Synthetic Benchmarks for Genetic Improvement

Aymeric Blot  Justyna Petke

University College London, UK
UK EPSRC grant EP/P023991/1

GI@ICSE — 3 July 2020
In a Nutshell

**Motivation:**
- Empirical comparisons of GI approaches
- Parameter configuration of GI
- Genetic improvement of GI
- Quick experimentation for GI ideas

**Idea:**
- Premise: GI applied on software is very slow
- Bottleneck: fitness evaluation
- Proposition: synthetic benchmarks
Synthetic Benchmarks

Issues with real-world benchmarks:
- Evaluation is expensive
- Good data is scarce
- Uncertain features

Possible solutions:
- Surrogate modelling
- Artificial instances
- Synthetic benchmarks

Dang et al., GECCO 2017 (AC(AC) using surrogate modelling)
Malitsky et al., LION 2016 (Structure preserving instance generation)
Formalism

Standard GI:

\[
\begin{align*}
&\text{optimise } E[o(s, i), i \in D] \\
&\text{subject to } s \in S
\end{align*}
\]

with:

- \( E \): statistical population parameter (e.g., average)
- \( o \): cost metric (e.g., running time)
- \( D \): input distribution (e.g., test cases, instances)
- \( s \): software variants
- \( S \): search space

Idea: Replacing \( E[o(s, i), i \in (D)] \) by a single instantaneous query
Software Analysis

Search space:
- Around \( n \) deletions
- Around \( n^2 \) replacements
- Around \( n^2 \) insertions

\[ \implies \sum_{i=1}^{k} (n^{2i}) \text{ sequences up to size } k \]
- that’s too big!

Assumption:
- Edits are independent

\[ \implies \text{only around } n^2 \text{ fitness values} \]
- reasonable to model
Synthetic Model

Empirical analysis:
- Sample edits
- Collect data, e.g.:
  - did it compile?
  - did it run?
  - was it correct?
  - how much better/worse?
- Compute underlying distribution

Contribution aggregation:
- Compilation errors propagate
- Runtime errors propagate
- Wrong outputs propagate
- Duplicate edits are ignored
- Fitness ratios are multiplied

E.g.: \([80\%, 100\%, 105\%] \rightarrow 84\%\)
Conclusion

Problem:
- GI(software) is much slower than software
- GI(GI(software)) is much much slower than GI(software)

Idea:
- Replace software with model
- model is free
- GI(model) is cheap
- GI(GI(model)) should be reasonable

Advantages:
- Cheap, reusable benchmarks
- Model as complex as designed
- Possible focus on particular software feature