1^{st} *Parameterized Algorithms & Computational Experiments* Challenge

Where it came from, how it went, who won, and what's next

August 24^{th}, IPEC 2016, Aarhus, Denmark

NET WORKS

WHERE PACE CAME FROM

Inception

- PACE was conceived in fall 2015 when many FPT researchers gathered at the Simons institute
- Born from a feeling that parameterized algorithmics should have a greater impact on practice
- Partially inspired by the success of SAT-solving competitions in neighboring communities
- Discussions with many members of the community (thanks for all your input!) led to a steering committee and two challenge tracks for 2015-2016 with program committees
  - Track A: Treewidth
  - Track B: Feedback Vertex Set

Goals

- Investigate the applicability of algorithmic ideas from parameterized algorithmics

1. provide bridge between algorithm design&analysis theory and algorithm engineering practice
2. inspire new theoretical developments
3. investigate the competitiveness of analytical and design frameworks developed in the communities
4. produce universally accessible libraries of implementations and repositories of benchmark instances
5. encourage dissemination of the findings in scientific papers

PACE organization

Steering committee:

Holger Dell	Saarland University & Cluster of Excellence
Bart M. P. Jansen	Eindhoven University of Technology
Thore Husfeldt	ITU Copenhagen and Lund University
Petteri Kaski	Aalto University
Christian Komusiewicz	Friedrich-Schiller-University Jena
Frances A. Rosamond [*chair*]	University of Bergen

PACE organization

Program committee track A, Treewidth:

Isolde Adler	University of Leeds
Holger Dell [*chair*]	Saarland University and Cluster of Excellence
Thore Husfeldt	ITU Copenhagen and Lund University
Lukas Larisch	University of Leeds
Felix Salfelder	Goethe University Frankfurt

Program committee track B, Feedback Vertex Set:

Falk Hüffner	*Industry*
Christian Komusiewicz	Friedrich-Schiller-University Jena

PACE timeline in 2015-2016

- March 1st 2016: Call for contributions, benchmark instances available, website online
- June 1st 2016: Register participation
- June 22nd 2016: Prizes and travel awards announced, sponsored by *Networks*
- August 1st 2016: Submission deadline
- August 24th 2016: Winner announcement

pacechallenge.wordpress.com

A word from the sponsor …

- We are offering a 2-year postdoc position in Network Algorithms at the Eindhoven University of Technology
  - Broad range: computational geometry, graph algorithms, or FPT algorithms
  - Contact Mark de Berg (m.t.d.berg@tue.nl) before August 31

NETWORKS is a project of
University of Amsterdam
Eindhoven University of Technology
Leiden University
Center for Mathematics and
Computer Science (CWI)

NETWORKS

thenetworkcenter.nl

How it went and who won

TRACK A: TREEWIDTH

PACE 2016
Track A: Tree width
…

Isolde Adler
Holger Dell
Thore Husfeldt
Lukas Larisch
Felix Salfelder

PACE challenges, Track A

exact tree width

Evaluation: The running time

3 submissions

heuristic tree width

Evaluation: The obtained width

7 submissions

instances

2 submissions

Treewidth

Given G and k, is tw(G) $\leq$ k ?

- NP-hard, but in time n^{k+2}
  (Arnborg, Corneil & Proskurowski 1987)

- in FPT time $\exp(k^3)\ n$
  (Bodlaender 1996)

- factor-5 approximation in time $\exp(k)\ n$
  (Bodlaender Drange Dregi Fomin Lokshtanov Pilipczuk 2013)

- open: PTAS?

Some Applications (outside of FPT)

- Register allocation in compilers
  (e.g., Thorup 1998)
- Preprocessing for shortest path
  (e.g., Chatterjee Ibsen-Jensen Pavlogiannis 2016)
- Treewidth of specific graph families
  (e.g., Kiyomia Okamotob Otachic 2015)
- Preprocessing for probabilistic inference
  (e.g., Otten Ihler Kask Dechter 2011)

Treewidth implementations pre-PACE

- Python SAGE: slow and buggy
- Outdated C++-library without documentation
- Some non-public implementations
- No standard input/output format
- Hard to compare

The submission requirements

- repository on github.com
- 2-page abstract
- DIMACS input format
- Output: tree decomposition

Benchmark instances

96 control flow graphs
79 special “named” graphs
56 DIMACS graph coloring instances
41 random instances
7 incidence graphs of SAT competition instance
2 transit networks
281 total

