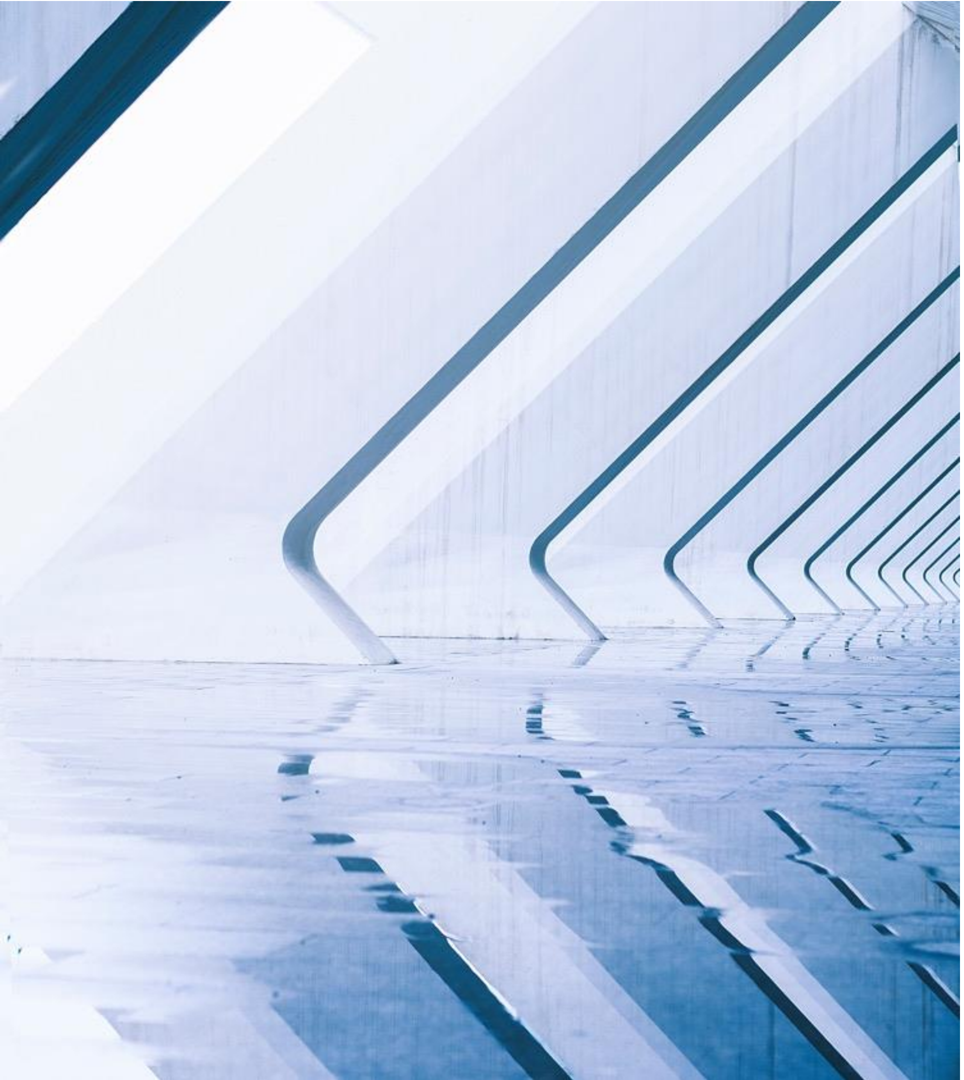


FICO

How to conduct computational experiments

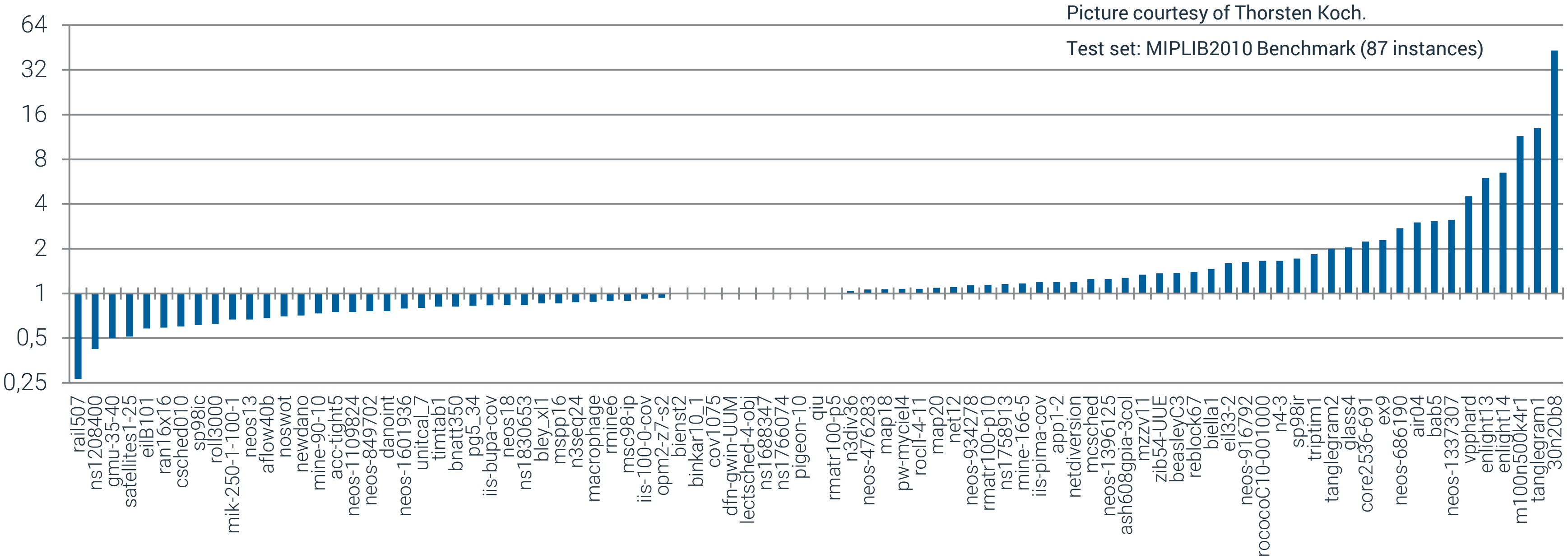
Who is the best?

Timo Berthold

Agenda

- Performance variability
- Statistical tests
- Best practices

What does X% faster mean?



Single solver version-to-version improvement: appx. 30%. HOWEVER:

- ~40% of the instances got slower (~50% got faster)
- Performance drop by up to a factor of 4 (versus improvements of up to factor 45)

Performance Variability

Performance variability

- Performance variability is a change in performance by seemingly performance-neutral changes, like changing the
  - Input format
    - Essentially, this leads to a permutation of the problem
  - Operating system (Does NOT occur with Xpress)
  - Different number of threads (can be overcome by control setting in Xpress)
- This occurs even though the underlying software is deterministic
- Reasons are imperfect tie-breaking, numerical round-off differences on different platforms…
- Related: Adding redundant information (E.g., a constraint $\sum_{i=1}^{n} x_i \leq n,\, x_i \in \{0,1\}$)

Impact of performance variability

- Performance variability
  - Can indicate improvements where there are none
  - Can shadow improvements
  - Can lead to wrong conclusions (“on some instances our method was bad, but on some, it was good”)
- Running Xpress with ten different random permutations:
  - 118/240 instances have a variability of more than a factor of 2
  - For 30, it is more than a factor of 10
  - Most extreme case: 0.02 seconds versus timeout
  - For large test sets, the effect averages out
  - Still 4% difference between „best“ and „worst“ permutation on MIPLIB2017

Exploiting Performance variability

- Performance variability can be utilized to
  - Distinguish between structural improvements and random noise
    - The more permutations/seeds you run, the clearer the picture for each single instance
  - Mimic performance on unknown instances from same application
    - Very, VERY often, customers give you one single example
- Performance variability gives you a poor man's parallelization:
  - Fire off X permutations of the problem, stop when the first one solves
  - Doesn't scale very well ;-)

