The 14th International Workshop on Genetic Improvement (GI @ ICSE 2025)

William B. Langdon
UCL, London, UK
w.langdon@cs.ucl.ac.uk

Aymeric Blot
Universite de Rennes, France
aymeric.blot@univ-rennes.fr

Oliver Krauss
FH Upper Austria, Austria
oliver.krauss@fh-hagenberg.at

ABSTRACT

The GI @ ICSE 2025 workshop, held 27 April, in addition to presentations, contained a keynote on the latest results on automated program generation for testing static program analyzers, which could have huge impact on GI for improving programs for APG, and a tutorial on Magpie, a widely used language independent GI tool. We summarise these, the papers, people, prizes, acknowledgements, discussions and hopes for the future.

1. INTRODUCTION

The fourteenth International Workshop on Genetic Improvement (GI 2025) was held in Ottawa, Ontario, Canada, co-located with the 47th International Conference on Software Engineering (ICSE 2025). The workshop offered both in-person and virtual participation through Zoom. Genetic Improvement (GI) is a research area focused on improving existing software using automated search techniques [51]. GI has been successfully applied to many software engineering tasks, as demonstrated by the wide range of work presented in the recent editions of the GI workshop. These include for example work on automated bug fixing [31, 32, 50, 59]; on optimisation of non-functional properties such as execution time [14, 24, 38, 58, 71], energy consumption [17], or network usage [21]; and on domain-specific context such as deep neural networks security [8], generative art [29], shoreline evolution forecasting models [1], or procedural story generation [27]. We have also observed a growing interest in work at the intersection of software improvements and large language models [70], investigating, for example, how LLMs can guide or accelerate automated software improvement or how GI techniques can be used to refine or adapt LLM-generated code [15, 33, 63]. GI research has won five "Humies" [6, 7, 34, 52, 65], prestigious cash prizes awarded for demonstrating human-competitive results at difficult-to-automate tasks. As of the workshop, the community website's GPbib-based living survey covers almost 600 GI-related work (Figure 1). As in previous editions, the workshop brought together GI researchers and enthusiasts, offering a productive space for exchanging ideas, fostering collaboration, and identifying new challenges and directions for the GI community.

2. WORKSHOP FORMAT

As per previous editions, this year's Genetic Improvement workshop was once again held as a full-day event. It took place on Sunday, April 27th, before the main ICSE 2025 conference. The complete program is available on the workshop website `http://geneticimprovementofsoftware.com/events/icse2025` as well as in the ICSE 2025 workshop proceedings [10]. GI @ ICSE 2025 featured one keynote, one tutorial, five research paper presentations, one position paper talk, and an open discussion session.

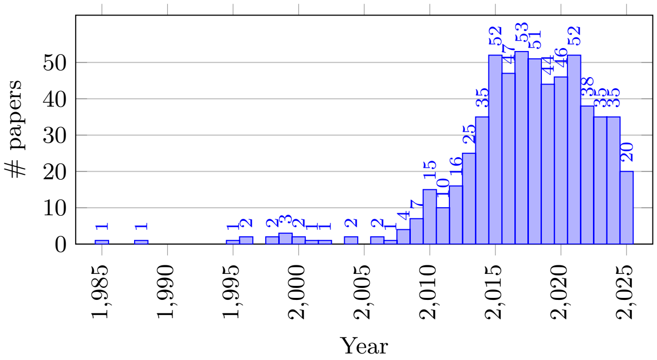
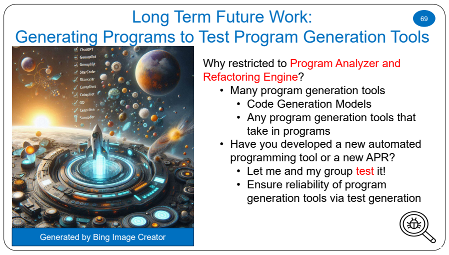


Figure 1: Publication year distribution of the 592 GI-related papers currently identified in the community website's living survey (`https://geneticimprovementofsoftware.com/learn/survey`)

Figure 2: Keynote speaker Dr. Shin Hwei Tan

Keynote Dr. Shin Hwei Tan:
Put on Your Tester Hat: Improving programs for Automated Program Generation.

