Characteristic Subsets of TPTP Benchmarks

Karel Chvalovský[1] Jan Jakubův[1,2]

[1] Czech Technical University in Prague, Prague, Czechia

[2] University of Innsbruck, Innsbruck, Austria

AITP'21, 9$^{\mathrm{th}}$ September 2021
Aussois and online, France

European Research Council
Established by the European Commission

Motivation: Common Problems

- Task: Prover evaluation over a large benchmark problem set
- The prover is often parametric
  - we want to evaluate several configurations (strategies)
- Benchmark problem sets are large
  - TPTP/FOL benchmark has more than 16000 problems
  - evaluation of a single strategy with 300 seconds time limit ...
  - ...takes really long ($\sim$ 56 days!)

Motivation: Common Solutions

- Task: Prover evaluation over a large benchmark problem set
- Parallelization: with 60 cores from 56 days to $\sim$ 23 hours
- Time restriction: evaluate with a shorter time limit
- Size restriction: evaluate only on some problems
  - specific benchmark problems selection
  - random benchmark problems subset

Motivation: Can We Do Better?

In this talk we address the following questions:

- many benchmark problems are similar
- we try to identify classes of similar problems . . .
- . . . and select just one problem from each class . . .
- and create a benchmark characteristic subset

The Rest of the Talk

Motivation: ATP Prover Evaluation over Large Benchmark

Benchmark Characteristic Subsets by Clustering

Evaluation Metrics: Strategy Selection and Grid Search

Experimental Evaluation

Benchmark Characteristic Subsets: Overview

- Idea: Lets make use of problem similarities.
- Represent each problem by a feature vector and ...
  ... employ machine learning clustering methods to ...
  ... construct clusters of similar problems.
- Take just one problem from each cluster and thusly ...
  ... construct a benchmark characteristic subset.

Problems as Vectors: Performance Features

- To use machine learning methods for clustering . . .
- . . . problems must be represented by numeric feature vectors.
- We experiment with 2 kinds of features:
  - Performance features: runtime statistics
  - ENIGMA features: syntactic features

Problems as Vectors: Performance Features

- Run E Prover strategy with a small resources limit (1000 generated clauses)
- Collect runtime statistics

= 10 counts like: processed clauses, paramodulations, subsumptions, rewriting steps...

- We reserve 10 E strategies to construct problem features.

⇒ We obtain a vector of length 100 representing each problem.

Problems as Vectors: ENIGMA Features

- ENIGMA features represent clauses as numeric vectors:
  - symbol anonymization by arity
  - cut the syntax tree into pieces
  - enumerate and count the pieces
  - feature hashing
- To represent a TPTP problem as a vector:
  - Translate a problem to a set of clauses.
  - Translate clauses to ENIGMA feature vectors.
  - Average the vectors to obtain the problem characteristic vector.

k-means Clustering

- Task: Split the problems into k different classes, such that ...
  ...similar problems end up in the same class.
- k-means clustering algorithm overview:
  1. Randomly select k vectors called centroids.
  2. Compute distances between problem vectors and centroids.
  3. Form clusters by assigning each problem to the closest centroid.
  4. Average the vectors in each cluster.
  5. Move centroids to the computed averages.
  6. Repeat from step 2 until the centroids stop moving.

Characteristic Subset Construction

- To construct a characteristic subset of size k ...
  ...we construct k clusters using k-means.
- Take the problem closest to the centroid from each cluster ...
  ...as the cluster representative.

