,f,g})$$

$C(K)$… edge pairs belonging to non-adjacent Kuratowski paths

Integer LO-feasible solution: satisfies all Kuratowski constraints $\Leftrightarrow$ **feasible**

Crossing Number as an ILP

$$\min \sum_{e,f} x_{\{e,f\}}$$

$$\left.\begin{array}{l} \left.\begin{array}{l} x_{\{e,f\}} \geq y_{e,f,g} \\ x_{\{e,g\}} \geq y_{e,f,g} \\ x_{\{e,f\}} + x_{\{e,g\}} \leq 1 + y_{e,f,g} + y_{e,g,f} \end{array}\right\} \text{Bind } x \text{ and } y \\ \left.\begin{array}{l} y_{e,f,g} + y_{e,g,f} \leq 1 \\ y_{e,f,g} + y_{e,g,h} + y_{e,h,f} \leq 2 \end{array}\right\} \text{Order } y \text{ if set} \end{array}\right\} \text{Linear order (LO) constraints}$$

exponentially many!

$$\sum_{\{e,f\} \in C(K)} x_{\{e,f\}} \geq 1 - \sum_{\{e,f\} \in \mathcal{X}_K[x',y']} (1 - x_{\{e,f\}}) - \sum_{(e,f,g) \in \mathcal{Y}_K[x',y']} (1 - y_{e,f,g})$$

∀ integer LO-feasible solutions (x',y'),
∀ Kuratowski subdivisions K in $G[x',y']$

$x_{\{e,f\}} \in \{0,1\}$ ∀ pairs of (non-adjacent) edges $e,f \in E$

$y_{e,f,g} \in \{0,1\}$ ∀ ordered triples of edges $e,f,g \in E$

How to solve such a formulation?

Necessary

- Integer solution required $\Rightarrow$ Branch-and-Bound
- Many constraints (i.p. exponentially many Kuratowski constraints)
  $\Rightarrow$ "cutting" = generate constraints on the fly as necessary

$\Rightarrow$ Branch-and-Cut algorithm

To make it practical

- Many variables $O(|E|^3)$ $\Rightarrow$ column generation (Branch-and-Cut-and-Price)
- Preprocessing (shrink input graph)
  $\Rightarrow$ non-planar-core reduction [Ch., Gutwenger 05]
- Primal heuristic (upper bounds)
  $\Rightarrow$ planarization heuristic [Gutwenger, Mutzel 03], [Ch., Gutwenger 11]
- Efficient extraction of multiple Kuratowski-subdivisions
  in a non-planar graph [Ch., Mutzel, Schmidt 07]

Branch-and-Cut (no column generation)

- initialize **current model**:
  - all LO-constraints except for cyclic-order ($y_{e,f,g} + y_{e,g,h} + y_{e,h,f} \leq 2$)
  - **no** Kuratowski constraints

1) Solve **LP relaxation** (i.e., ignore integrality req.) of current model $\rightarrow (x',y')$
2) Separation **A**: identify violated cyclic-order constraints; add and goto (1)
3) Integer interpretation (x'',y'') of (x',y')
   a) Rounding: $x''_{\{e,f\}}=1$ if $x'_{\{e,f\}}>\tau$; F_e=edges f with $x''_{\{e,f\}}=1$
   b) $\forall e$: complete graph G_e on nodes F_e: edge $\{f,g\}$ has weight $y'_{e,f,g}$
   c) $\forall e$: (heuristically) solve linear order problem on $G_e \rightarrow$ gives y''
4) Separation **B** (heuristic): Kuratowski constraints
   a) search for Kuratowski subdivisions in $G[x'',y'']$
   b) add corresponding constraint if violated, goto (1)
5) Branch…

Branch-and-Cut (no column generation)