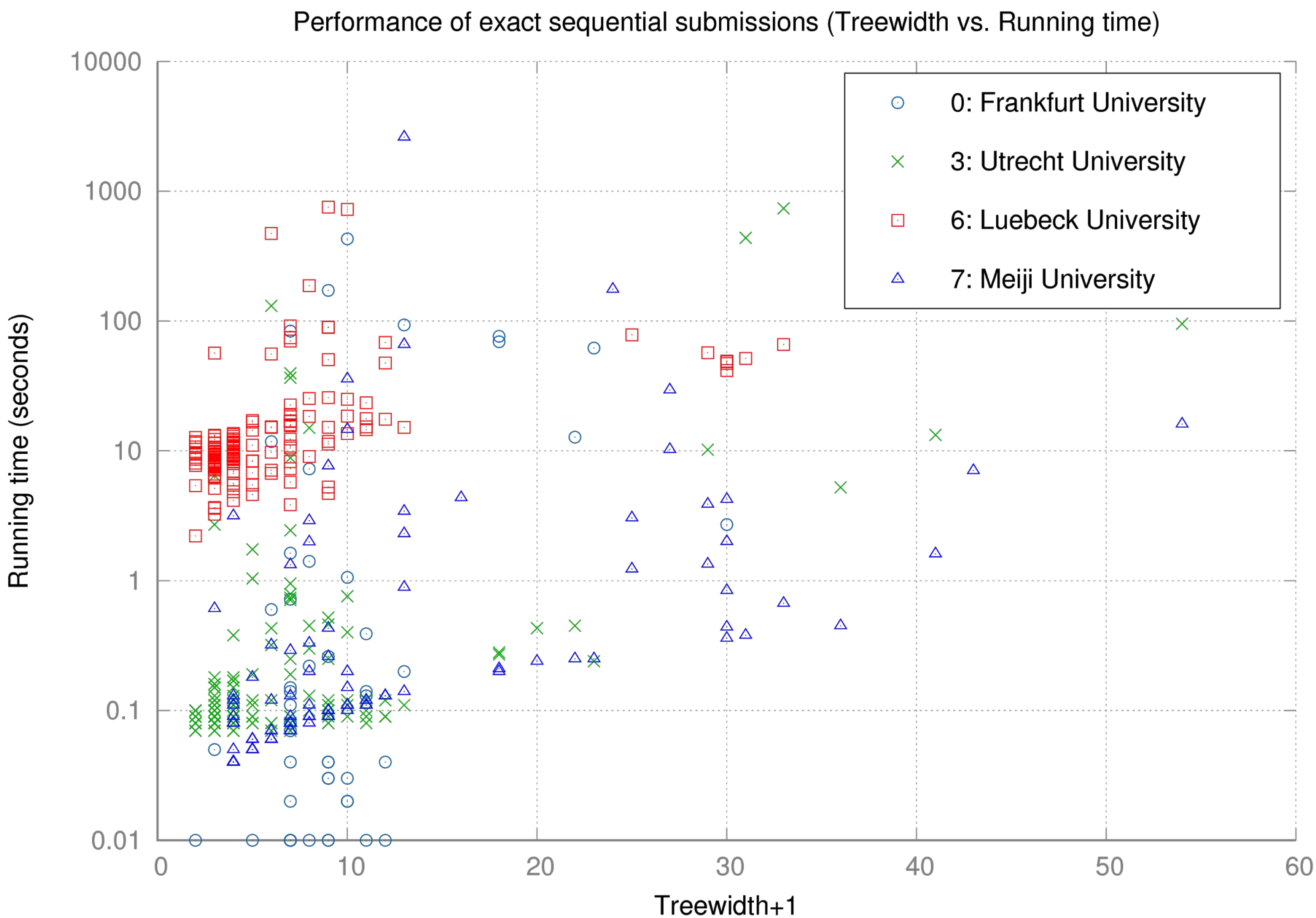
Detailed results, benchmark instances, and tools to easily reproduce the results:
https://github.com/holgerdell/PACE-treewidth-testbed

Submission programming languages

- C++-11
- C# / Mono
- Java 8

Exact treewidth

Exact Treewidth Competition Results



Exact Treewidth Competition Results

instances solved in timeout:

166 Berndt, Bannach, Ehlers (Universtität zu Lübeck)
171 Larisch & Salfelder (baseline)
173 Bodlaender & Van der Zanden (Utrecht University)
199 Tamaki (Meiji University)

Algorithmic ideas

Use SAT-solver to find elimination order (**Team Lübeck**)

