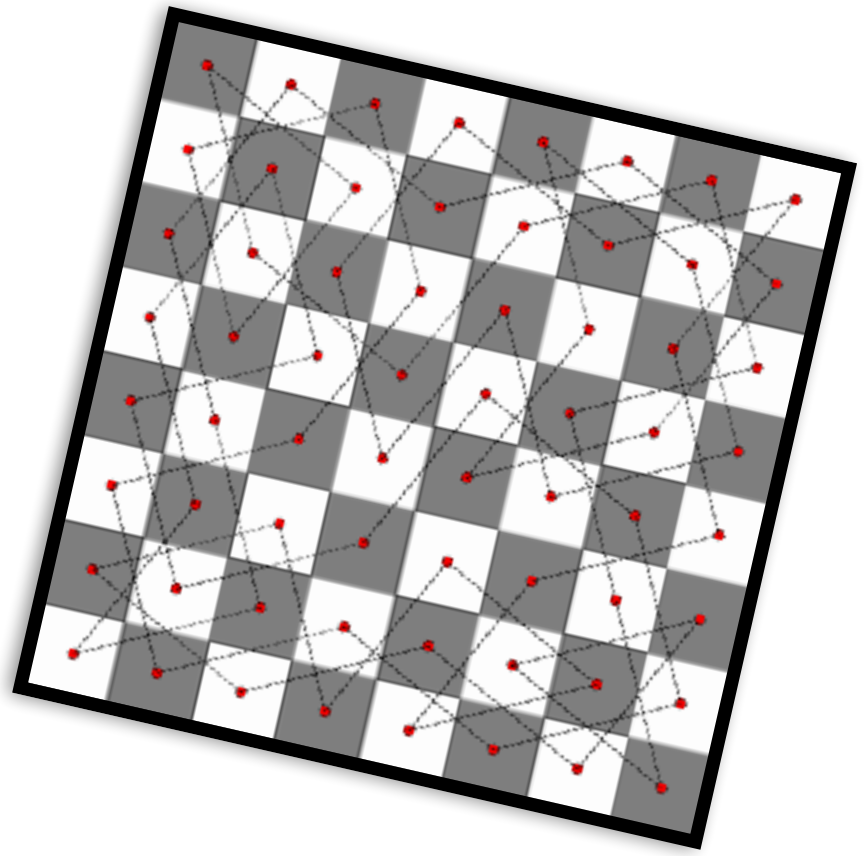
Benchmarking Optimization

LINFO2266

Pierre Schaus



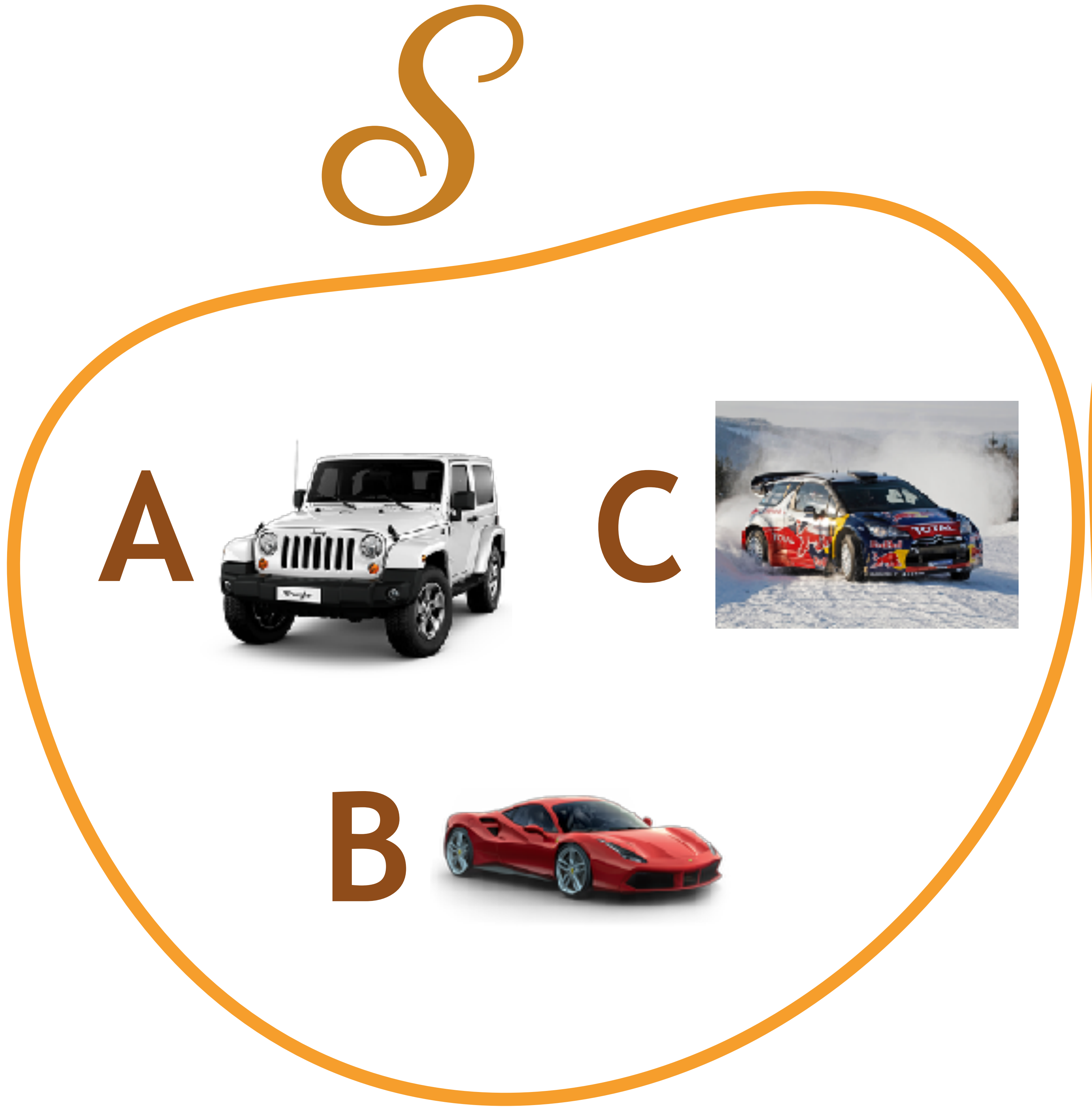
Benchmarking Optimization

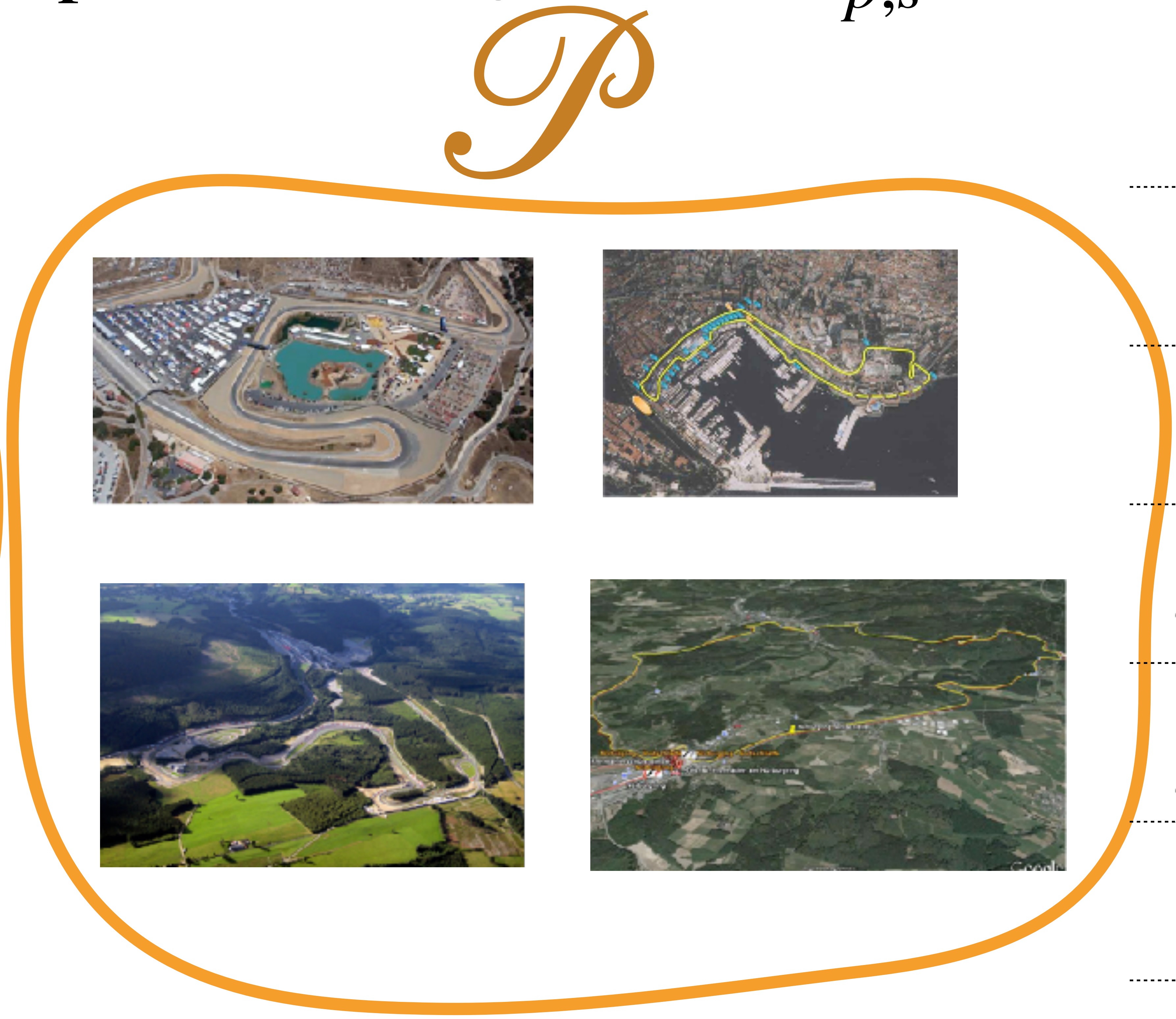
- You have a set of approaches or algorithms to solve an optimization problem.
- You have a set of benchmark instances (supposed to be representative of an unknown future instance).
- How do you compare those algorithms?

Evaluation through Benchmarking

- $\mathscr{P}$ is the set of problems, $\mathscr{S}$ is the set of solvers , $n_p = |\mathscr{P}|$, $n_s = |\mathscr{S}|$

- For each solver s, problem p, computing times = $t_{p,s}$





Instance	A	B	C
	100	10	15
	30	7	12
	40	45	35
	50	55	40
	10	30	15
	20	50	25

Which car is the best?

Objectives of evaluating different solvers/strategies

- Derive general conclusions provided the benchmarks are representative enough
- Quantify how much faster/slower the solver is compared to others

- Estimate the probability that a solver is the best on any unseen instance (requires statistical testing, not covered)

Proposal 1: The average

- Computing the average computing time over all instances?