- initialize **current model**:
  - only x variables

1) Solve **LP relaxation** (i.e., ignore integrality req.) of current model $\rightarrow$ (x',y')
2) Separation **A**: identify violated cyclic-order constraints; add and goto (1)
3) Integer interpretation (x'',y'') of (x',y')
   a) Rounding: $x''_{\{e,f\}}=1$ if $x'_{\{e,f\}}>\tau$; F_e=edges f with $x''_{\{e,f\}}=1$
   +) if $y_{e,f,g}$ not in curent model for some $f,g \in F_e$:
   add $y_{e,f,g}$ + all necessary LO-constraints (except cyclic-order), goto (1)
   b) $\forall$e: complete graph G_e on nodes F_e: edge $\{f,g\}$ has weight $y'_{e,f,g}$
   c) $\forall$e: (heuristically) solve linear order problem on $G_e \rightarrow$ gives y''
4) Separation **B** (heuristic): Kuratowski constraints, probably goto (1)
5) Branch…

Experiments

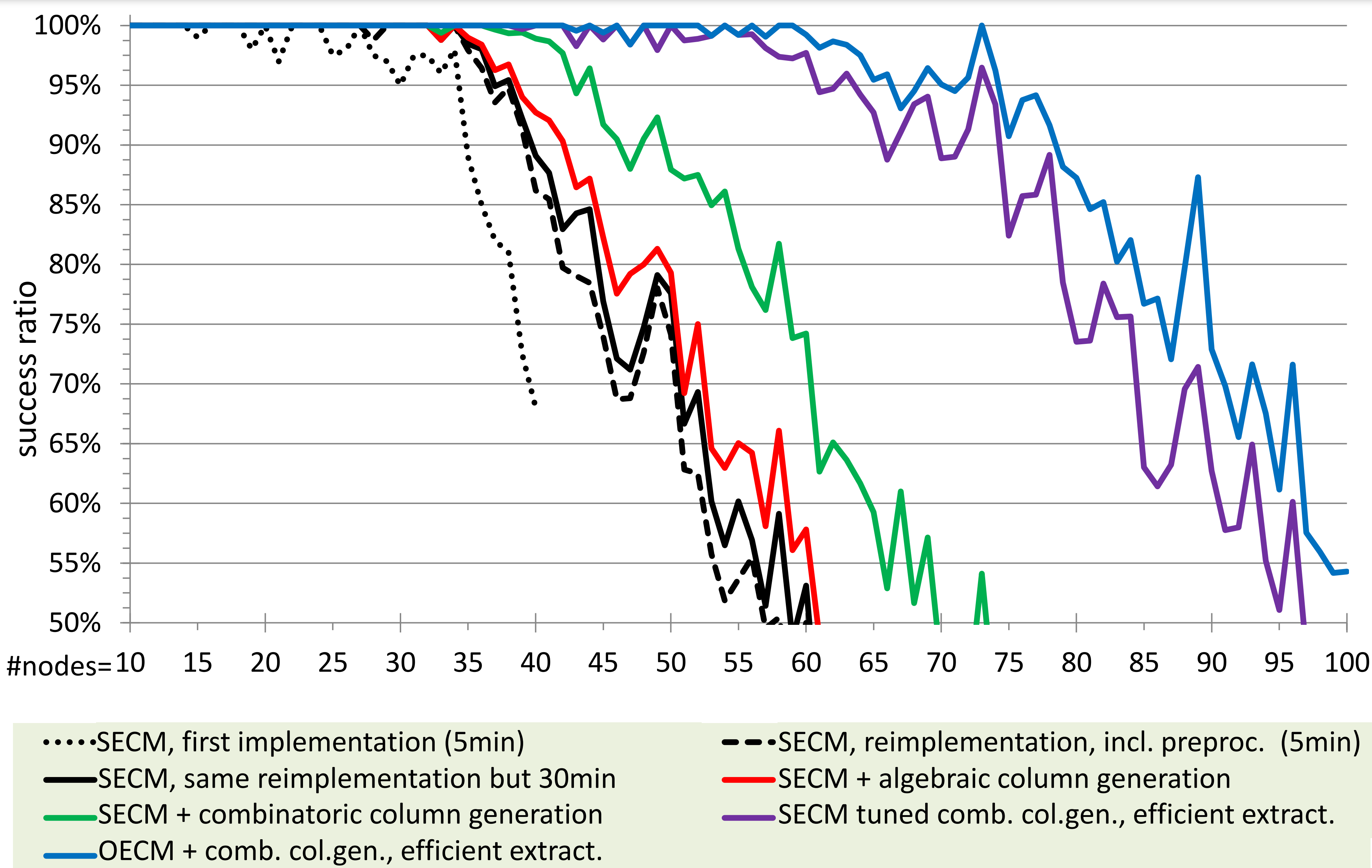
Machine

- AMD Opteron 2.4 GHz, 32bit, 2GB RAM for program
- Open Graph Drawing Framework (OGDF) [GPL]
- Abacus (Branch&Cut&Price-Framework) [LGPL]
- IBM Ilog CPLEX [free for academic use]
- 30 min time-out per graph

