Genetic Improvement @ ICSE 2021

Personal reflection of a Workshop Participant

Alexander E. I. Brownlee
University of Stirling, UK

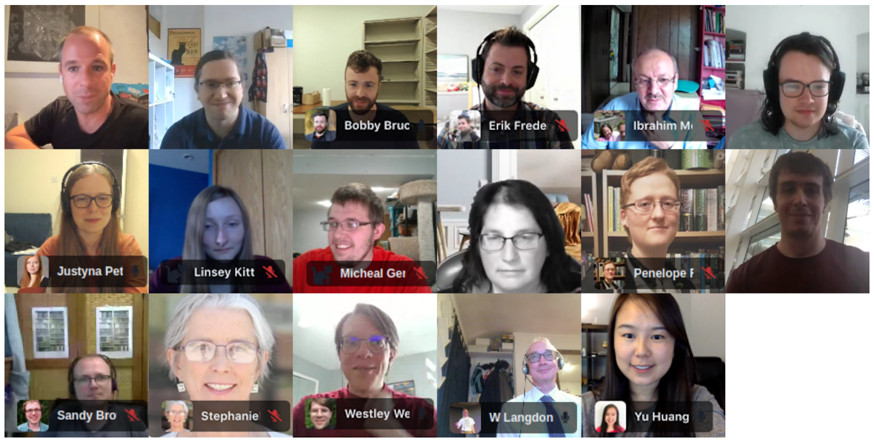

Figure 1. Screenshot of the workshop in progress. Images of all participants shown are reproduced with their consent.

Abstract

Following Dr. Stephanie Forrest of Arizona State University's keynote presentation there was a wide ranging discussion at the tenth international Genetic Improvement workshop, GI-2021 @ ICSE (held as part of the International Conference on Software Engineering on Sunday 30th May 2021). Topics included a growing range of target systems and applications, algorithmic improvements, wide-ranging questions about how other fields (especially evolutionary computation) can inform advances in GI, and about how GI is 'branded' to other disciplines.

We give a personal perspective on the workshop's proceedings, the discussions that took place, and resulting prospective directions for future research.

1 GI 2021

The tenth international workshop on Genetic Improvement was held as a fully-online event (Fig. 1) using Clowdr in conjunction with ICSE 2021. The full programme and list of accepted papers, with links to videos, can be found at `http://geneticimprovementofsoftware.com/events/icse2021.html`. This year represented a consolidation of the workshop. As the topic of GI has grown and found publications in other venues, parallel GI workshops have been hosted in multiple conferences in recent years since the first workshop at GECCO 2015, with three separate events running in 2020 (ICSE, GECCO and WCCI). The virtual setting allowed the community to meet in a single event while still drawing attendees from multiple communities. As such, the number of attendees at GI 2021 reached 34, and featured eleven papers [1], [2], [3], [4], [5] (Winner Best Presentation), [6], [7] (Winner Best Paper), [8], [9], [10], [11], in addition to the keynote talk and an extensive time of discussion. This year the workshop was also complemented by a tutorial at the Genetic and Evolutionary Computation Conference (GECCO) 2021, organised by ACM SIGEVO [12]. The tutorial serves as an introduction to GI and call for participation in the community, with the workshop providing an ideal forum for that participation to flourish.

The workshop keynote and all of the presentations were recorded and are available via YouTube `https://youtu.be/LVLdIb18cBg?list=PLI8fiFpB7BoKDaxvS7SQp0iA7fN7rrvDD`).

Particpants came from across four continents and seventeen time zones. (An edited record of last year's workshop was published in the ACM SIGSOFT's Software Engineering Notes [13] therefore we do not repeat that information here.)

2 What is Genetic Improvement

Genetic Improvement (GI) is a branch of Artificial Intelligence (AI) and Software Engineering which applies optimisation to improve existing programs. It is always possible to compare the new code with the existing code (effectively treating the program as its own specification) allowing GI to make measurable improvements to today's software. Improvements may be functional (e.g., does the new code have fewer bugs? does it have a new feature? does it give more accurate answers?) or non-functional (e.g. does it have better battery life? is it more reliable?)

3 The Programme

Perhaps it is the relatively small scale of the community, or simply the people involved, but the GI community is one of the most collegiate and friendly groups of researchers I have the pleasure to be a part of. As is the norm at these events, generous time allowances were made for discussion following each talk, with a mixture of formats: 15 minutes + 10 for questions for technical papers and 2 minutes + 8 for discussion for position papers.