Outline

Motivation: Benchmark Subset Quality

- Suppose we somehow select a benchmark subset.
- We would like to measure how "good" this subset is, ...
- ... that is, how well
  - the performance on the subset correlates with
  - the performance on all problems.
- We use 3 different evaluation metrics:
  - the best strategy selection
  - best cover construction
  - strategy parameters grid search

Metric 1: Best Strategy Selection

- Task: Select the best out of 444 E strategies.
- Measure the quality of a benchmark subset P_{sub} as:
    1. Select the best strategy S on P_{sub}
    2. Compute the performance of S on all problems (approx)
    3. Compare S with the best strategy on all problems (optimal)

$$\mathit{error}(P_{\text{sub}}) = 100 \cdot \left|1 - \frac{\mathit{approx}}{\mathit{optimal}}\right|$$

Metric 2: Best Cover Construction

- Task: Select n out of 444 E strategies ...
  ... maximizing the count of solved problems.
- Greedy cover construction:
  1. Evaluate all strategies on all problems.
  2. First select the strategy that solves most problems.
  3. Remove the problems solved by this strategy.
  4. Iterate.
- Exact cover construction: NP-hard.

Metric 3: Strategy Parameter Grid Search

- E strategies has many parameters.
- Task: Select the best values for selected parameters.
- Example (part of the best strategy on TPTP):

```
--destructive-er --destructive-er-aggressive --forward-context-sr
--forward-demod-level=1 --simul-paramod --sos-uses-input-types
--strong-destructive-er --term-ordering=KBO6 [..]
-H'(1*ConjectureRelativeSymbolWeight(SimulateSOS,0.5,100,100,100,..),
    4*ConjectureRelativeSymbolWeight(ConstPrio,0.1,100,100,100,..),
    1*FIFOWeight(PreferProcessed),
    1*ConjectureRelativeSymbolWeight(PreferNonGoals,0.5,100,100,..),
    4*Refinedweight(SimulateSOS,3,2,2,1.5,2))'
```

Metric 3: Strategy Parameter Grid Search

- Task: Try to find better values for 4 selected parameters.

  1*ConjectureRelativeSymbolWeight(SimulateSOS,0.5,100,100,100,..),

  4*ConjectureRelativeSymbolWeight(ConstPrio,0.1,100,100,100,..),

  1*FIFOWeight(PreferProcessed),

  1*ConjectureRelativeSymbolWeight(PreferNonGoals,0.5,100,100,..),

  4*Refinedweight(SimulateSOS,3,2,2,1.5,2)

Metric 3: Strategy Parameter Grid Search

- Task: Try to find better values for 4 selected parameters.

1*ConjectureRelativeSymbolWeight(SimulateSOS,0.5,100,100,100,..),

4*ConjectureRelativeSymbolWeight(ConstPrio,0.1,100,100,100,..),

1*FIFOWeight(PreferProcessed),

1*ConjectureRelativeSymbolWeight(PreferNonGoals,0.5,100,100,..),

4*Refinedweight(SimulateSOS,3,2,2,1.5,2)

Metric 3: Strategy Parameter Grid Search

- Task: Try to find better values for 4 selected parameters.

  a*ConjectureRelativeSymbolWeight(SimulateSOS,0.5,100,100,100,..),

  b*ConjectureRelativeSymbolWeight(ConstPrio,0.1,100,100,100,..),

  1*FIFOWeight(PreferProcessed),

  c*ConjectureRelativeSymbolWeight(PreferNonGoals,0.5,100,100,..),

  d*Refinedweight(SimulateSOS,3,2,2,1.5,2)