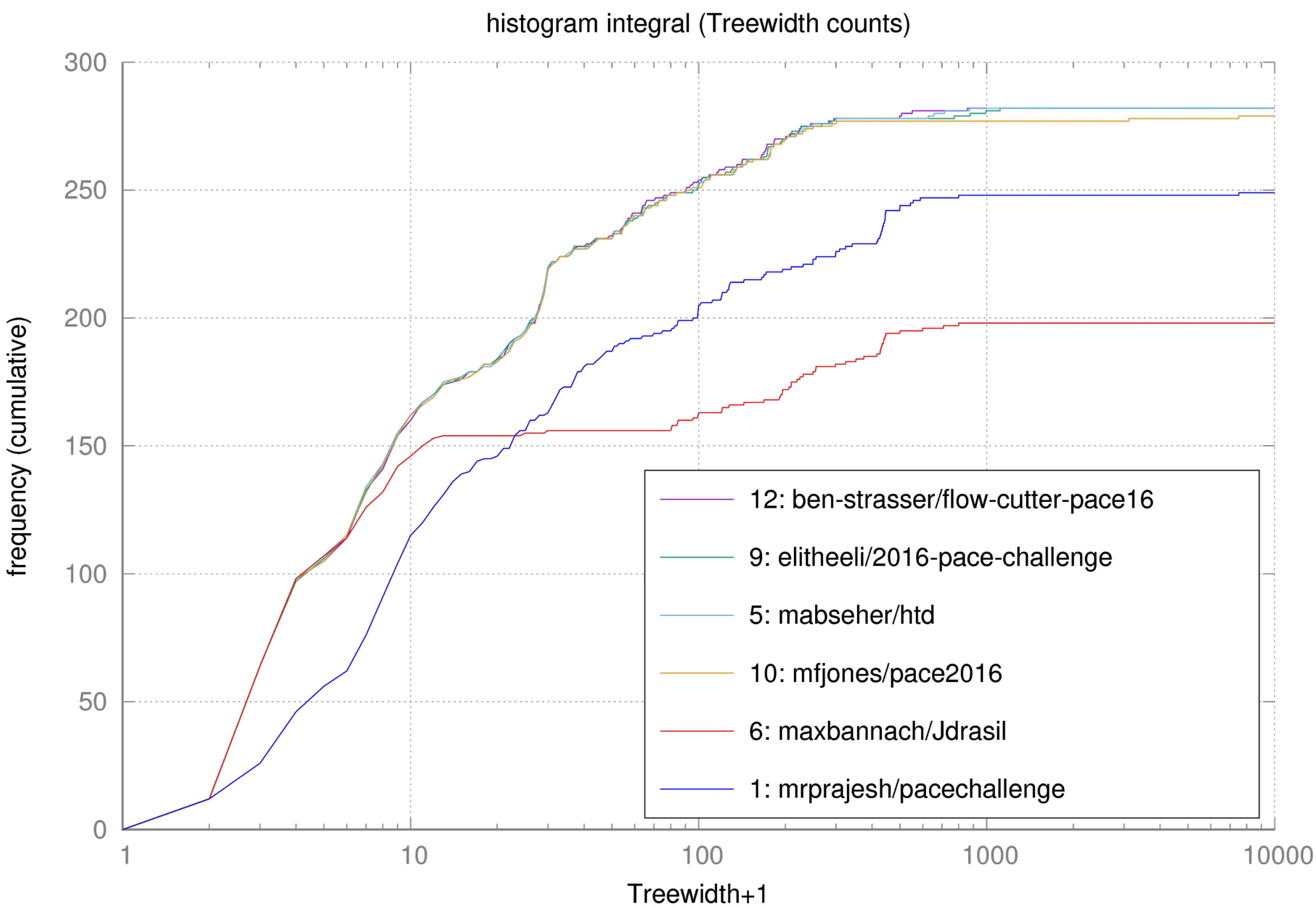
Branch on balanced separators + DP (**Team Utrecht**)

Tamaki:

- Modify n^k brute-force approach of Arnborg et al. (1987) in an upcoming publication
- Running time not known to be in $n^{f(k)}$