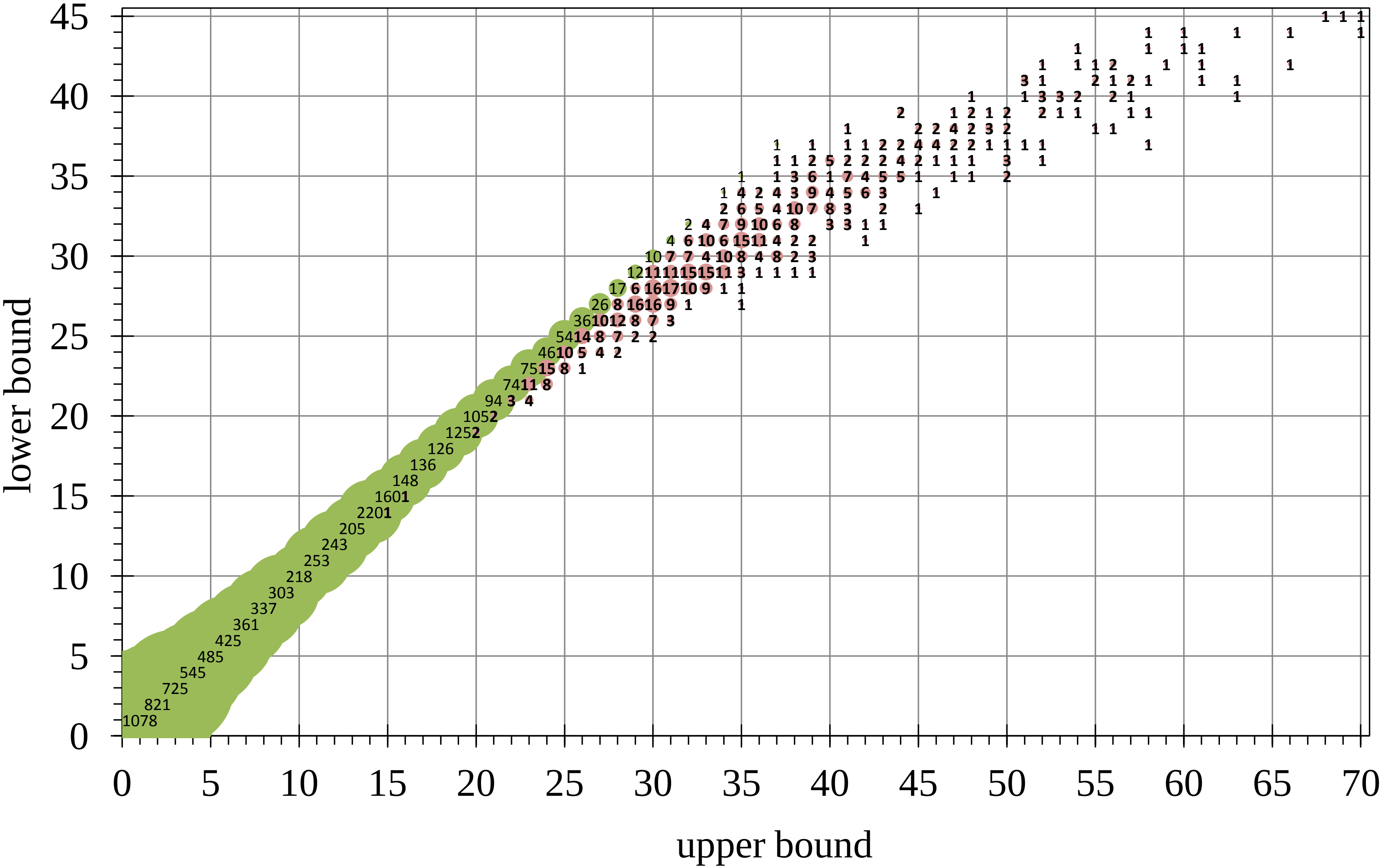
Benchmark

- Rome graph library
- 11.389 “real world” graphs