Most difficult instances dominates ❌

Discarding Unsolved Instances

Bias towards most robust solvers ❌

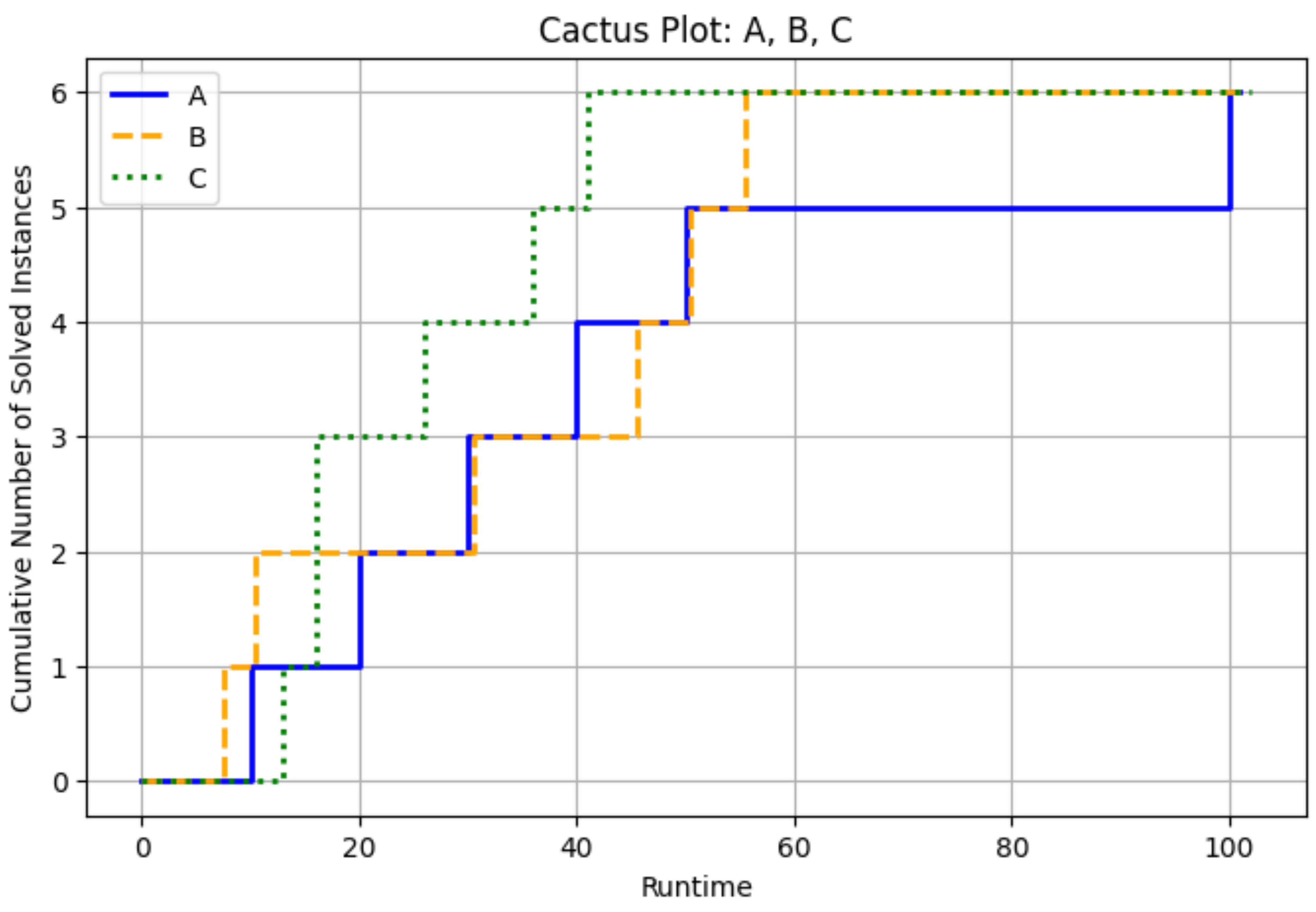
Proposal 2: Count solved instances

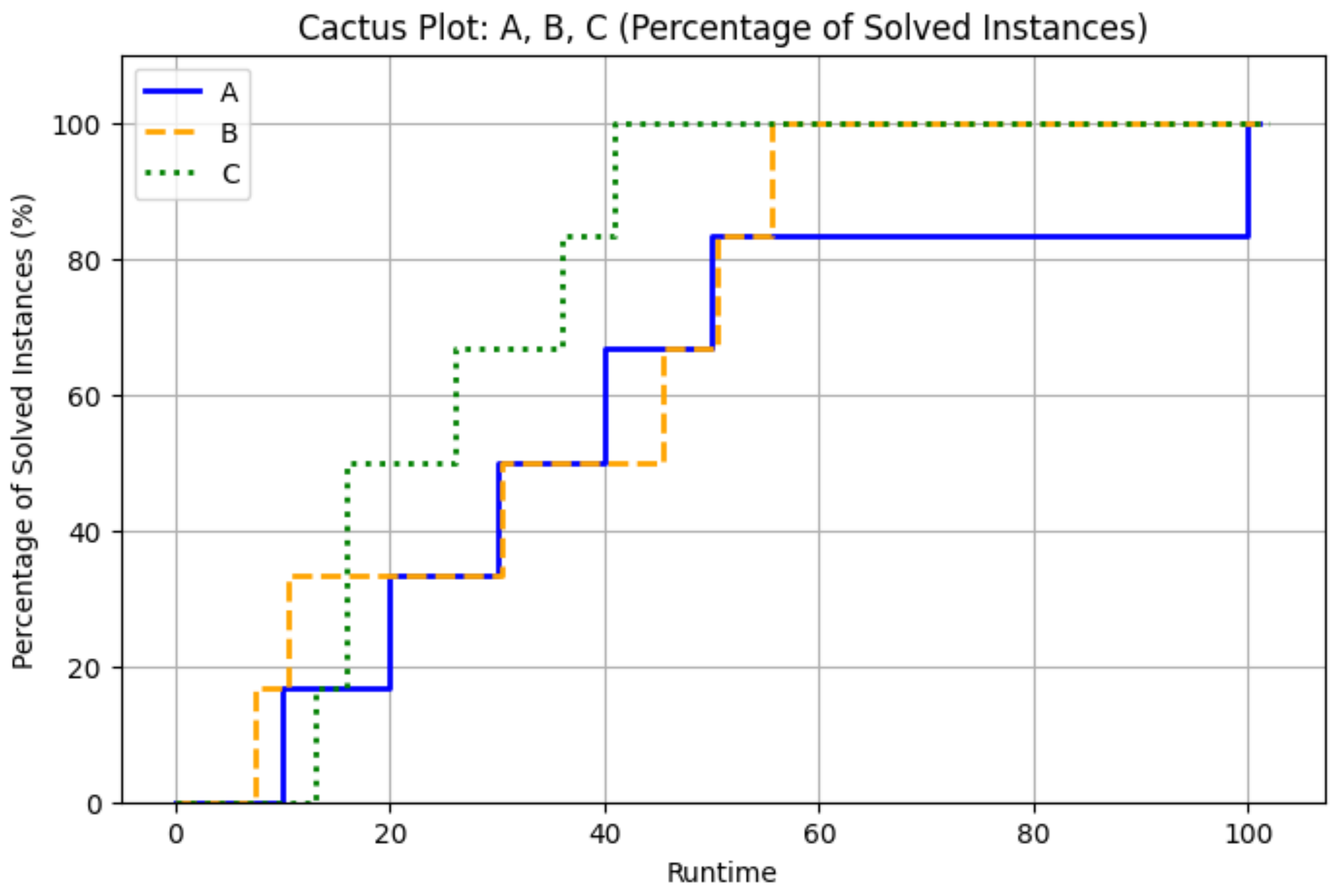
- What about ranking the solvers by number of solved instances?
  - Very coarse grained ❌
  - Relative order per instance is lost ❌

Proposal 3: Cactus Plot

- Fine grained ✅
- Relative order per instance is lost ❌

$$\kappa_s(\tau) = \frac{1}{n_p}\text{size}\{p \in \mathcal{P} : t_{p,s} \leq \tau\}$$





Proposal 4: ratio to *virtual best*

Virtual best

Instance	A	B	C	min	$r_{i,A}$	$r_{i,B}$	$r_{i,C}$
	100	10	15	10	10.0	1.0	1.5
	30	7	12	7	4.3	1.0	1.7
	40	45	35	35	1.1	1.3	1.0
	50	55	40	40	1.3	1.4	1.0
	10	30	15	10	1.0	3.0	1.5
	20	50	25	20	1.0	2.5	1.3

ratio to virtual best

- $$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in \mathcal{S}\}}$$

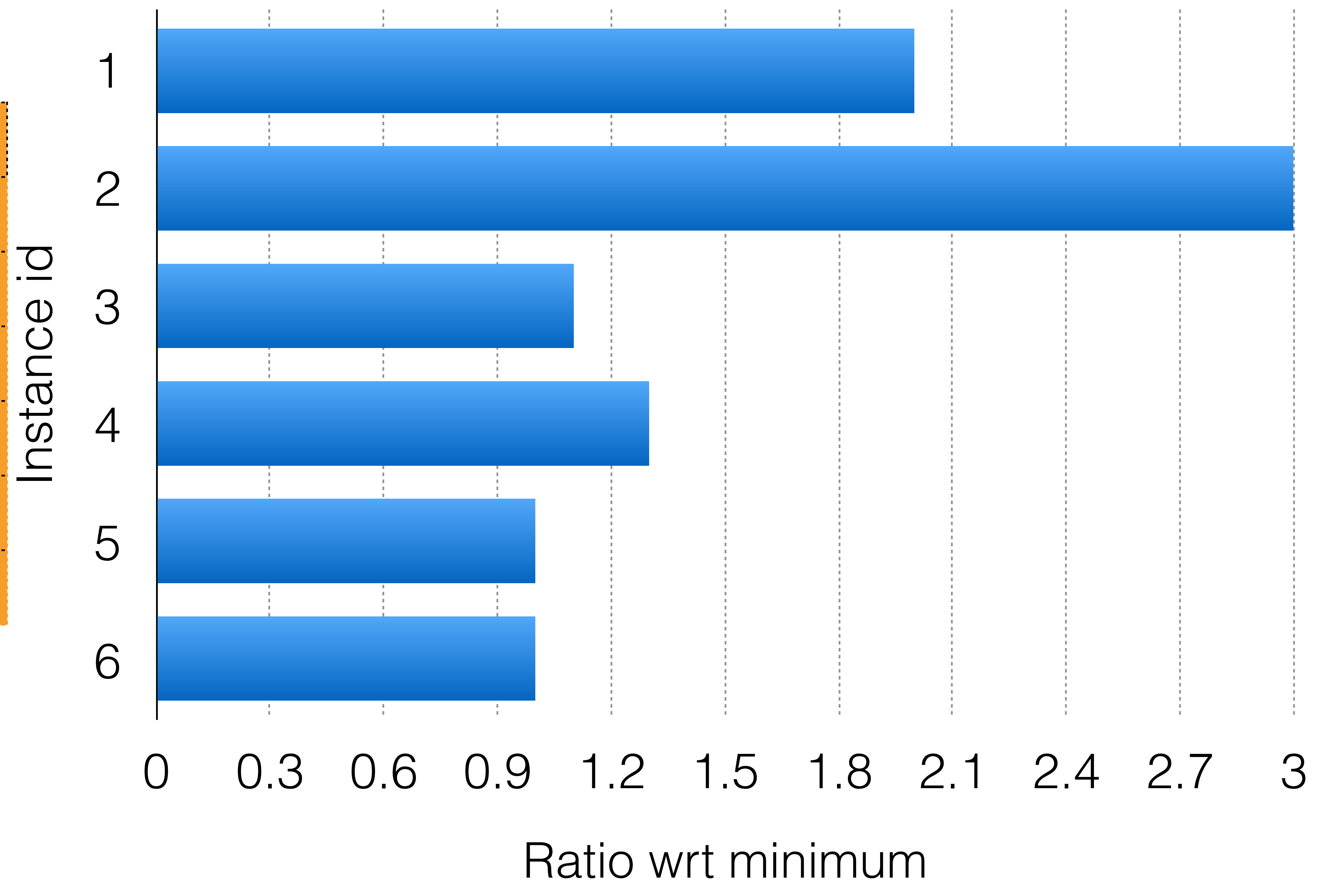
Dolan, E. D., & Moré, J. J. (2002). **Benchmarking optimization software with performance profiles**. Mathematical programming, 91(2), 201-213.