- $a, b, c, d \in \{1, 2, 3, 4, 5, 10, 15\}$

Outline

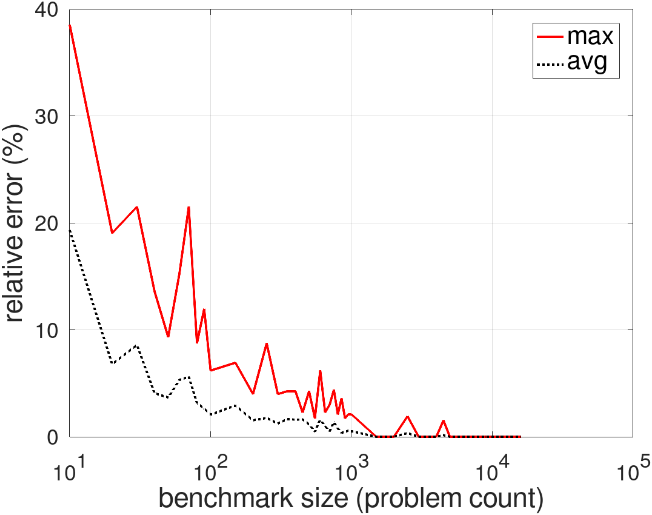
Random Subsets: Baseline

- Construct random benchmark subsets of different sizes:
  $10, 20, \ldots 100, 150, \ldots, 1000, 1500, \ldots, 16000$
- Compute the error for metrics (1, 2, 3) for each subset.
- Do this 10 times with different random selection and compute
  - the worst case error
  - the average error

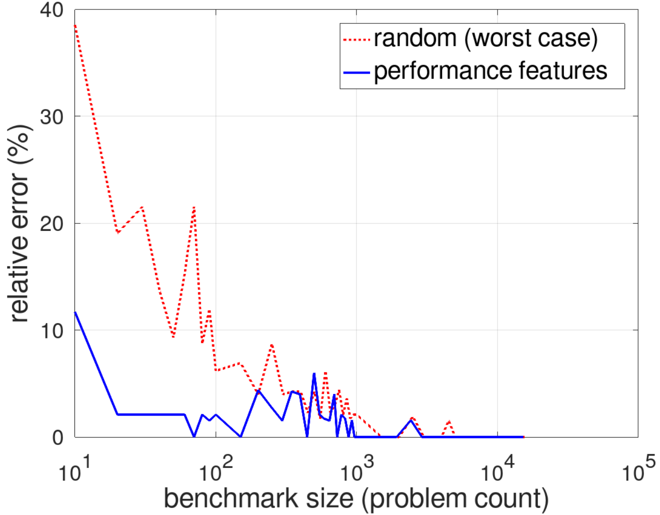
Characteristic Subsets: Data

- Construct characteristic subsets of the same sizes (as random)
- Metric 1,2: All strategies evaluated on all TPTP problems (5s)
- Metric 3: All combinations (7^4) of parameters (1s)
- More than a year of a single CPU time.
- Metric 2: Greedy covers of various sizes (2,3,5,. . . ,300)
  . . . taking the average error

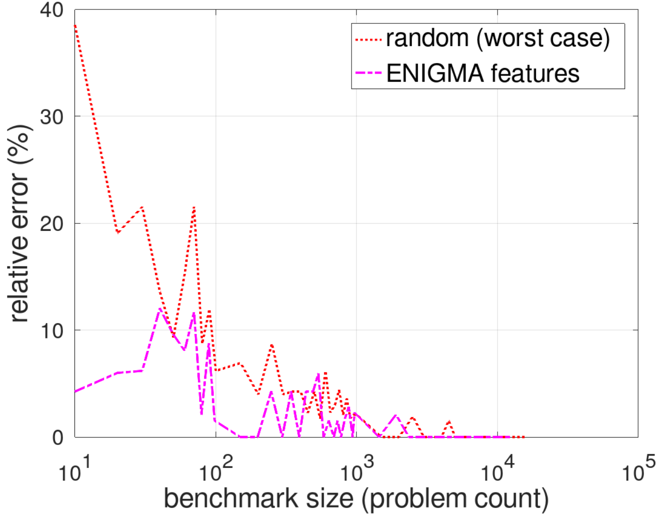
Metric 1 (Best Strategy): Random Subsets