- 10-100 vertices, average degree of non-planar graphs: 2.7

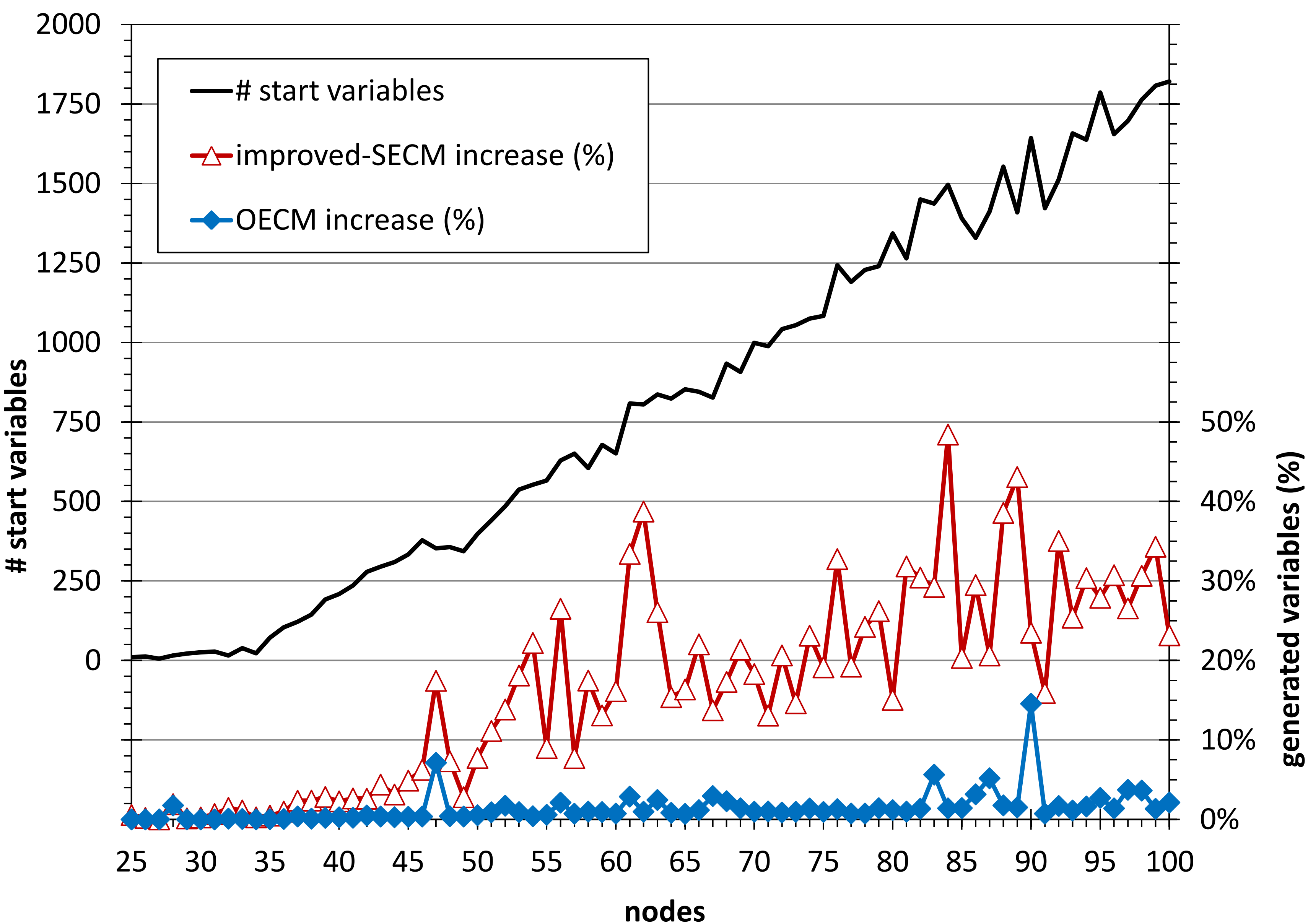
% solved (Rome instances)