Bar-Plot of the ratio?

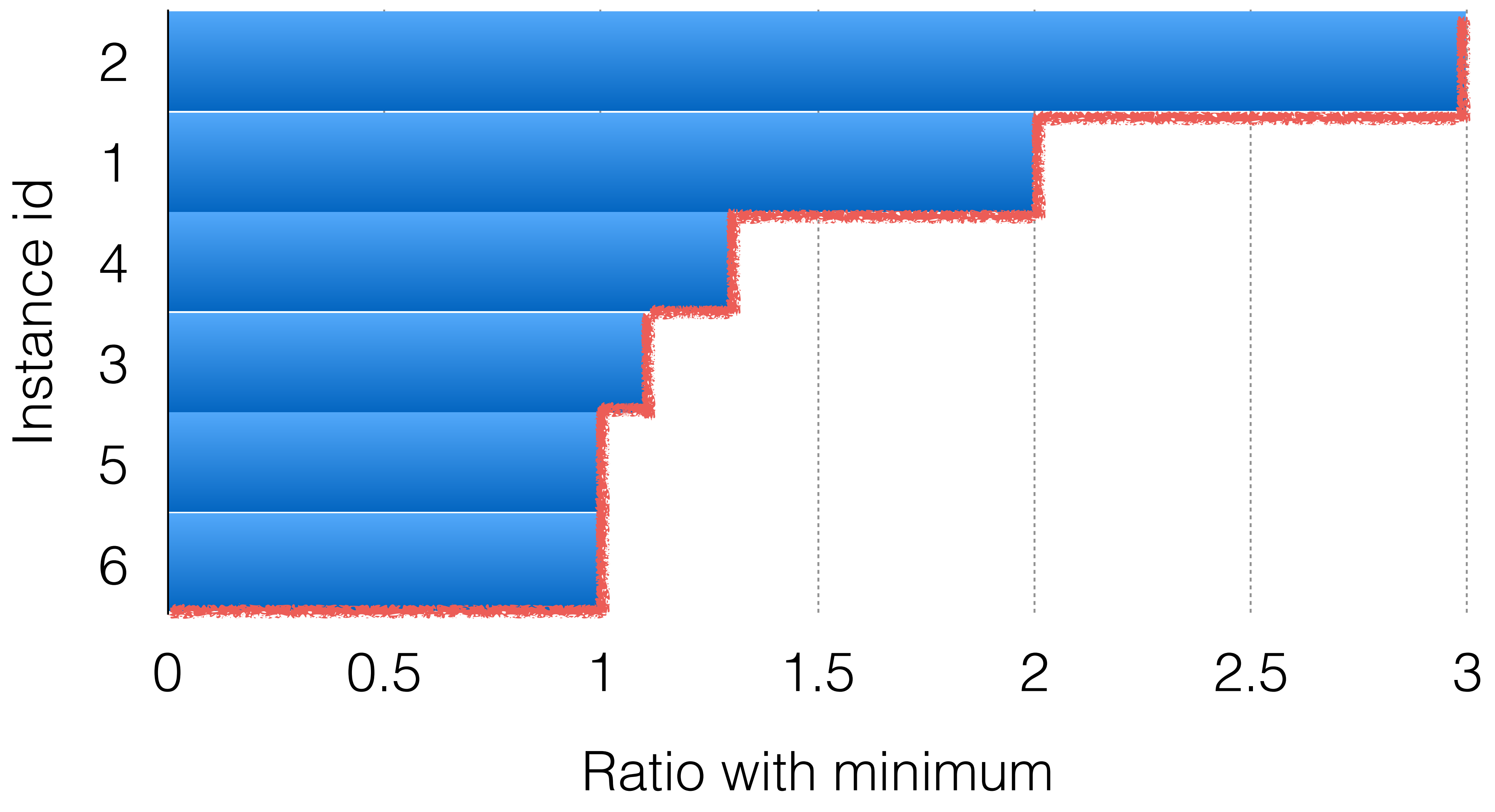
Solver A

Instance id	A	min	$r_{i,A}$
1	20	10	2.0
2	21	7	3.0
3	40	35	1.1
4	50	40	1.3
5	10	10	1.0
6	20	20	1.0

Not very informative 🤔

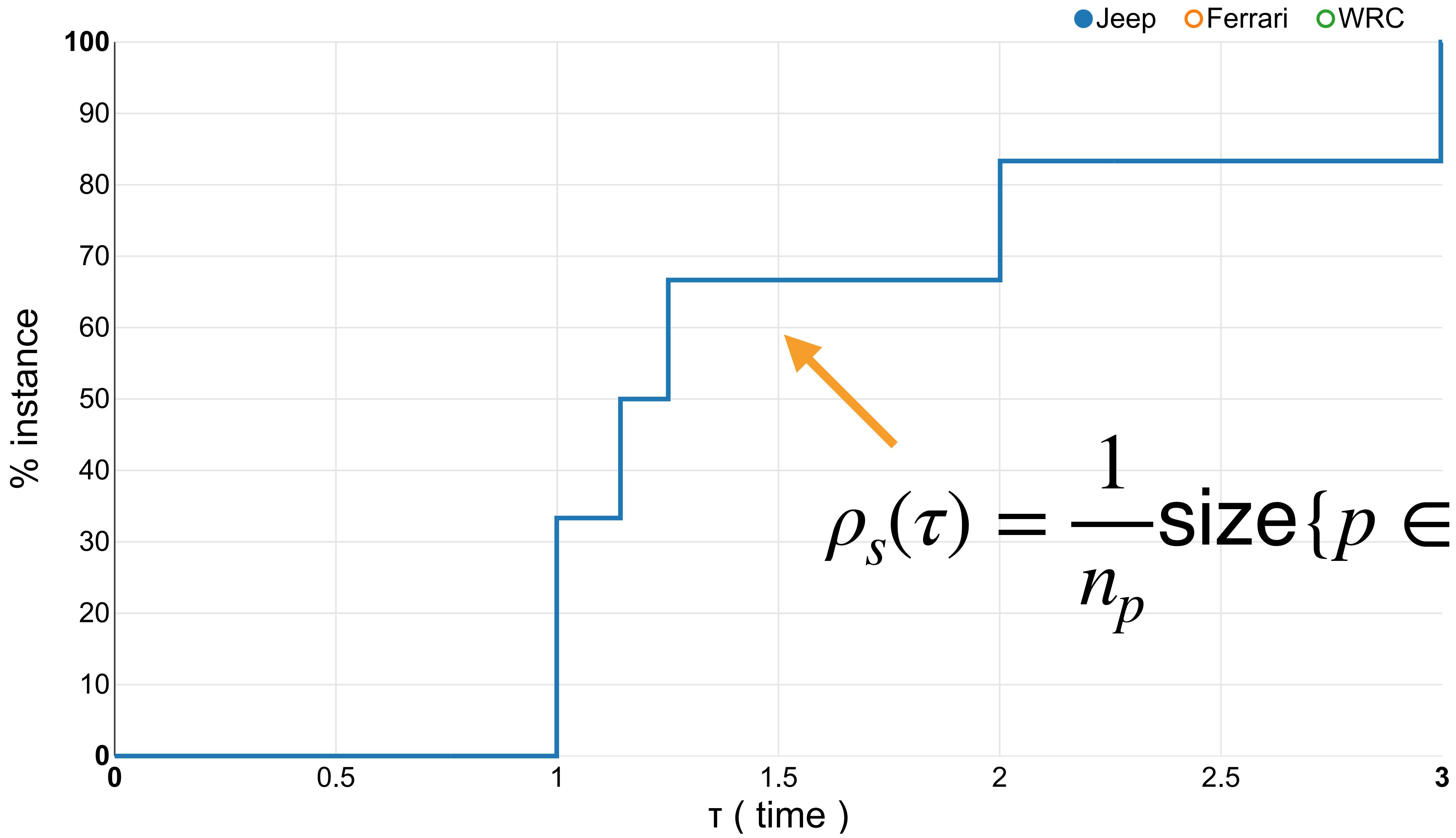


Performance Profile = Profile of virtual best ratios



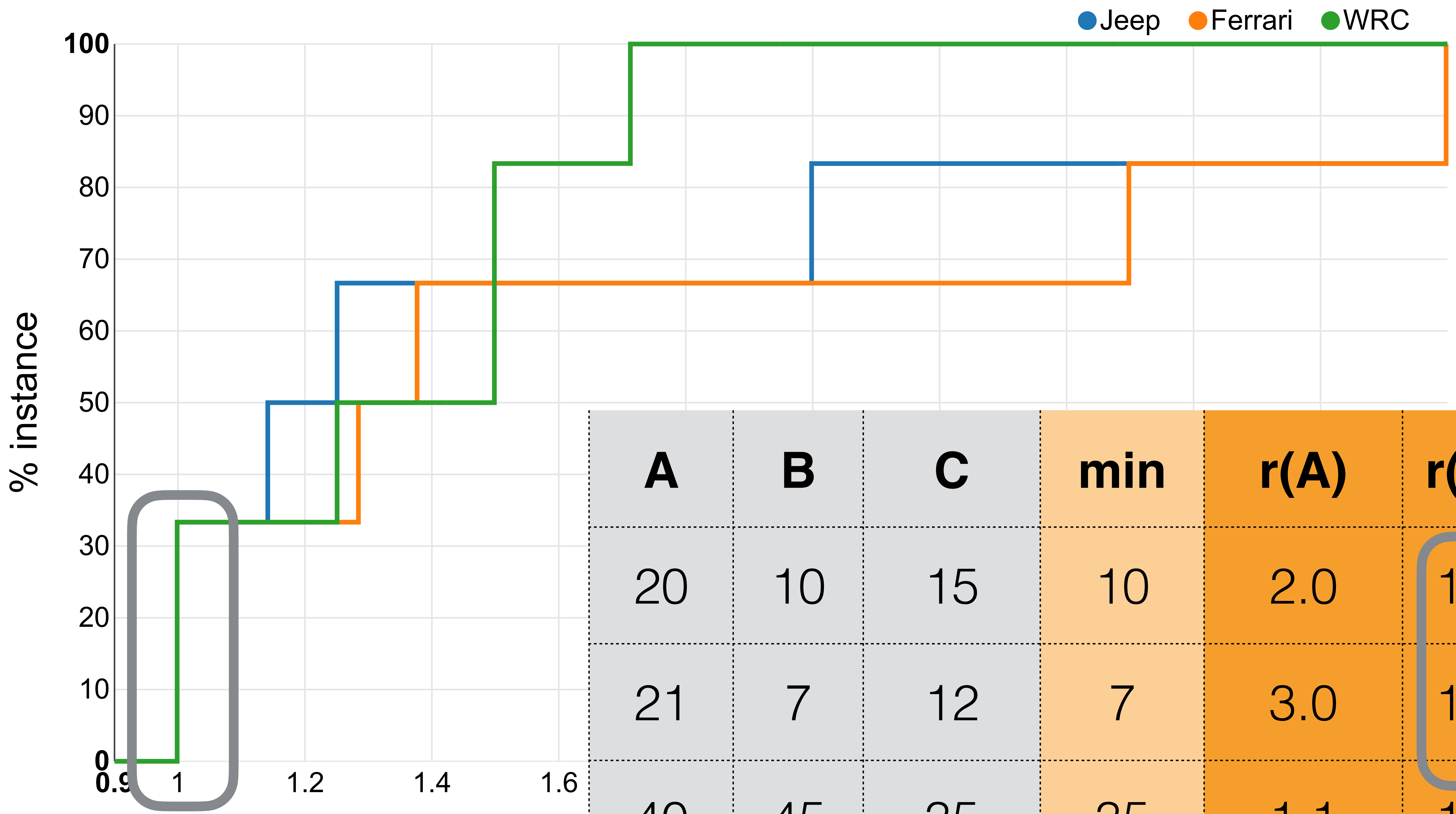
Instance id	A	min	$r_{i,A}$
2	21	7	3.0
1	20	10	2.0
4	50	40	1.3
3	40	35	1.1
5	10	10	1.0
6	20	20	1.0