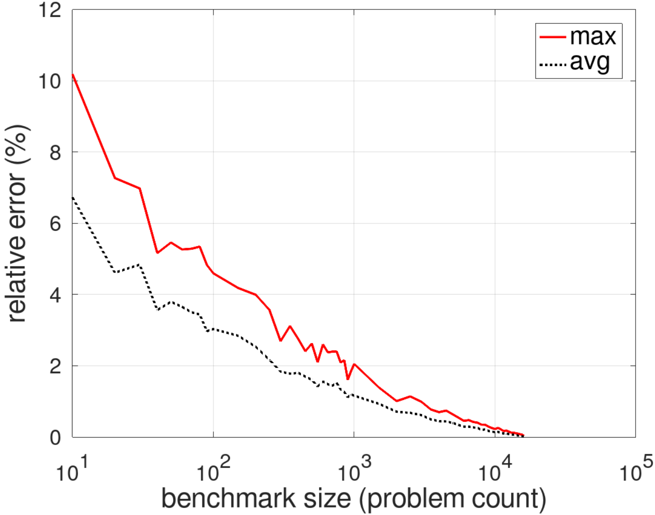
Metric 1 (Best Strategy): k-means Clustering



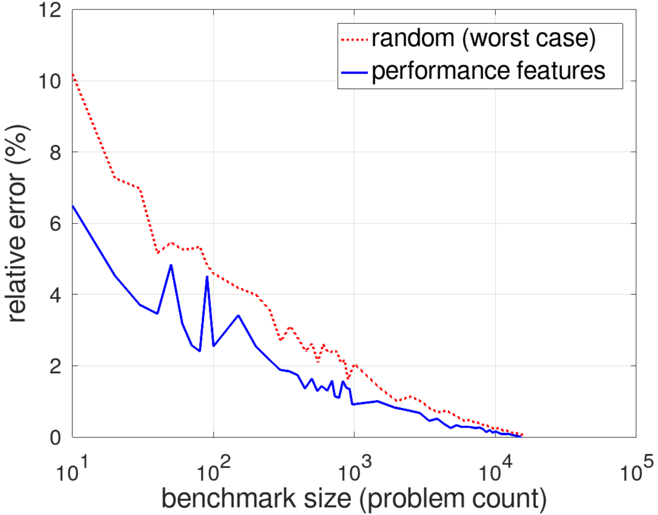
Metric 1 (Best Strategy): k-means Clustering



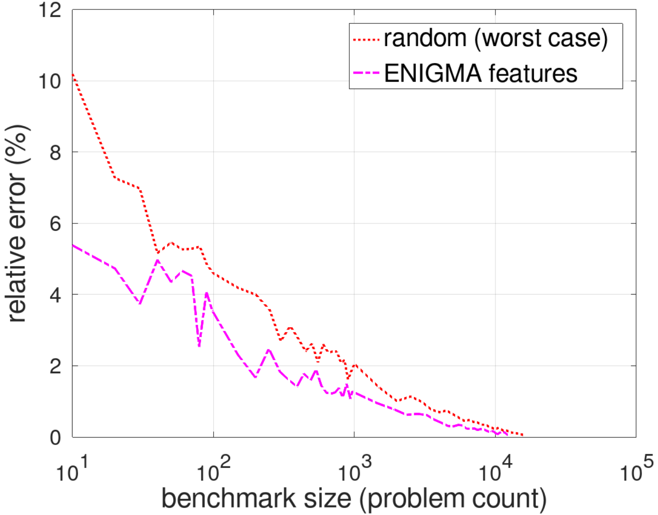
Metric 2 (Greedy Cover): Random Subsets



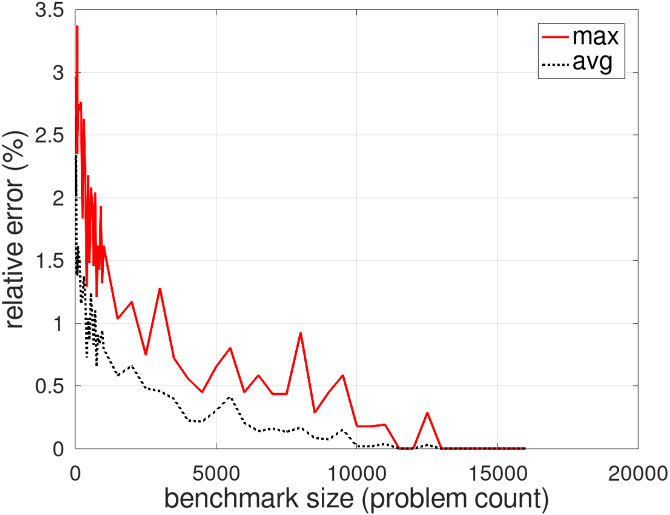
Metric 2 (Greedy Cover): k-means Clustering