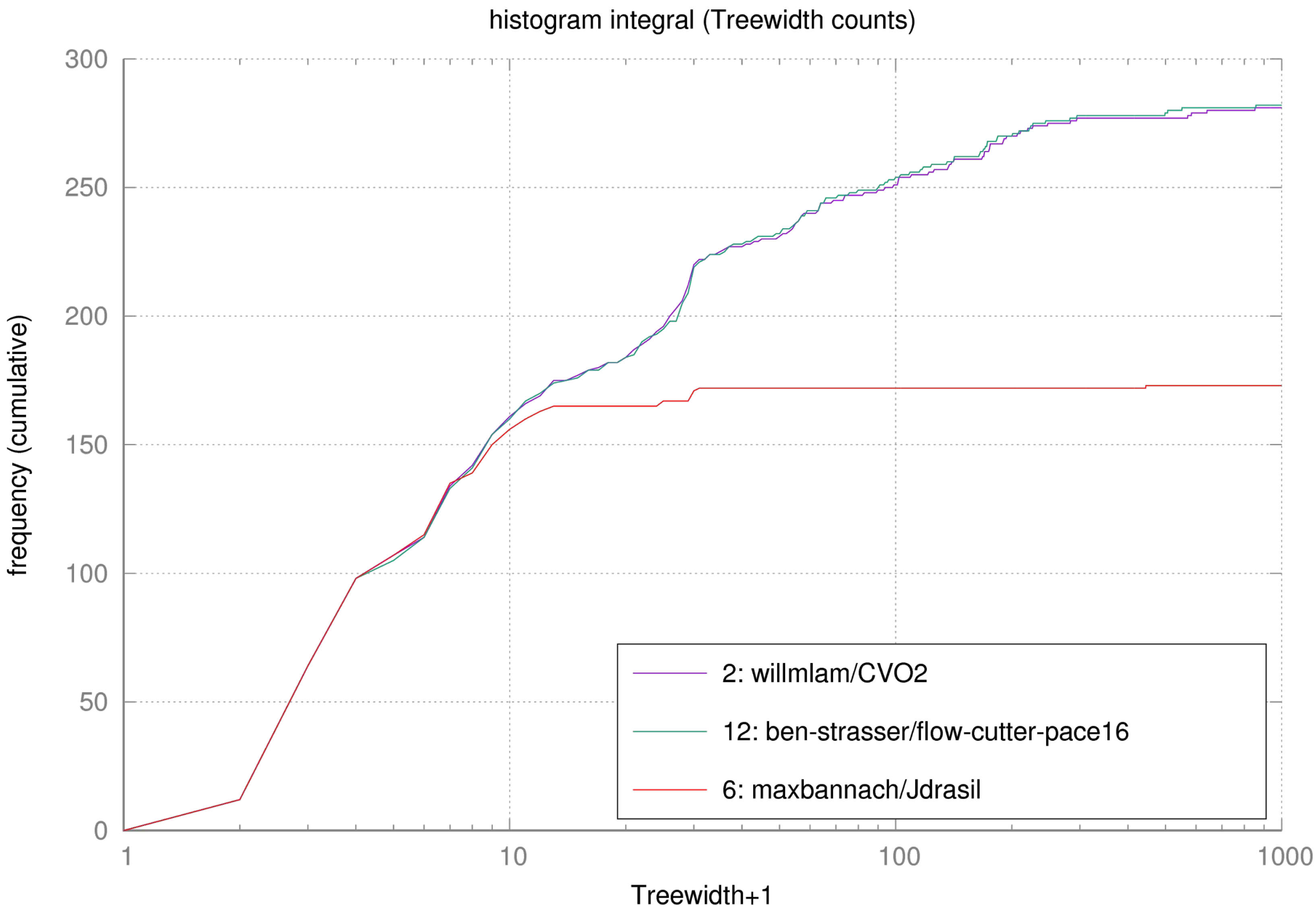
Heuristic treewidth

Heuristic Sequential Treewidth Competition



Heuristic Sequential Treewidth Competition

Heuristic Parallel Treewidth Competition



Evaluation Scheme

6s11-opt.gaifman.gr

submission	width after 100s
5	672
12	957
9	994
1	33279
10	33279

Preferential voting scheme

Instances=Voters

Use Schulze method to combine votes

Heuristic Competition Results

Sequential algorithm

1. Ben Strasser
   (Karlsruhe Institute of Technology)
2. Eli Fox-Epstein
   (Brown University)
3. Abseher, Musliu, Woltran
   (TU Wien)

Parallel algorithm

1. Kask, Lam
   (University of California at Irvine)
2. Ben Strasser
   (Karlsruhe Institute of Technology)
3. Bannach, Berndt, Ehlers
   (Universität zu Lübeck)

Condorcet Winners

Heuristic sequential:

12 (Strasser) better than

1 (IIT Madras)	on 100% of instances
6 (Lübeck)	on 95.5% of instances
10 (Australia)	on 71% of instances
5 (TU Wien)	on 61% of instances
9 (Fox-Eppstein)	on **55%** of instances

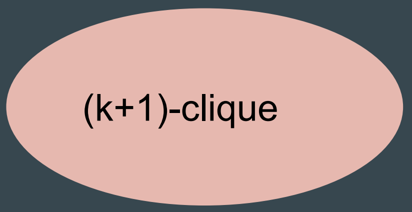
Heuristic parallel:

2 (UC Irvine) better than

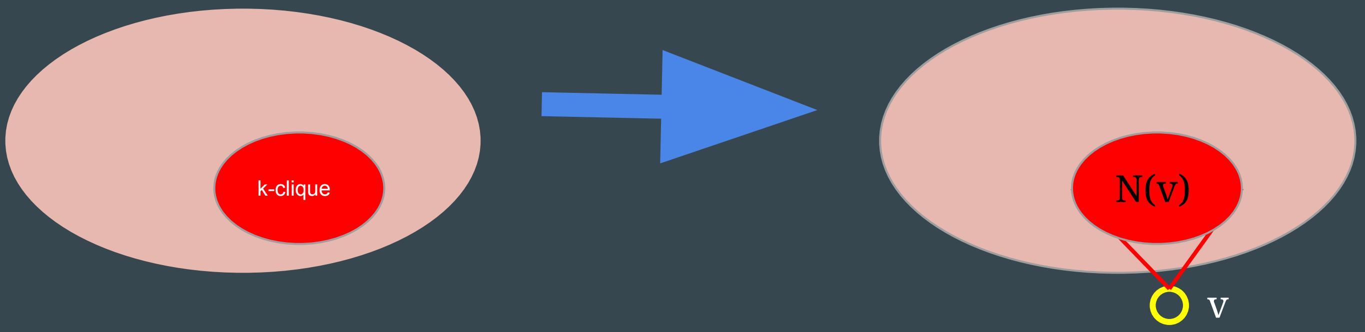
6 (Lübeck) on 99% of instances
12 (Strasser) on **63%** of instances

Definition of k-Trees

1.



2.



subgraphs of k-trees = treewidth k graphs

elimination order: reverse of insertion order

Main Algorithmic Ideas for Heuristic TW

Minimum Fill-In Heuristic

Guess elimination order:

- Choose vertex v randomly so that
  few edges need to be added to turn N(v) into clique

Team Australia (rank 4)

"Turbocharging treewidth heuristics" (IPEC 2016)

PACE challenges, Track A

exact tree width

Evaluation: The running time

3 submissions

heuristic tree width

Evaluation: The obtained width

7 submissions

instances

2 submissions

How it went and who won

TRACK B: FEEDBACK VERTEX SET

The 1st Parameterized Algorithms and Computational Experiments Challenge: Track B Feedback Vertex Set

Falk Hüffner
Technische Universität Berlin

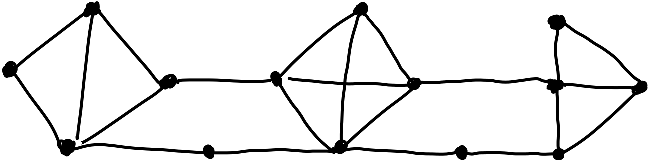
Christian Komusiewicz
Friedrich-Schiller-Universität Jena

Challenge Problem

Feedback Vertex Set

Input: An undirected graph $G = (V, E)$.