An improved way to utilize performance variability

- Performance variability for benchmarking is typically induced by
  - Permuting the matrix: High variability 😊 but structural performance “loss” 😒
    - Also: more cache misses
  - Random seeds: Lower variability 😒 but preserves average performance 😊
- New method: Cyclic shifts
  - Light-weight permutation that shifts the rows and columns
  - Only two „breakpoints“
  - High variability 😊 and preserves average performance 😊
  - Can be combined with random seeds
- As a solver developer, after each release:
  - We change the benchmarking permutation seeds
  - The offset to the random seed

Statistical Tests

Idea of statistical tests

- To test a hypothesis, we compare it to the null-hypothesis: “There is no such relationship”
  - When the null-hypothesis can be rejected with high probability, the result is deemed significant
  - Essentially, this expresses the confidence in the result. How safe is it to draw conclusions from the test results?
  - “How likely is it that a random draw would have delivered the same (or an even more extreme) result?”
  - Typically, p-values less than 5% are considered significant.
    - For a normal distribution, 95% are within mean +/- two standard deviations
- There is a huge variety of statistical tests, require different assumptions, e.g., concerning the distribution. In our case typically: distribution-free (non-parametric)
- Distinguish between nominal data (yes/no) and rational data

McNemar test (1947)

- Applied to 2x2 contigency table of paired nominal data
  - E.g., does the solver find a solution with/without a certain setting (yes/no, yes/no)
- Considers the cases where both differ (the counter diagonal of the table)
  - Compute $\chi^2 = \frac{(b-c)^2}{b+c}$
  - For random drawed b, c, this would follow a chi-squared-distribution.
    - Lookup p-value
- Well-suited for categorical data (found a solution yes/no, solved to optimality yes/no)

Wilcoxon signed rank test (1945)

- Non-parametric difference test for paired rational data
  - Sorts observations by absolute value of the logarithm of their ratio
- Assigns ranks from 1 to n
- Split into positive and negative group, compute rank sums
  - The more different those sums, the more likely that the setting with the larger sum outperforms the one with the lower sum
- Wilcoxon statistic: $z = \dfrac{\min(W_-, W_+) - \frac{N(N+1)}{4}}{\sqrt{\frac{N(N+1)(2N+1)}{24}}}$
  - Would follow a normal distribution for randomly drawn data
- Well-suited for numerical data (runtime, number of nodes, PDI)

Best practices

Assorted list of advice

- Choose a suitable test set
  - Are there standard test sets in your field?
  - Use large test sets
  - Use diverse test sets (within your target domain)
  - Use „real“ instances if possible, not randomly generated

- Be careful, what to exclude and how to split test sets
  - NEVER EVER do „easy“ vs. „hard“ only w.r.t. one „baseline“ setting
  - Some measures might only make sense on „all optimal“ or „all timeout“
    - Report number of solved instances
  - Explain why certain instances had to be excluded and how many were affected

Assorted list of advice

- Choose a suitable test set
  - Are there standard test sets in your field?
  - Use large test sets
  - Use diverse test sets (within your target domain)
  - Use „real“ instances if possible, not randomly generated

- Be careful, what to exclude and how to split test sets
  - NEVER EVER do „easy“ vs. „hard“ only w.r.t. one „baseline“ setting
  - Some measures might only make sense on „all optimal“ or „all timeout“
    - Report number of solved instances
  - Explain why certain instances had to be excluded and how many were affected