$$r_{p,s} = \frac{t_{p,s}}{\min\{t_{p,s} : s \in \mathcal{S}\}}$$



$$\rho_s(\tau) = \frac{1}{n_p}\text{size}\{p \in \mathcal{P} : r_{p,s} \leq \tau\}$$

Performances Profiles Interpretation

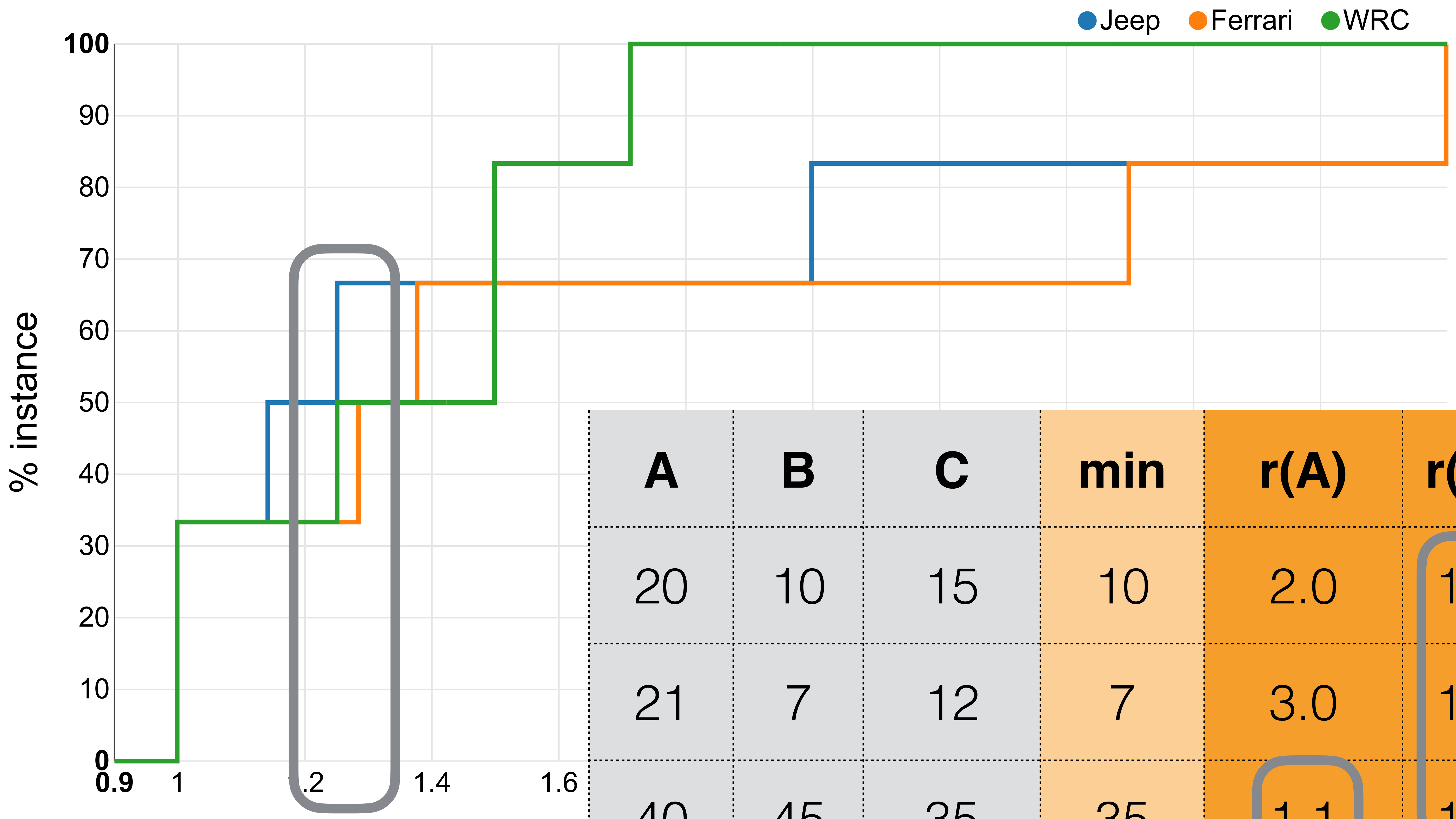


A	B	C	min	r(A)	r(B)	r(C)
20	10	15	10	2.0	1.0	1.5
21	7	12	7	3.0	1.0	1.7
40	45	35	35	1.1	1.3	1.0
50	55	40	40	1.3	1.4	1.0
10	30	15	10	1.0	3.0	1.5
20	50	25	20	1.0	2.5	1.3

r = 1

A,B,C are the best on 33% of the instances

Performances Profiles Interpretation

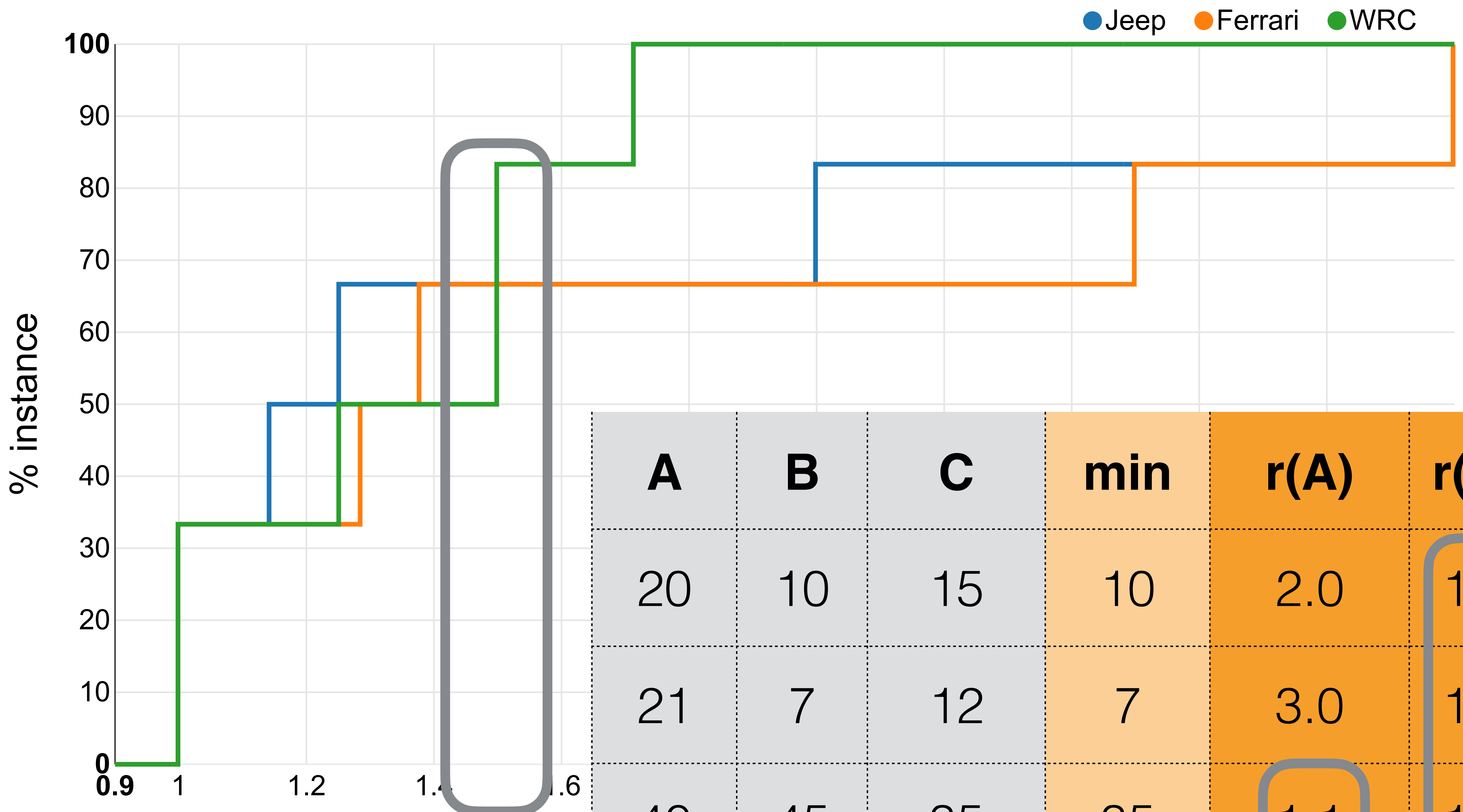


r = 1,3

A is at most 1.3x slower than the virtual best on 66% of the instances

A	B	C	min	r(A)	r(B)	r(C)
20	10	15	10	2.0	1.0	1.5
21	7	12	7	3.0	1.0	1.7
40	45	35	35	1.1	1.3	1.0
50	55	40	40	1.3	1.4	1.0
10	30	15	10	1.0	3.0	1.5
20	50	25	20	1.0	2.5	1.3

Performances Profiles Interpretation



r = 1,5

C is at most 1.5x slower than the virtual best on 83% of the instances

A	B	C	min	r(A)	r(B)	r(C)
20	10	15	10	2.0	1.0	1.5
21	7	12	7	3.0	1.0	1.7
40	45	35	35	1.1	1.3	1.0
50	55	40	40	1.3	1.4	1.0
10	30	15	10	1.0	3.0	1.5
20	50	25	20	1.0	2.5	1.3