Task: Find a minimum set $S \subseteq V$ such that $G - S$ is a forest.

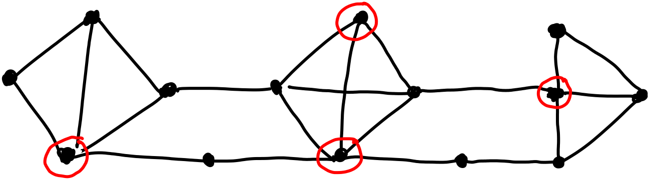


Challenge Problem

Feedback Vertex Set

Input: An undirected graph $G = (V, E)$.

Task: Find a minimum set $S \subseteq V$ such that $G - S$ is a forest.

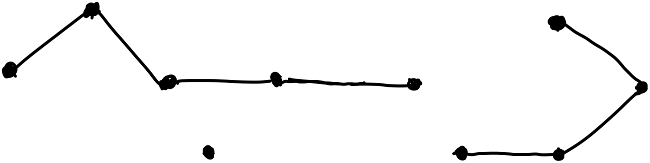


Challenge Problem

Feedback Vertex Set

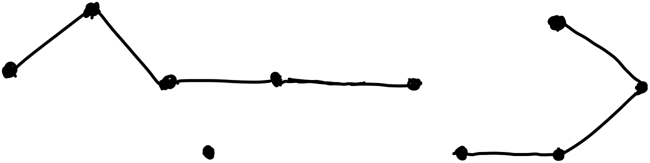
Input: An undirected graph $G = (V, E)$.

Task: Find a minimum set $S \subseteq V$ such that $G - S$ is a forest.



Challenge Problem

Feedback Vertex Set
Input: An undirected graph $G = (V, E)$.
Task: Find a minimum set $S \subseteq V$ such that $G - S$ is a forest.



Feedback Vertex Set is fixed-parameter tractable e.g. parameterized by solution size $|S|$, amenable to different techniques: branching, iterative compression, kernelization, randomized branching,...

Challenge Setup

Benchmark Instances: 230 instances, 100 public instances and 130 hidden instances

Instance origin: Social networks, biological networks, incidence graphs of CNF formulas, road networks, power networks

Challenge Setup

Benchmark Instances: 230 instances, 100 public instances and 130 hidden instances
Instance origin: Social networks, biological networks, incidence graphs of CNF formulas, road networks, power networks
Properties: (hidden benchmark instances)

Challenge Setup

Benchmark Instances: 230 instances, 100 public instances and 130 hidden instances

Instance origin: Social networks, biological networks, incidence graphs of CNF formulas, road networks, power networks

Properties: (hidden benchmark instances)

	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert S\rvert$
min	32	63	5
median	308.5	1305	34
$\varnothing$	2079	4185	153
max	19362	32081	6400

Winner Criterion: # solved instances within 30 minutes (each) on the set of hidden instances

Challenge Setup

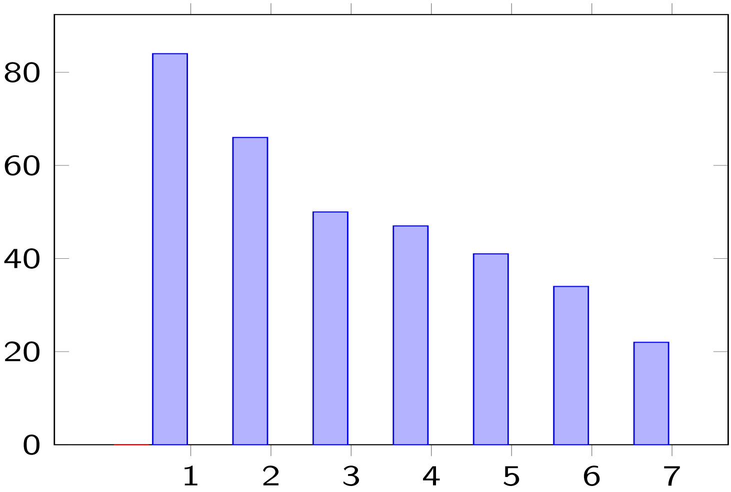
Benchmark Instances: 230 instances, 100 public instances and 130 hidden instances
Instance origin: Social networks, biological networks, incidence graphs of CNF formulas, road networks, power networks
Properties: (hidden benchmark instances)

	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert S\rvert$
min	32	63	5
median	308.5	1305	34
$\varnothing$	2079	4185	153
max	19362	32081	6400