Experimente (2)



Required Variables



Observations (for Rome graphs & similar)

- **Planarization heuristic is really good!**
  For instances small enough for the ILP to solve:
    Heuristic typically gives the optimal solution (or 1 off),
    the ILP mainly proves optimality

- **Column generation is crucial!**
  Otherwise: ILP **much** too large to tackle even small problems.
  Only very few *y*-Variables necessary!

- **Kuratowski-constraints seem weak!**
  Many constraints necessary,
  any single constraint raises the lower bound only **very slightly**
  YET: Kuratowski-subdivisions are facets of the polytope! [Ch. 11]

    →**Strong additional constraints would be VERY interesting!**

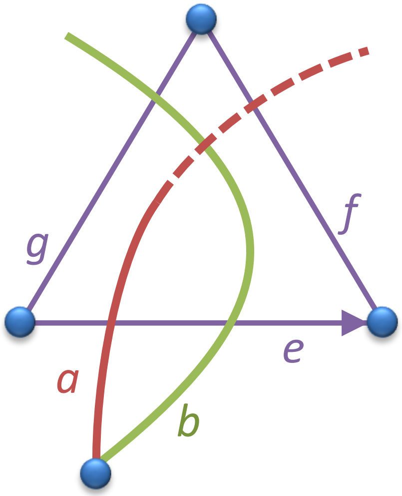
Further constraints

Triangle Constraints

Triangle *e,f,g*

Adjacent edges *a*,*b*

$y_{e,a,b} + x_{\{f,a\}} + x_{\{g,b\}} \leq 2 + x_{\{f,b\}} + x_{\{g,a\}}$



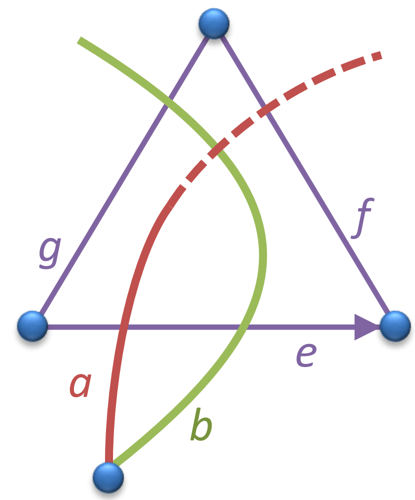
Further constraints

Triangle Constraints

Triangle *e,f,g*

Adjacent edges *a*,*b*

$$y_{e,a,b} + x_{\{f,a\}} + x_{\{g,b\}} \leq 2 + x_{\{f,b\}} + x_{\{g,a\}}$$



Extended Triangle Constraints

Triangle *e,f,g*

Non-adjacent edges *a*,*b*, joined over path *P*

$$y_{e,a,b} + x_{\{f,a\}} + x_{\{g,b\}} \leq 2 + x_{\{f,b\}} + x_{\{g,a\}} + x_{\{a,b\}} + \sum_{e' \in \{e,f,g\}} \sum_{f' \in P} x_{\{e',f'\}}$$

Special graph classes

What when we consider graph classes interesting for **theory** (not practice).