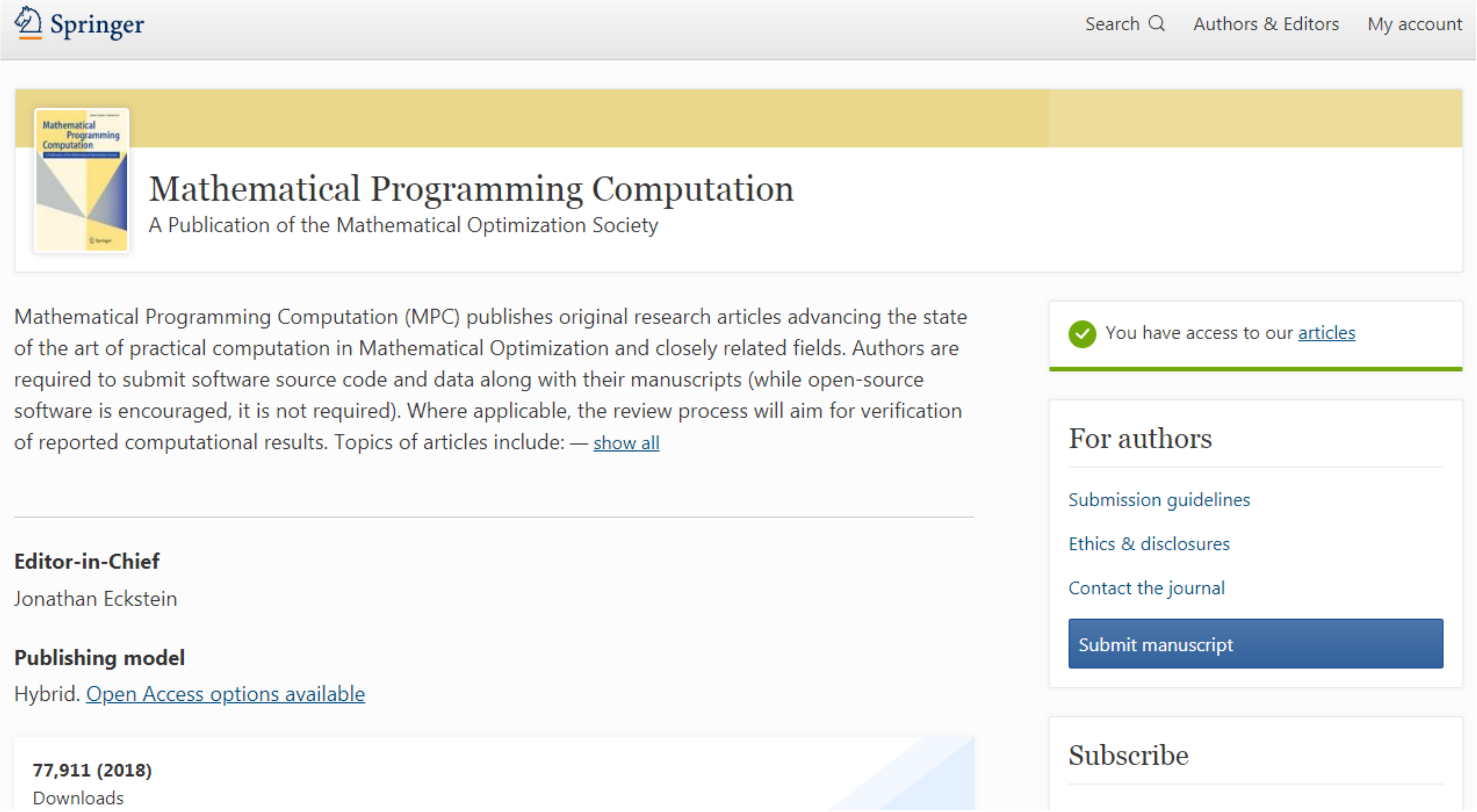
Assorted list of advice

- Report machine specification, software versions and working limits used
- Make fair comparisons
  - What is the state-of-the-art?
    - Not only software, but also models
  - Does the benchmark solver have the same purpose (e.g., heuristics vs exact solvers)

Assorted list of advice

- State the question(s) that you want to answer and the means of doing so
  - @Reviewers: This might not be the same question and the same methodology that you would have chosen....
- Run on otherwise idle machines
  - Run at most one job per CPU (and bind to CPU)
- Never trust your own results
  - Investigate outliers, negative and positive ones
  - Use methods like permutations, statistical tests etc to double-check your results
- Use diverse measures
- Also report negative results

Publish your computational studies!