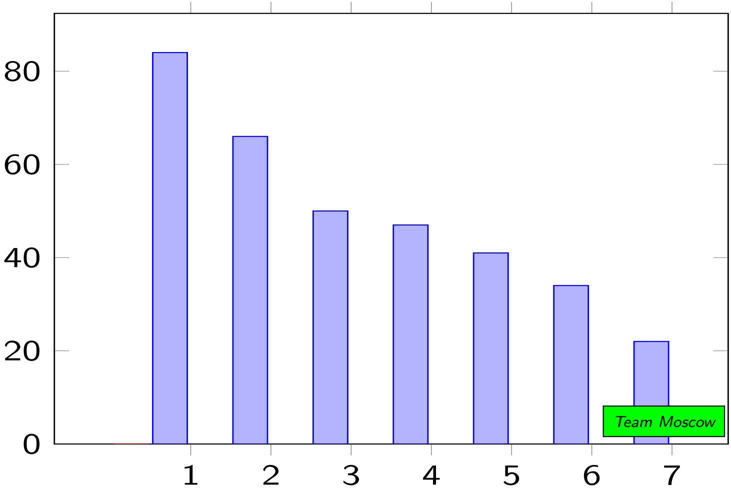
Winner Criterion: # solved instances within 30 minutes (each) on the set of hidden instances

Participation: 14 registrations, 7 submissions

Results

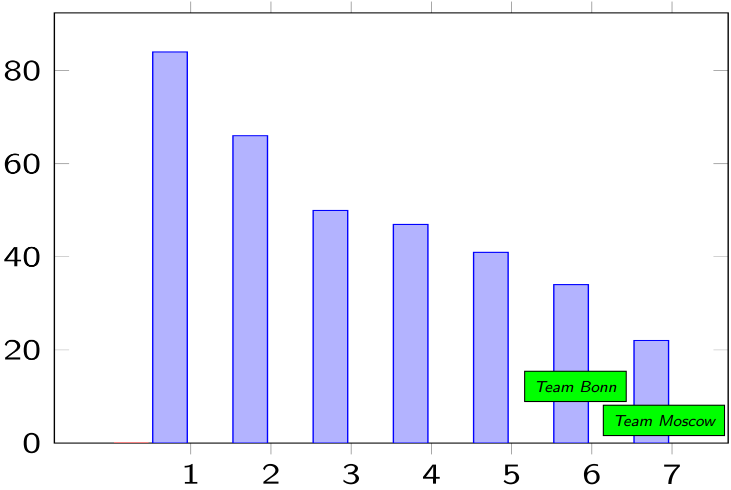


Results



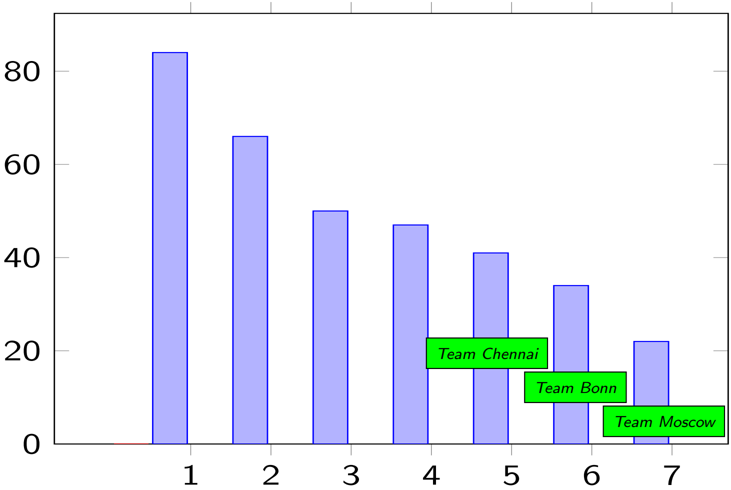
1 participant, C++
randomized branching

Results



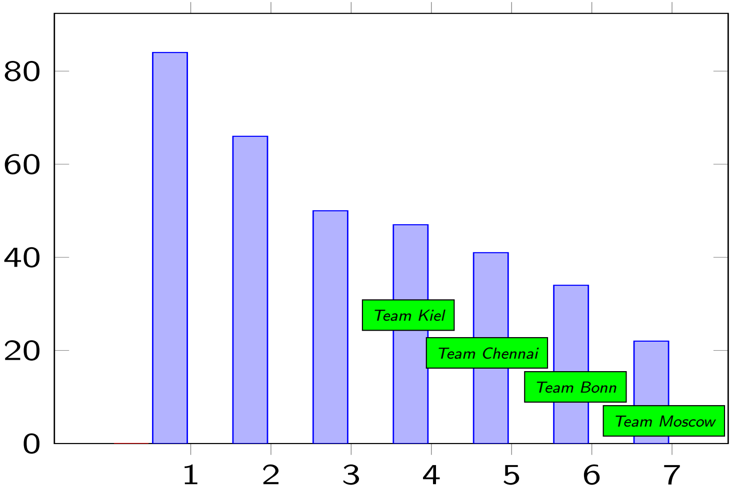
5 participants, C++
iterative compression, subcubic graphs $\in P$