The workshop began with an invited keynote given by Dr. Stephanie Forrest, director of the Biodesign Center for Biocomputing, Security and Society at Arizona State University, on the topic of *Engineering and Evolving Software*. The major theme of the talk was that GI, as it is practiced today, is not leveraging all the power of evolutionary computation, while evolutionary computation itself is a pale imitation of natural evolution. In order to solve bigger and more realistic problems in software improvement, we need to look at moving beyond single repairs, and the basic code mutation operators of copy, delete and some form of swap or move. Typically GI mutation acts, e.g. via the program's AST, to change the source code before compiling and testing the modification. We also need to better understand the 'landscape' of software mutations: i.e., the relationship between mutation operators, search algorithms, and optimisation objectives. This includes more use of gene-gene interaction (epistasis), better understanding of the software mutational robustness (neutrality) in these landscapes, understanding evolutionary drift, and thinking more seriously about selective pressure, so we can design algorithms that more efficiently and more effectively explore and exploit the space. Stephanie also explored how software development already shows aspects of natural evolution as libraries and components are recombined and incrementally added to build new systems. She also left an open challenge to the community: what could we do with the same amount of CPU power that's used for training deep learning models?

The broader point about drawing more from the evolutionary computation community became a running theme throughout the rest of the workshop: especially the consideration of search spaces including neutrality and interaction between multiple changes (epistasis), more sophisticated search algorithms, and moving beyond the simpler operators that remain popular.

The keynote was followed by formal presentations of papers, spanning a variety of topics. The major thing that was apparent this year was the increasing diversity of target systems that GI is being applied to: quantum computing [5], database SQL queries [2], procedural story generation for games [6], chemical reaction networks [7], dataflow programming [8], and emergent systems [4], all presenting their own challenges and unique questions compared to well-established application of GI to procedural and Object Orientated software. In addition to speed and functionality (bug fixing), applications also targeted optimisation of energy consumption [3] and generation of novelty [6], which also provided a nice link back to the keynote, which included some observations on novelty in natural evolution and the potential for novelty search algorithms in GI. Throughout, in part reflecting on the suggestions from the keynote, the discussions considered what might be learned from the wider Evolutionary Computing community on topics such as the neutrality in each space, the mutation operators being used, and how to efficiently handle the processing of many tests cases when evaluating the fitness of new solutions. Two speakers looking at deeper algorithmic questions considered whether weighting the choice of source code change operators to apply might ensure more even sampling [1] and refinement of Genetic Programming for evolving object oriented code, including weighting to bias the Genetic Programming operators based on context within the program, to generate valid new programs more often [9]. This very much aligns with a recent trend in the community of designing more sophisticated operators, and more intelligent targeting of operators to specific sections or types of code, and has the potential to represent a very interesting and fruitful research direction.

As has been the case in previous years, there was also some focus on software testing: including partial specifications for automated program repair [10], a new approach to optimisation of covering arrays [11], and discussions came back multiple times to the risk of overfitting to test suites and the need to avoid it.

4 Discussion and Future directions

An open discussion followed the main programme and award of best paper/speaker prizes, with the majority of participants staying on for a lively debate on possible future research directions. This included considerations of other ways we can draw inspiration from the evolutionary computation community, including approaches to handling noise, constraints, and long-running fitness functions. There may be scope for bringing in methods like different approaches to selection (e.g. lexicase selection to make more efficient use of test suites). There was also an interesting discussion on the challenges of including hardware in the loop.

Finally, a much bigger question for the community: should we be rebranding Genetic Improvement as AI or machine learning for software?

The workshop benefited from excellent organisation and certainly flourished in the online setting. In common with many other similar events, it is likely that future events will be hybrid if not also fully online.

References

[1] Marta Smigielska, Aymeric Blot, and Justyna Petke. Uniform edit selection for genetic improvement: Empirical analysis of mutation operator efficacy. In Justyna Petke, Bobby R. Bruce, Yu Huang, Aymeric Blot, Westley Weimer, and W. B. Langdon, editors, *GI @ ICSE 2021*, pages 1–8, internet, 30 May 2021. IEEE. URL: `http://dx.doi.org/10.1109/GI52543.2021.00009`.

[2] James Callan and Justyna Petke. Optimising SQL queries using genetic improvement. In Justyna Petke, Bobby R. Bruce, Yu Huang, Aymeric Blot, Westley Weimer, and W. B. Langdon, editors, *GI @ ICSE 2021*, pages 9–10, internet, 30 May 2021. IEEE. URL: `http://dx.doi.org/10.1109/GI52543.2021.00010`.

[3] Alexander E. I. Brownlee, Jason Adair, Saemundur O. Haraldsson, and John Jabbo. Exploring the accuracy – energy trade-off in machine learning. In Justyna Petke, Bobby R. Bruce, Yu Huang, Aymeric Blot, Westley Weimer, and W. B. Langdon, editors, *GI @ ICSE 2021*, pages 11–18, internet, 30 May 2021. IEEE. URL: `http://dx.doi.org/10.1109/GI52543.2021.00011`.