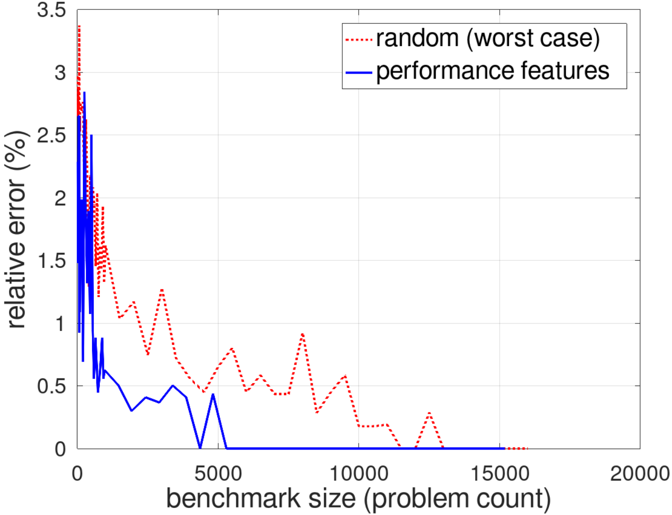
Metric 2 (Greedy Cover): k-means Clustering



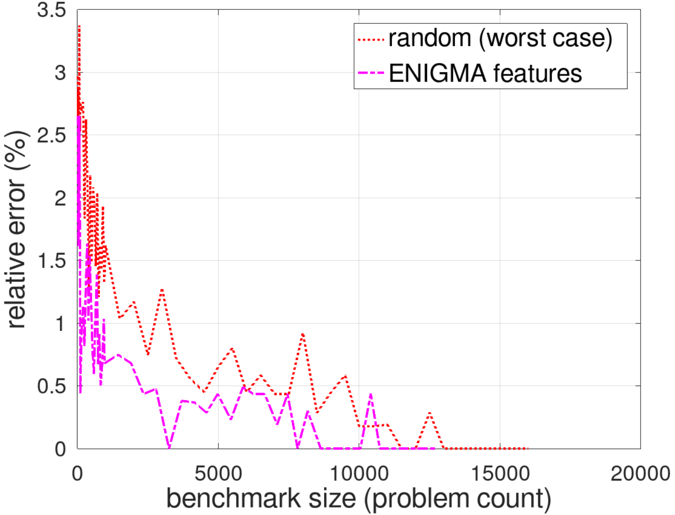
Metric 3 (Grid Search): Random Subsets



Metric 3 (Grid Search): k-means Clustering



Metric 3 (Grid Search): k-means Clustering



Benchmark Characteristic Subsets: Conclusions

- It is possible to construct characteristic subsets...
  ...better than random subset selection.
- k-means gives smaller error then random subset selection.
- Performance features performs better than ENIGMA features.
- The error approaches the average error on random subsets.

$\Rightarrow$ less coincidental construction

Finally. . .

Our computed benchmark characteristic subsets of TPTP

can be downloaded:

`https://github.com/ai4reason/public/blob/master/AITP2021`

Performance Feature Statistics

```
Processed clauses
Generated clauses
Removed by relevancy pruning/SinE
Backward-subsumed
Backward-rewritten
Paramodulations
Factorizations
Equation resolutions
Clause-clause subsumption calls
Termbank termtop insertions
```

Cluster Sizes for k=100 (in % of TPTP size)