Performance Profile Interpretation: Real Example

LOQO has the highest probability to have the smaller run time

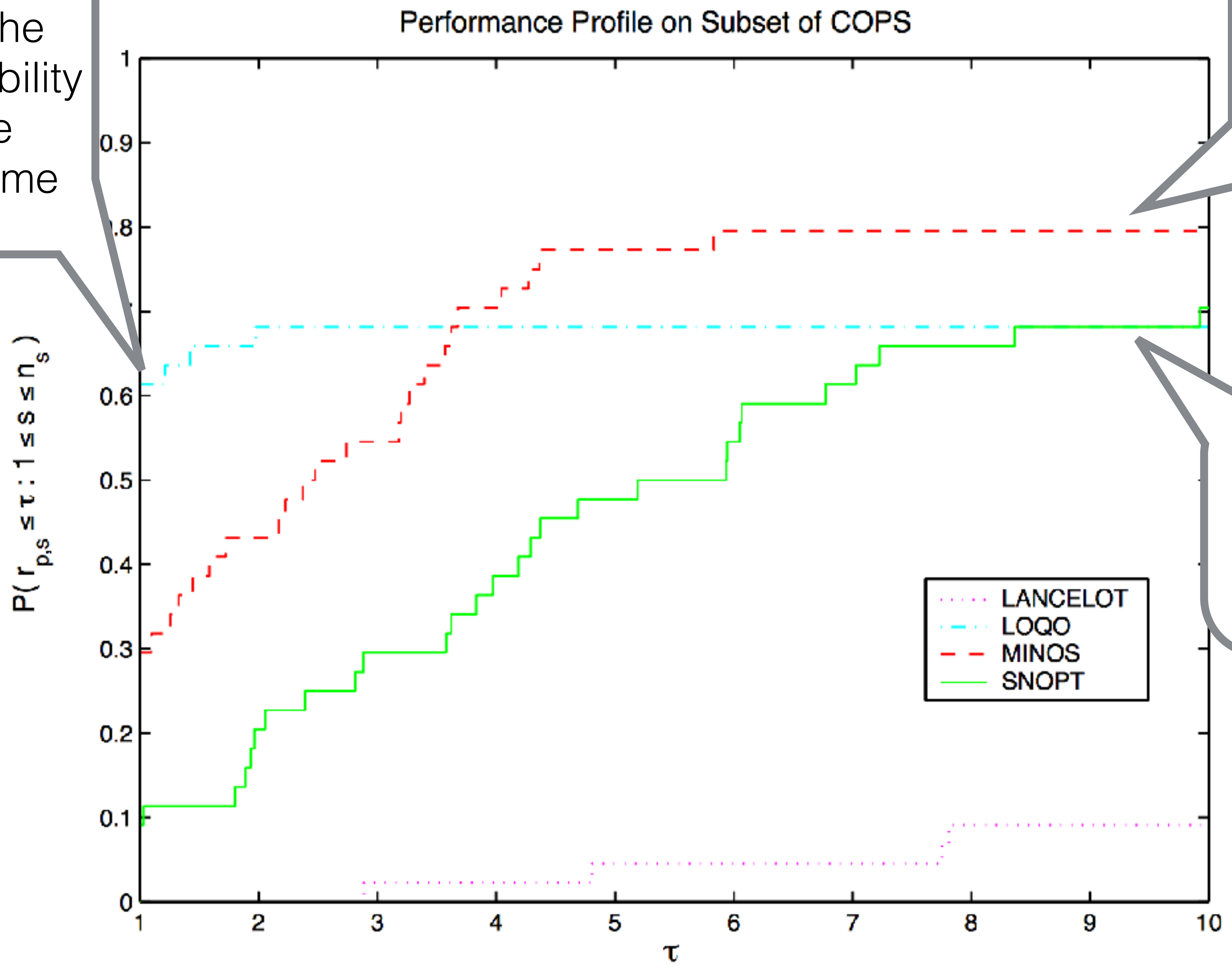


Fig. 4.1. Performance profile on [0, 10]

LANCELOT is more robust, it can solve more instances than any other solvers if you allow it to be within a factor >= 4 of the best run time.

SNOPT is still raising, would be interesting to see what happens further => log2 scale

Linear Scale

Dolan, E. D., & Moré, J. J. (2002). Benchmarking optimization software with performance profiles.

Performance Profile Interpretation: Real Example

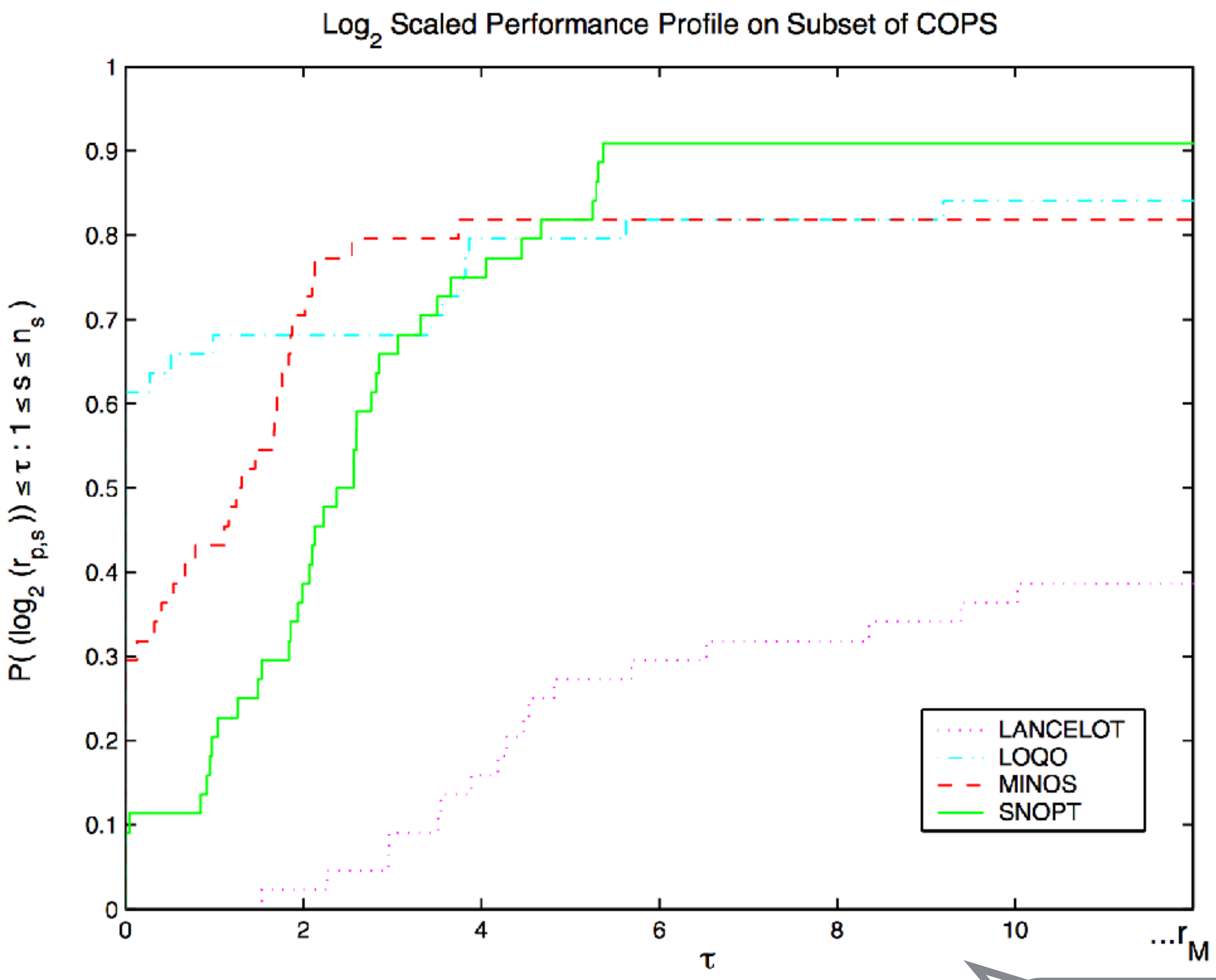


Fig. 4.3. Performance profile in a $\log_2$ scale

now it is clear that if we allow larger computational time, SNOPTis more robust in the sense that it can solve more instances

r_M is our timeout or a measure larger than our timeout

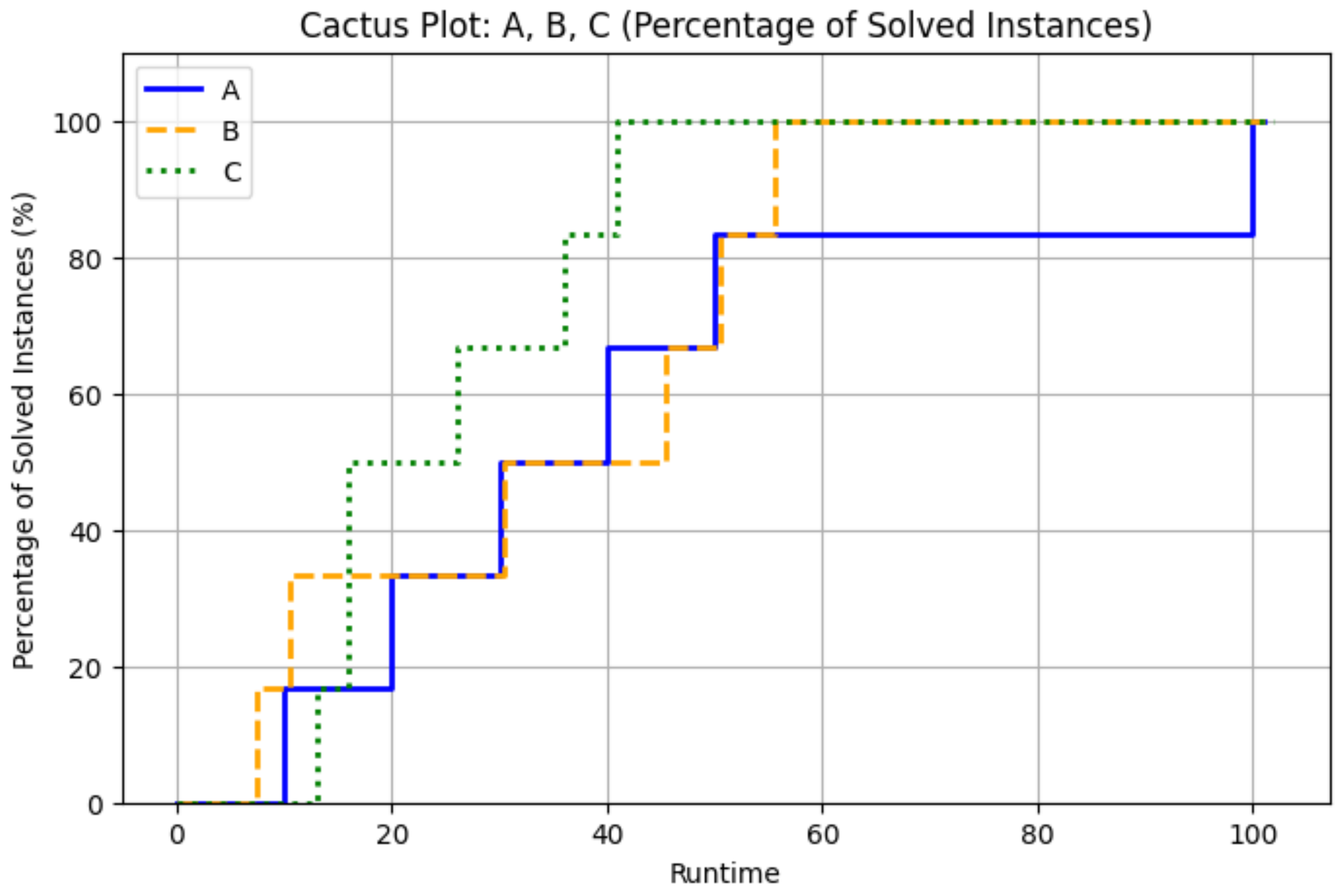
Log Scale

Dolan, E. D., & Moré, J. J. (2002). Benchmarking optimization software with performance profiles.