Results



4 participants, Python
branching on shortest cycle, search tree pruning via lb

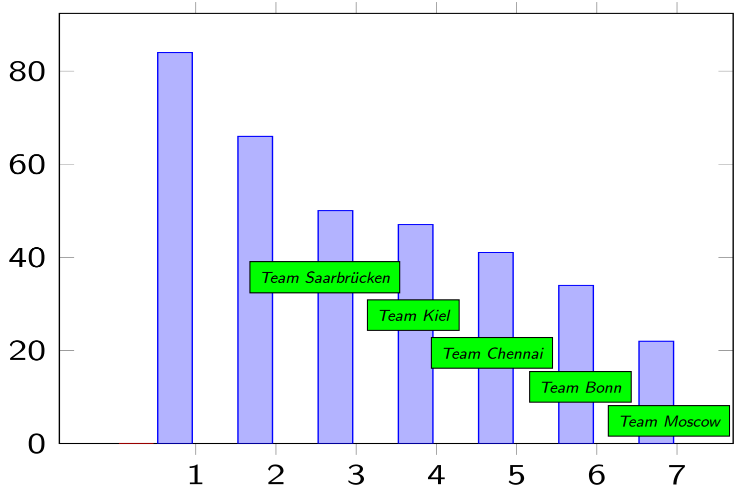
Results



4 participants, C#

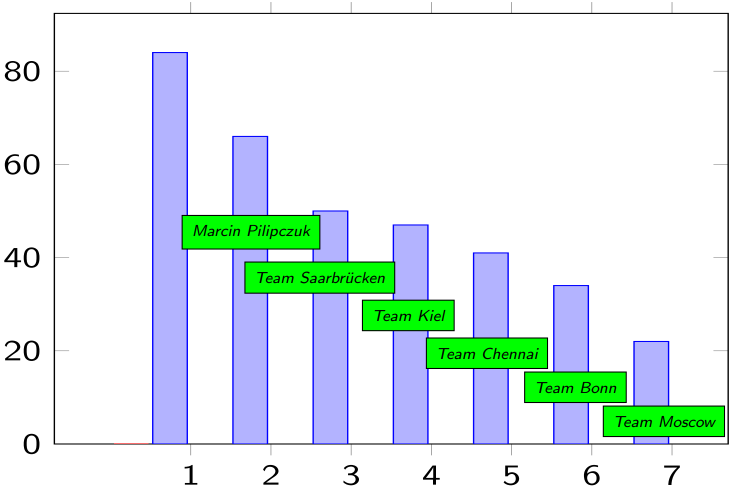
iterative compression & branching, subcubic graphs $\in P$

Results



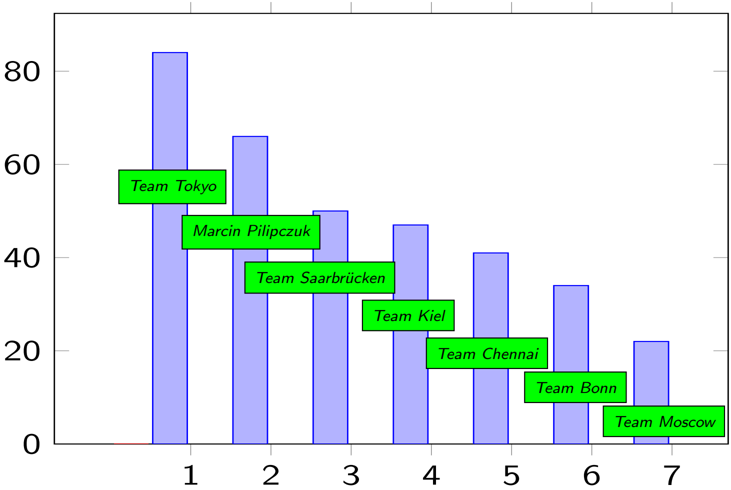
5 participants, C++,
branching, search tree pruning via lower and upper bounds

Results



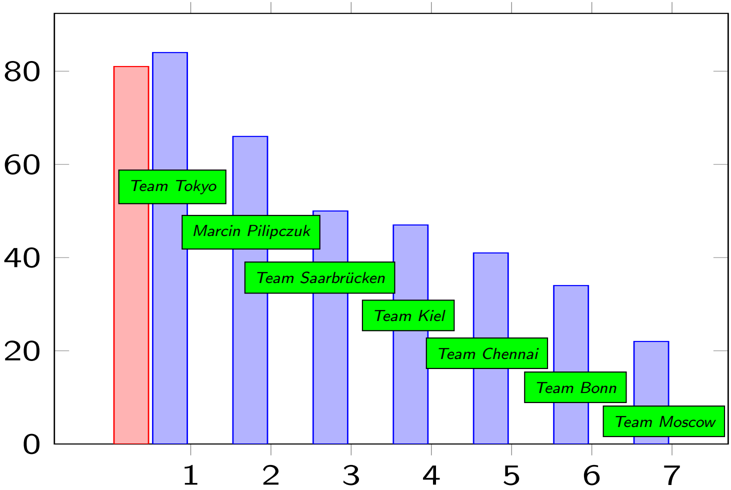
1 participant, C++,
branching, subcubic graphs $\in P$, DP on tree decomposition

Results



2 participants, Java,
LP-based branching and kernelization

Results



ILP, gurobi

data reduction, lazy constraints adding short remaining cycles

WHAT’S NEXT?