[4] Penelope Faulkner Rainford and Barry Porter. Open challenges in genetic improvement for emergent software systems. In Justyna Petke, Bobby R. Bruce, Yu Huang, Aymeric Blot, Westley Weimer, and W. B. Langdon, editors, *GI @ ICSE 2021*, pages 43–44, internet, 30 May 2021. IEEE. URL: `http://dx.doi.org/10.1109/GI52543.2021.00018`.

[5] George O'Brien and John Clark. Using genetic improvement to retarget quantum software on differing hardware. In Justyna Petke, Bobby R. Bruce, Yu Huang, Aymeric Blot, Westley Weimer, and W. B. Langdon, editors, *GI @ ICSE 2021*, pages 31–38, internet, 30 May 2021. IEEE. Winner Best Presentation. URL: `http://dx.doi.org/10.1109/GI52543.2021.00015`.

[6] Erik Fredericks and Byron DeVries. (genetically) improving novelty in procedural story generation. In Justyna Petke, Bobby R. Bruce, Yu Huang, Aymeric Blot, Westley Weimer, and W. B. Langdon, editors, *GI @ ICSE 2021*, pages 39–40, internet, 30 May 2021. IEEE. URL: `http://dx.doi.org/10.1109/GI52543.2021.00016`.

[7] Ibrahim Mesecan, Michael C. Gerten, James I. Lathrop, Myra B. Cohen, and Tomas Haddad Caldas. CRNRepair: Automated program repair of chemical reaction networks. In Justyna Petke, Bobby R. Bruce, Yu Huang, Aymeric Blot, Westley Weimer, and W. B. Langdon, editors, *GI @ ICSE 2021*, pages 23–30, internet, 30 May 2021. IEEE. Winner Best Paper. URL: `http://dx.doi.org/10.1109/GI52543.2021.00014`.

[8] Yu Huang, Hammad Ahmad, Stephanie Forrest, and Westley Weimer. Applying automated program repair to dataflow programming languages. In Justyna Petke, Bobby R. Bruce, Yu Huang, Aymeric Blot, Westley Weimer, and W. B. Langdon, editors, *GI @ ICSE 2021*, pages 21–22, internet, 30 May 2021. IEEE. URL: `http://dx.doi.org/10.1109/GI52543.2021.00013`.

[9] Vicente Illanes and Alexandre Bergel. Generating objected-oriented source code using genetic programming. In Justyna Petke, Bobby R. Bruce, Yu Huang, Aymeric Blot, Westley Weimer, and W. B. Langdon, editors, *GI @ ICSE 2021*, pages 45–50, internet, 30 May 2021. IEEE. URL: `http://dx.doi.org/10.1109/GI52543.2021.00019`.

[10] Linsey Kitt and Myra B. Cohen. Partial specifications for program repair. In Justyna Petke, Bobby R. Bruce, Yu Huang, Aymeric Blot, Westley Weimer, and W. B. Langdon, editors, *GI @ ICSE 2021*, pages 19–20, internet, 30 May 2021. IEEE. URL: `http://dx.doi.org/10.1109/GI52543.2021.00012`.

[11] Ryan Dougherty and Xi Jiang. A permutation representation of covering arrays. In Justyna Petke, Bobby R. Bruce, Yu Huang, Aymeric Blot, Westley Weimer, and W. B. Langdon, editors, *GI @ ICSE 2021*, pages 41–42, internet, 30 May 2021. IEEE. URL: `http://dx.doi.org/10.1109/GI52543.2021.00017`.

[12] Sæmundur Ó. Haraldsson, Alexander Brownlee, John R. Woodward, Markus Wagner, and Bradley Alexander. Genetic improvement: Taking real-world source code and improving it using genetic programming. In Francisco Chicano et al., editors, *Proceedings of the 2020 Genetic and Evolutionary Computation Conference Companion*, GECCO '21, page 786–817, internet, July 10-14 2021. Association for Computing Machinery. GI tutorial. URL: `http://dx.doi.org/10.1145/3377929.3389885`.

[13] William B. Langdon, Westley Weimer, Justyna Petke, Erik Fredericks, Seongmin Lee, Emily Winter, Michail Basios, Myra B. Cohen, Aymeric Blot, Markus Wagner, Bobby R. Bruce, Shin Yoo, Simos Gerasimou, Oliver Krauss, Yu Huang, and Michael Gerten. Genetic improvement @ icse 2020. *SIGSOFT Software Engineering Notes*, 45(4):24–30, October 2020. URL: `https://arxiv.org/abs/2007.15987`, `doi:doi:10.1145/3417564.3417575`.