Cactus Plot vs Performance Profiles

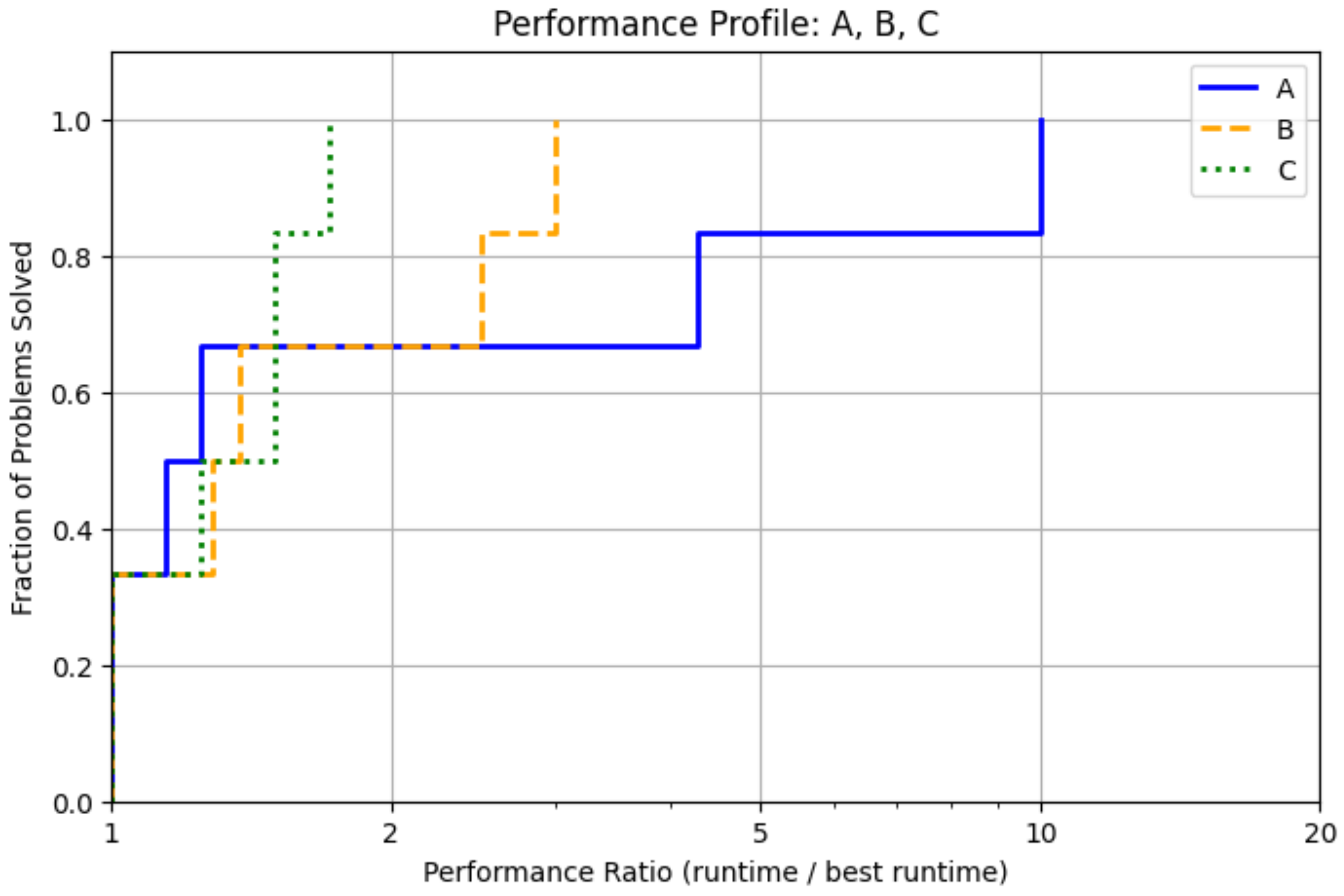
Cactus Plot

$$\kappa_s(\tau) = \frac{1}{n_p}\text{size}\{p \in \mathscr{P} : t_{p,s} \leq \tau\}$$



Performance Profile

$$\rho_s(\tau) = \frac{1}{n_p}\text{size}\{p \in \mathscr{P} : r_{p,s} \leq \tau\}$$



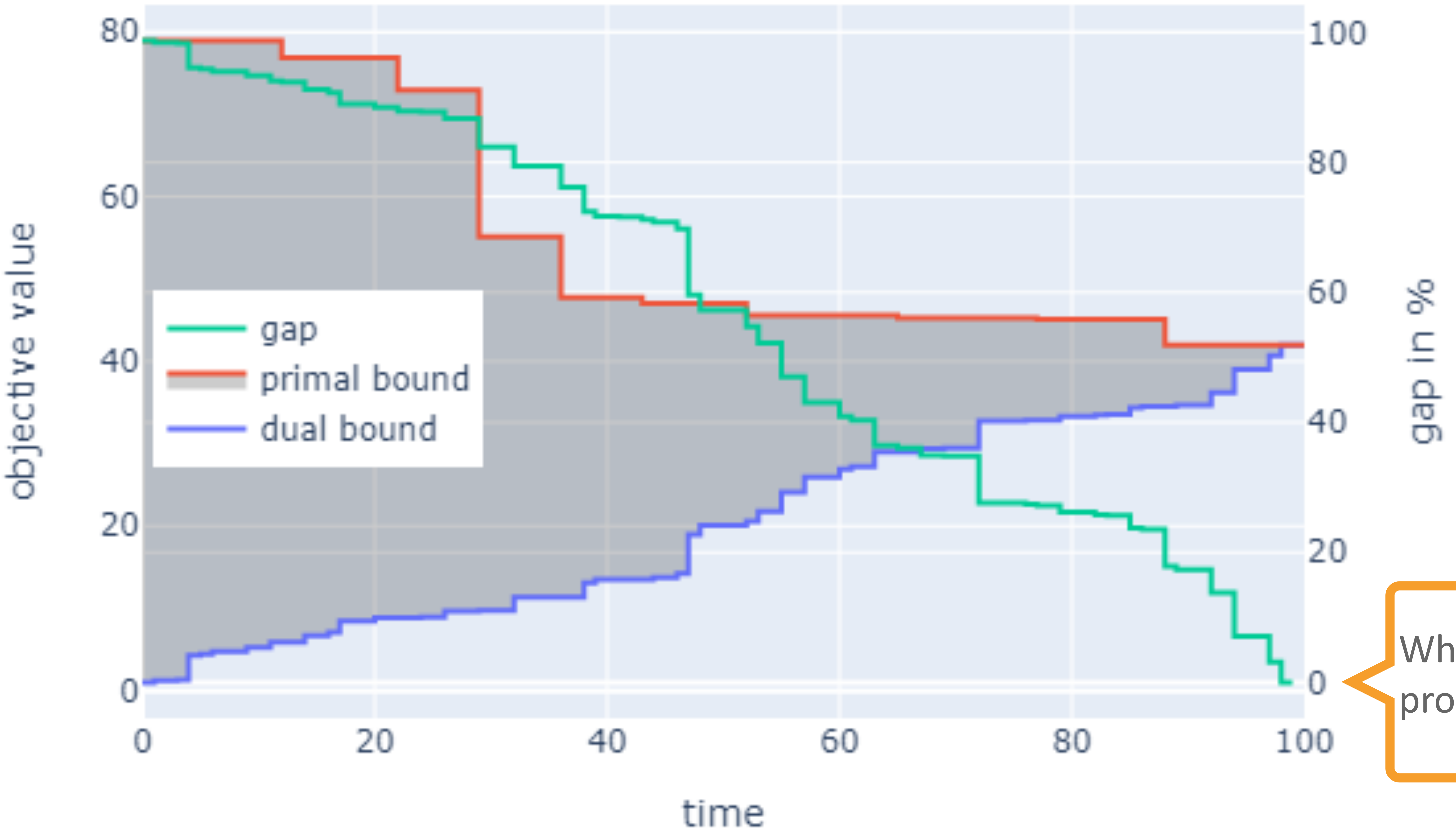
Optimality GAP (Minimization Problem)

- z^* is the optimal objective value of the problem.
- $\bar{z}$ is the best solution found so far

$$\gamma = \frac{|\bar{z} - z^*|}{|\bar{z}|}$$

- Note that if the optimal objective value is not known, a lower bound $\underline{z}$ can be used instead of z^* to compute the optimality gap.
- This lower bound can be provided by the solver itself (best lower bound of open-nodes). MIP solvers report the gap in %

Optimality Gap Decrease: Real Example (Gurobi)