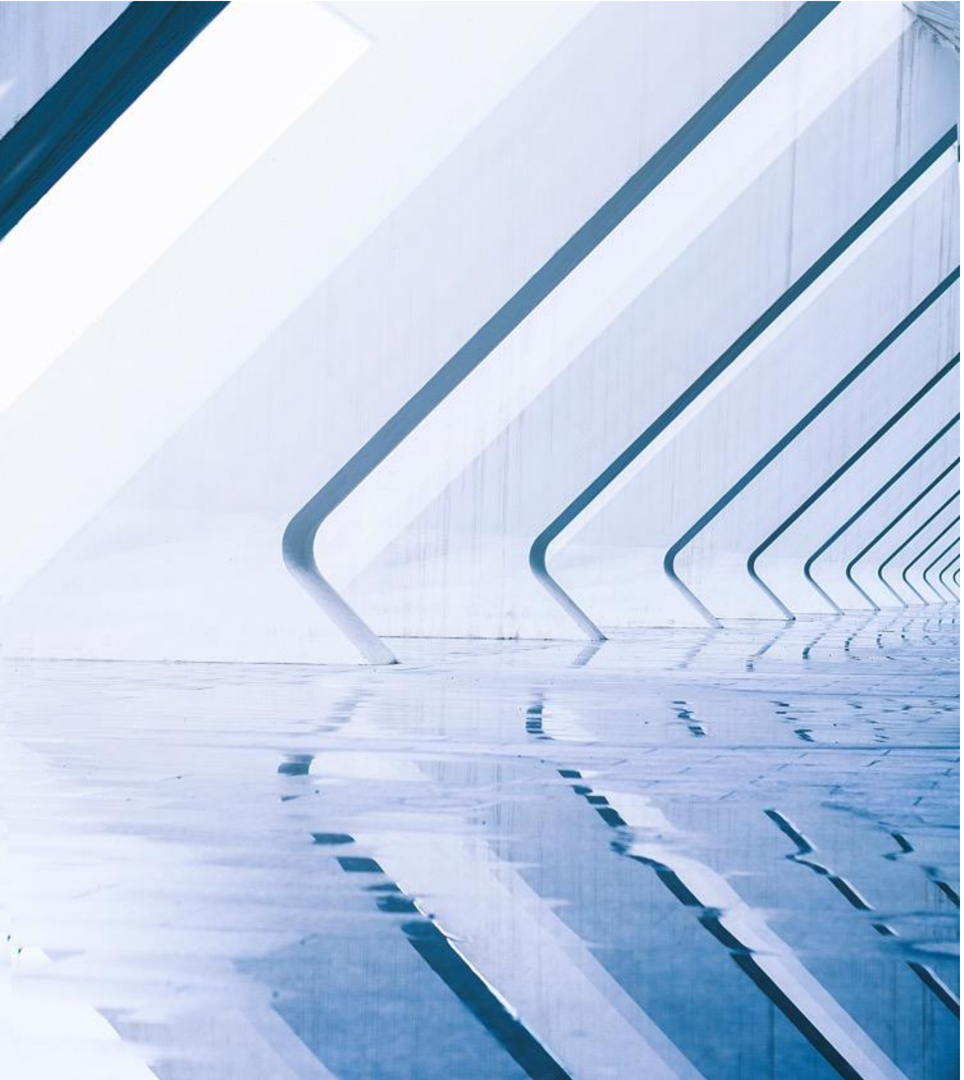


Quiz time

- Performance variability describes
  a) Large runtime differences caused by seemingly neutral changes
  b) A method to tell whether a result is significant
  c) A performance measure
- What is NOT a best practice for computational experiments?
  a) Use diverse machine setups
  b) Use diverse performance measures
  c) Run on otherwise idle machines
- When comparing runtimes, the significance should be checked by
  a) A McNemar test
  b) A Cyclic shift test
  c) A Wilcoxon signed rank test

Quiz time

- Performance variability describes
  a) Large runtime differences caused by seemingly neutral changes
  b) A method to tell whether a result is significant
  c) A performance measure
- What is NOT a best practice for computational experiments?
  a) Use diverse machine setups
  b) Use diverse performance measures
  c) Run on otherwise idle machines
- When comparing runtimes, the significance should be checked by
  a) A McNemar test
  b) A Cyclic shift test
  c) A Wilcoxon signed rank test



Thank You!

Timo Berthold