Long term plan

- Have a PACE challenge every year to continually drive the transition from theory to practice
  - Challenge problems may change from year to year
- PACE does not aim to be a publication venue for papers
  - Authors of submissions are encouraged to submit papers describing their implementations to established venues (IPEC, ESA track B, ALENEX, etc.)
- Desire to have the award ceremony at IPEC every year
  - (To be discussed with IPEC steering committee)

PACE 2016-2017

- PACE will again have two tracks next year

1. Treewidth track
    - Similarly to this year but without a subtrack for parallel algorithms
2. Track for “Problem X”
    - Problem still to be determined, to be solved exactly by FPT methods

- Time schedule:

1. November 1st 2016: Announcement of problems and inputs
2. March 1st 2017: Submission of prototype program to check input/output formats
3. May 1st 2017: Submission of final program
4. June 1st 2017: Result are announced
5. Early September 2017: Award ceremony at IPEC

Input from the community

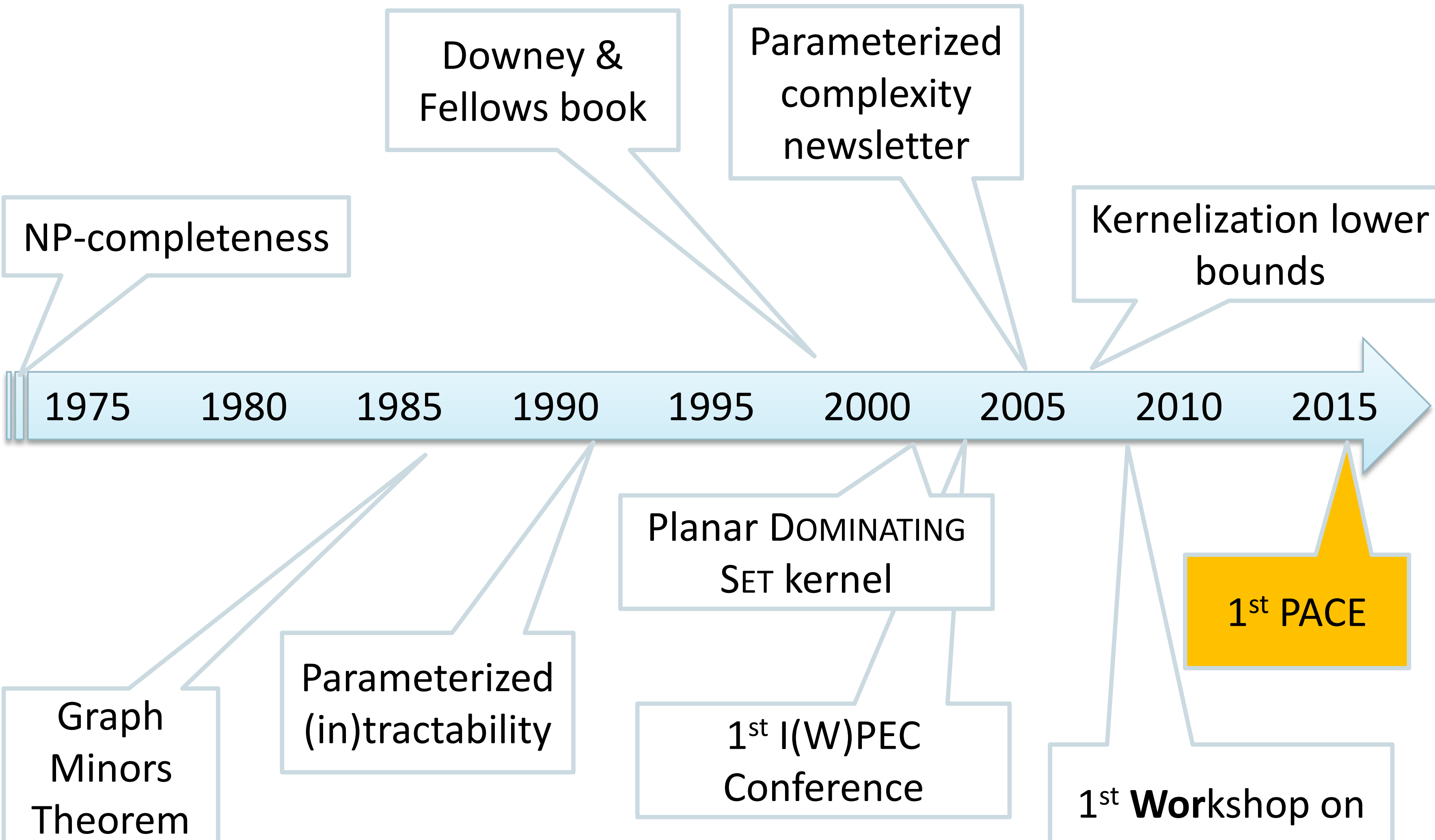
- Which “problem X” to use for the second track next year?
- Preferably, problem X:

1. Has been analyzed successfully from the theoretical perspective, with several different approaches for obtaining FPT algorithms
2. Is relevant in practice and it is possible to find real-world instances with moderate parameter values

Feedback

- Comments? Suggestions? Tips?

History of parameterized complexity



THANK YOU!