Typical graphs:

- Complete graphs, complete bipartite, Petersen graphs, Toroidal grids, etc.

Common properties:

- Often dense (-r than Rome&Co)
- **Very regular!**
  Symmetric solutions bad for branching $\rightarrow$ symmetry-breaking constraints
- **A lot of structure known!**
  Simple to find „stronger“ subgraphs than K_5, $K_{3,3}$ subdivisions

E.g., Complete Graphs

Use theory-results

Consider some K_{2n+1}. All its solutions have the same parity [Kleitman 76]

$\rightarrow$ Assume we have an upper bound N, then any lower bound >N-2 suffices.

$\rightarrow$ Branch on the parity of the crossings of induces K_5-subdivisions

Symmetry-breaking: 2 Alternatives

Node/Kuratowski Symmetry Constraints

Label the nodes arbitrary 1…n, and let $X(v_i)$ be the crossings on edges incident to v_i.

$$X(v_1) \geq X(v_2) \geq \ldots \geq X(v_n)$$

Edge Symmetry Constraints

Pick arbitrary node as v_1 and label the incident edges arbitrary 1…n-1, and let $X(e_i)$ be the crossings on edges e_i.

$$X(e_1) \geq X(e_2) \geq \ldots \geq X(e_{n-1})$$

$$X(v_1) \geq X(v_i) \qquad \forall i>1$$

E.g., Complete Graphs

Larger Kuratowski constraints

In a $\mathbf{K_n}$ it is trivial to enumerate all $\mathbf{K_{n-1}}$, $\mathbf{K_{n-2}}$,… subgraphs, and we know their crossing numbers:

$$X(\mathbf{K_{n-1}}) \geq cr(\mathbf{K_{n-1}})$$

Add further knowledge, e.g., proof of cr(K_{11})=100 [Pan, Richter 07]

"A good drawing of $\mathbf{K_{11}}$ with fewer than 100 crossings contains a good drawing of $\mathbf{K_{10}}$ with at most 62 crossings. Any good drawing of $\mathbf{K_{10}}$ with at most 62 crossings contains an optimal drawing of $\mathbf{K_9}$. A good optimal drawing of $\mathbf{K_9}$ contains a good drawing of $\mathbf{K_8}$ with at most 20 crossings. Any good drawing of $\mathbf{K_8}$ with at most 20 crossings contains an optimal drawing of $\mathbf{K_7}$."

Still… we need more to solve K_{13}!

Proofs / Certificate

Theory (Computer Proof $\rightarrow$ Certificate)

The ILP algorithm (if implemented totally bug free, using a bug-free LP-solver, complier, computer, etc.) gives a formal proof.

Current Status

Two different ILPs with implementation.
When both are used and they prove the same number…

Next Steps (ongoing)

Extract easy-to-check proofs from the ILP after it was run:

- Case distinction from branch information (leaves of B&B tree)
- For each case: Set of Kuratowski subdivisions
- For each case: Independent/small program to transform each case and Kuratowski set into an LP.
- Use any LP-solver to obtain fractional solution (lower bound to the ILP) which is less then 1 smaller than the assumed optimal solution.

DEMO

Command-line tool

DEMO

http://webcompute.ae.uni-jena.de

(currently in Beta)

Conclusion

Observations

- SECM and OECM are able to solve many „real-world“ graphs to provable optimality
- Even if computation is not successful within our time limits, we still obtain at least upper and lower bounds

Current/Future work

- Implement automatic proof/certification system

Open question

- Certain Kuratowski-constraints define facets (those without the crossing shadow). What about the others?
- Further strengthening constraint classes, either for general graphs or for special graph classes.
  $\rightarrow$ Find something good enough to tackle $\mathbf{K_{13}}$!
- Complete graphs: Realizability is polynomial [Kynčl 07]
  $\rightarrow$ ILP approach solely on x variables?