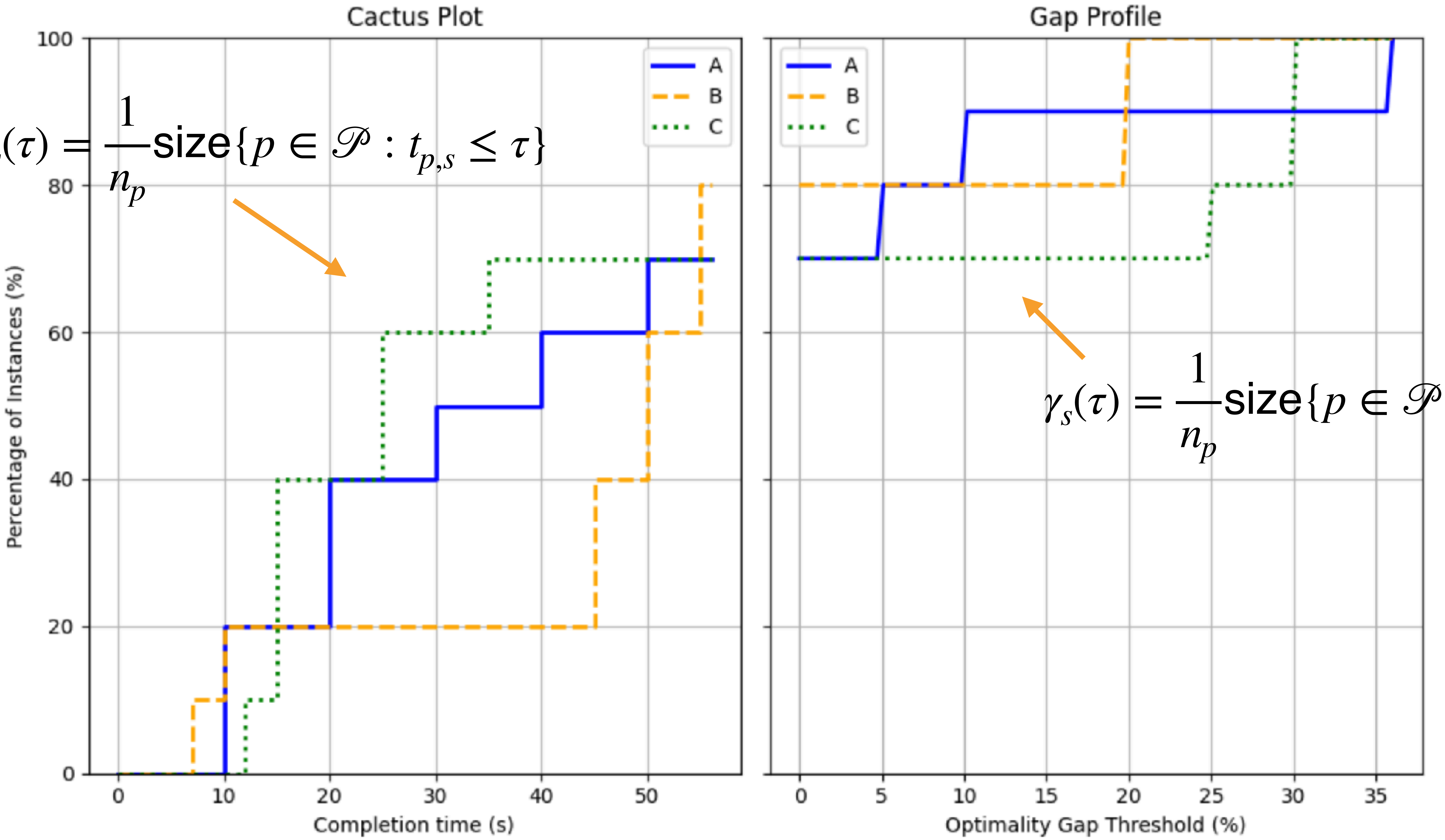
When optimality is proven, the gap is 0

[Https://Support.Gurobi.Com/Hc/En-Us/Articles/8265539575953-What-Is-the-MIPGap](https://support.gurobi.com/hc/en-us/articles/8265539575953-What-Is-the-MIPGap)

Optimization: Objective Value

- Time to prove optimality + GAP at the time out

Instance	A		B		C	
	Time	ɣ	Time	ɣ	Time	ɣ
1	TO	5	10	0	15	0
2	30	0	7	0	12	0
3	40	0	45	0	35	0
4	TO	36	55	0	TO	30
5	10	0	TO	20	15	0
6	20	0	50	0	25	0
7	TO	10	45	0	TO	25
8	50	0	55	0	TO	30
9	10	0	TO	20	15	0
10	20	0	50	0	25	0

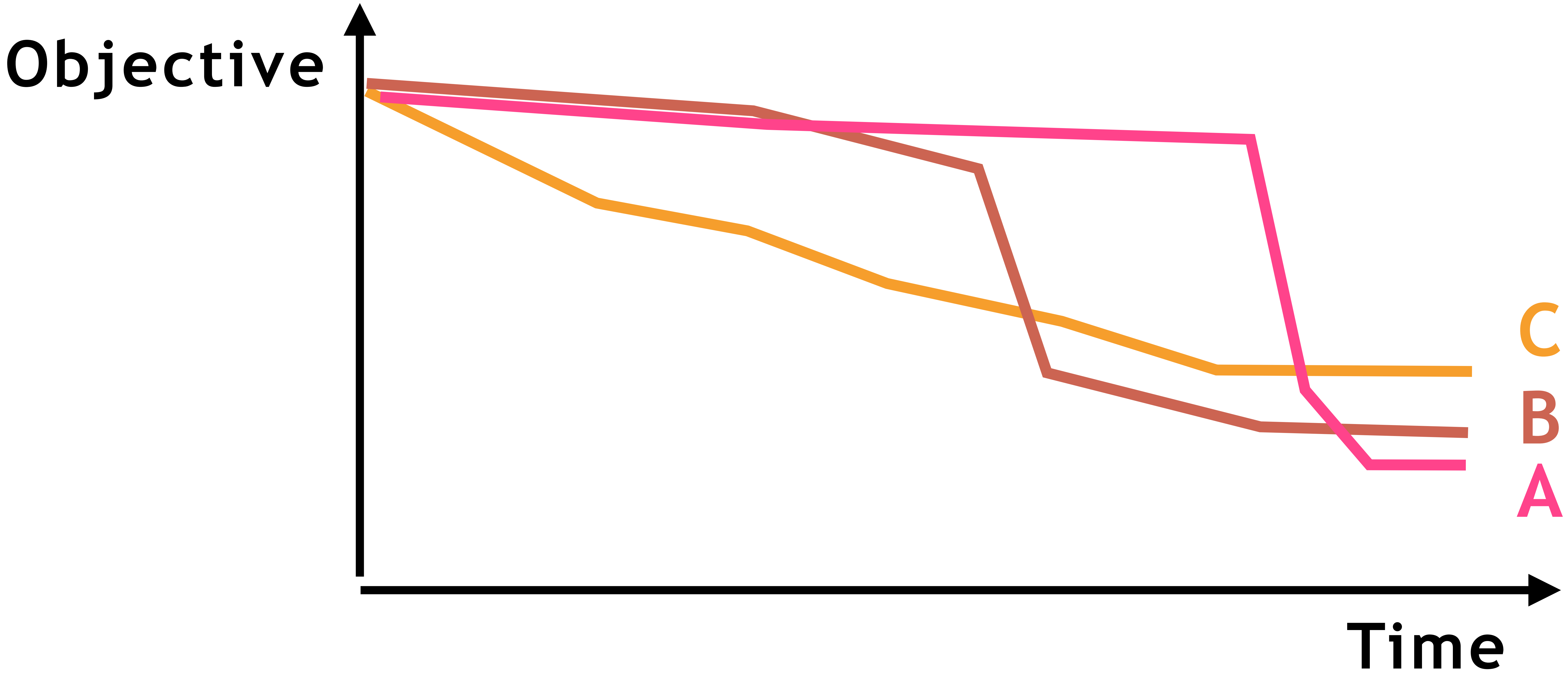


$$\kappa_s(\tau) = \frac{1}{n_p}\text{size}\{p \in \mathcal{P} : t_{p,s} \le \tau\}$$

$$\gamma_s(\tau) = \frac{1}{n_p}\text{size}\{p \in \mathcal{P} : \gamma_{p,s} \le \tau\}$$

Anytime Algorithm

- An algorithm has good anytime behavior when it is able to find high-quality solutions, even when when the search is stopped before completion.



What algorithm do you prefer ?

Berthold, T. (2013). Measuring the impact of primal heuristics. *Operations Research Letters*, *41*(6), 611-614.

Primal Gap Revisited

- Let $\bar{z}$ be a solution for minimization problem, and z^* be an optimal (or best known) solution for that problem.
- We define the primal gap $\gamma \in [0,1]$ of $\bar{z}$ as follows:

$$\gamma(\bar{z}) := \begin{cases} 0, & \textbf{if } |z^*| = |\bar{z}| = 0, \\ 1, & \textbf{if } z^* \cdot \bar{z} < 0, \\ \dfrac{|z^* - \bar{z}|}{\max\{|z^*|, |\bar{z}|\}}, & \textbf{else.} \end{cases}$$

- Now, assume that we have available the objective function values of intermediate incumbent solutions and the points in time when they have been found, for a given a solver, a certain problem instance, and a fixed computational environment.

- Let $t_{\max} \in \mathbb{R}_{\geq 0}$ be a limit on the solution time for the solver.

- Its primal gap function $p : [0, t_{\max}] \mapsto [0,1]$ is defined as follows:

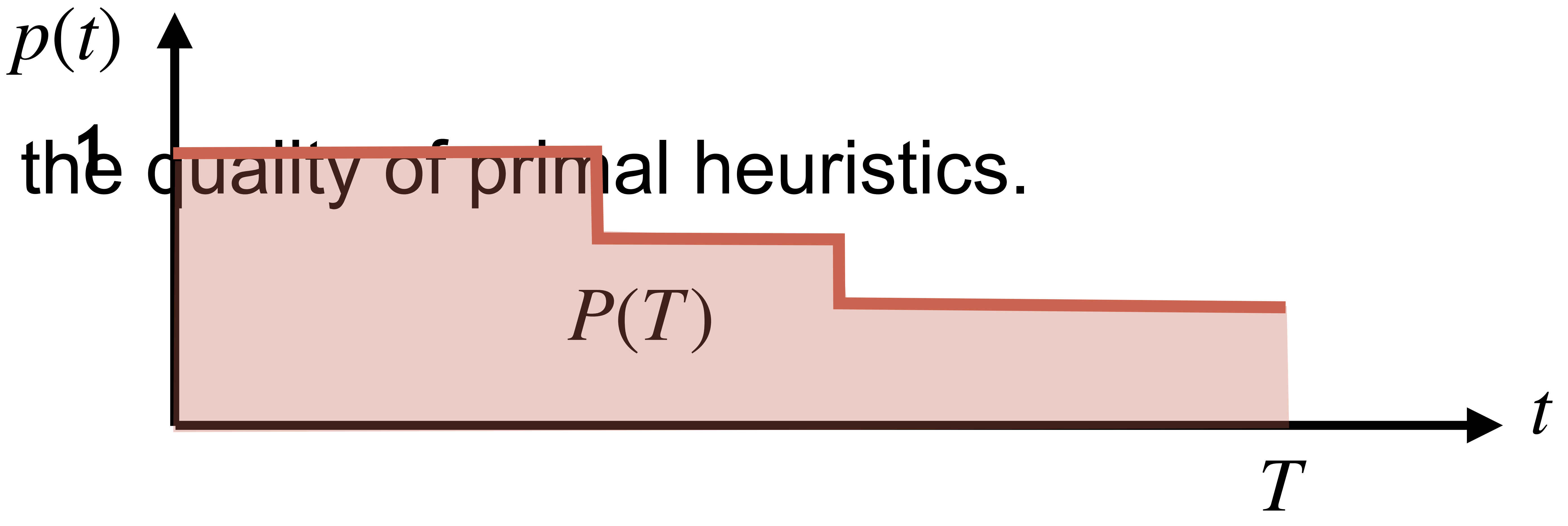
$$p(t) := \begin{cases} 1, & \text{if no incumbent until point } t, \\ \gamma(\bar{z}(t)), & \text{with } \bar{z}(t) \text{ being the incumbent solution} \\ & \text{at point } t, \text{ else.} \end{cases}$$

Primal Integral

- Let $T \in [0, t_{\max}]$, and let $t_i \in [0, T]$ for $i \in \{1, \dots, I-1\}$ be the points in time when a new incumbent solution is found, $t_0 = 0$, $t_I = T$.

- We define the primal integral $P(T)$ of a run as follows:

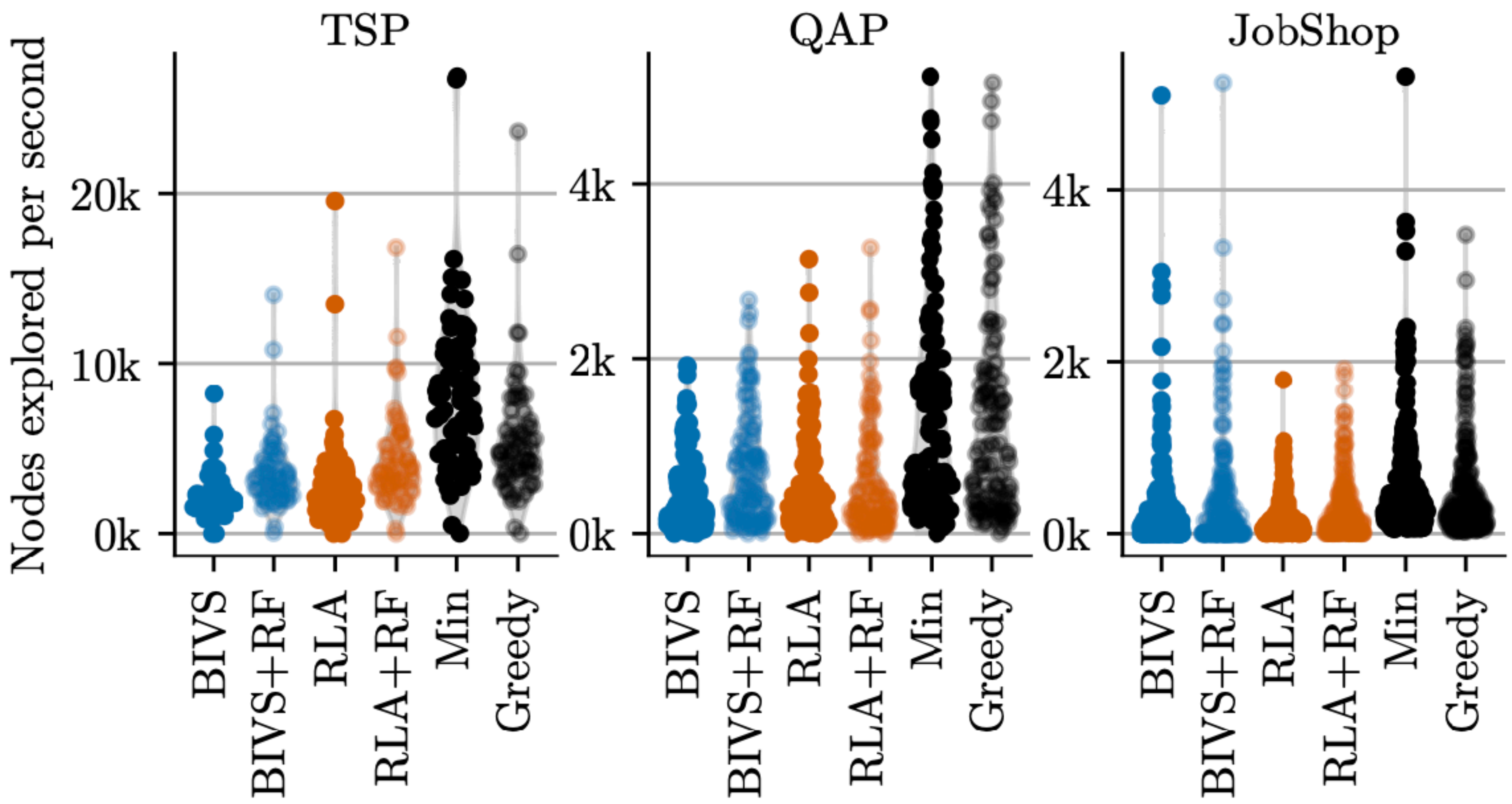
$$P(T) := \sum_{i=1}^{I} p(t_{i-1}) \cdot (t_i - t_{i-1})$$



- $P(t_{\max})$ measures the quality of primal heuristics.

Other interesting plot: Scatter plot Algo A vs Algo B

- SinaPlot (more info than box-plot that gives nothing about distribution)

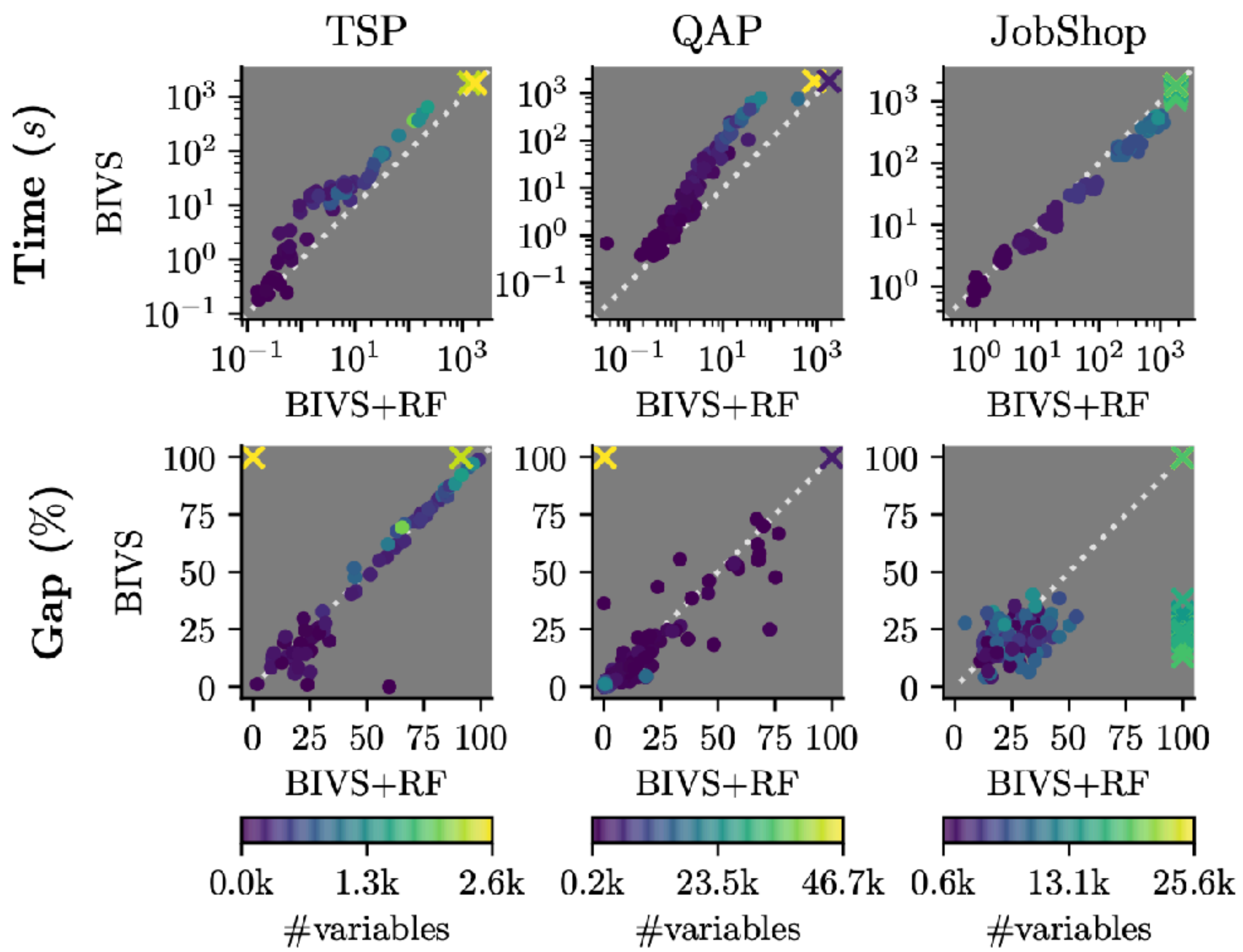


N. Sidiropoulos, S. H. Sohi, T. L. Pedersen, B. T. Porse, O. Winther, N. Rapin, and F. O. Bagger. “SinaPlot: an enhanced chart for simple and truthful representation of single observations over multiple classes”. In: *Journal of Computational and Graphical Statistics* 27.3 (2018), pp. 673–676.

A. Delecluse 2025, *Sequence Variables and Search Heuristics for Vehicle Routing Problems in CP*, PhD Thesis

Other interesting plot: Scatter plot Algo A vs Algo B

To compare time, backtracks, gaps, etc



A. Delecluse 2025, *Sequence Variables and Search Heuristics for Vehicle Routing Problems in CP*, PhD Thesis

⚠️ Arithmetic Mean of Normalized Results = ☠️

- Assume that for each instance, we have measured the time required to reach optimality for algorithms R,M,Z

- Normalize the results wrt first algorithm (R) and then look at the arithmetic mean of the normalized results to compare.

Benchmark	algorithm		
	R	M	Z
E	417 (1.00)	244 (0.59)	134 (0.32)
F	83 (1.00)	70 (0.84)	70 (0.85)
H	66 (1.00)	153 (2.32)	135 (2.05)
I	39,449 (1.00)	33,527 (0.85)	66,000 (1.67)
K	772 (1.00)	368 (0.48)	369 (0.45)
Arithmetic mean	(1.00)	(1.01)	(1.07)

R is the winner

M is 1% slower than R ?

Arithmetic mean, cont

- Now normalize wrt to algorithm M
- Same data, (… oblem?
- The problem … **lts gives meaning les**

algorithm

Benchmark	R	M	Z
E	417 (1.71)	244 (1.00)	134 (0.55)
F	83 (1.19)	70 (1.00)	70 (1.00)
H	66 (0.43)	153 (1.00)	135 (0.88)
I	39,449 (1.18)	33,527 (1.00)	66,000 (1.97)
K	772 (2.10)	368 (1.00)	369 (1.00)
Arithmetic mean	(1.32)	(1.00)	(1.08)

R is 32% slower than R ?

M is the winner

Solution: Geometric Mean

- With geometric mean, whatever column you choose for normalization, you come to the same winner
- Geometric Mean is the only correct way to aggregate normalized results.
- *HOW NOT TO LIE WITH STATISTICS: THE CORRECT WAY TO SUMMARIZE BENCHMARK RESULTS* (Fleming & Wallace, 1986)

$$\text{geometric mean} = \left(\prod_{n=1}^{k} x_n\right)^{\frac{1}{k}}$$

Benchmark	algorithm		
	R	M	Z
E	417 (1.00)	244 (0.59)	134 (0.32)
F	83 (1.00)	70 (0.84)	70 (0.85)
H	66 (1.00)	153 (2.32)	135 (2.05)
I	39,449 (1.00)	33,527 (0.85)	66,000 (1.67)
K	772 (1.00)	368 (0.48)	369 (0.45)
Geometric mean	(1.00)	(0.86)	(0.84)

Benchmark	algorithm		
	R	M	Z
E	417 (1.71)	244 (1.00)	134 (0.55)
F	83 (1.19)	70 (1.00)	70 (1.00)
H	66 (0.43)	153 (1.00)	135 (0.88)
I	39,449 (1.18)	33,527 (1.00)	66,000 (1.97)
K	772 (2.10)	368 (1.00)	369 (1.00)
Geometric mean	(1.17)	(1.00)	(0.99)