This year we were treated to an invited keynote presentation by Dr. Shin Hwei Tan of Concordia University [59], Figure 2. Dr. Tan obtained her PhD degree from National University of Singapore with Abhik Roychoudhury, having previously studied for both undergraduate and masters degrees at the University of Illinois at Urbana-Champaign (USA). During her Masters degree she was co-advised by Darko Marinov and Lin Tan. Prior to joining Concordia two years ago, Dr. Tan was a lecturer in SUSTech (China). She organised the GI workshop, in 2019, when ICSE was also held in Canada (Montreal) [46], before founding the international series



Long Term Future Work:
Generating Programs to Test Program Generation Tools

Why restricted to Program Analyzer and Refactoring Engine?

- Many program generation tools
  - Code Generation Models
  - Any program generation tools that take in programs
- Have you developed a new automated programming tool or a new APR?
  - Let me and my group test it!
  - Ensure reliability of program generation tools via test generation

Figure 3: Put on Your Tester Hat, Dr. Shin Hwei Tan keynote. Genetic Improvement testing program generation tools [59].

of workshops on Automated Program Repair (APR) at ICSE in 2020. Next year she will be the general chair of the ACM's International Conference on the Foundations of Software Engineering (FSE 2026), when it will be hosted by Concordia University in Montreal.

Shin Hwei took as her title "Put on Your Tester Hat: Improving programs for Automated Program Generation", Figure 3 (see also tester's red hat in Figure 4). After a whistle stop tour through early work [53, 60–62, 64, 69], she concentrated upon automated ways to create tests to test static analysers. Like computer compilers, static analysers process computer programs. Today Artificial Intelligence (AI) is quite capable of generating small programs. However Dr. Tan proposed the clever bit was not just generating source code but also having ways to check that the source analyser was giving the right answer. I.e. having a test oracle [47]. Ways forward included metamorphic testing and comparing answers given by different static analysers. Although she noted that of the five static analysers she had tested, there were cases when they contained similar bugs. Nonetheless mutation operators which produced non-trivial semantically equivalent source code, should cause a static analyser to give the same answer as before the mutation. That is, if the outputs differ, the test has found a bug. Also production of nearly equivalent mutations may induce interesting corner-cases. These tests may cause the analyser to fail completely or it may fail to give "similar" answers.

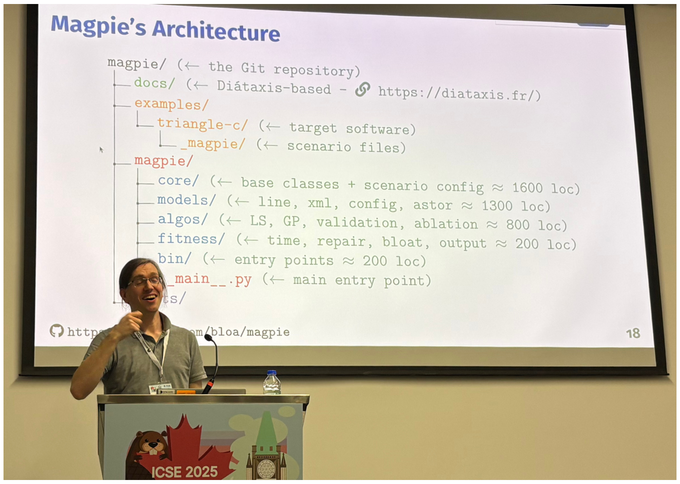
Also these ways of automatically generating source code might be good at finding bugs in source code refactoring tools.

The Q&A was joined by Prof. Roychoudhury. The discussion included possible ways to increase the tradeoff between performance and functionality. Also to what extent software engineers feel able to understand and interact with automatically generated program source code, as well as ways to ameliorate developer induced errors.

The keynote slides are available via `http://gpbib.cs.ucl.ac.uk/gi2025/gi_2025_slides/GIworkshop-keynote.pdf`

Tutorial.
Dr. Aymeric Blot (Figure 5) gave an extensive review of his Magpie system [11], tracing its development from the earlier PyGGI [3, 4] GI framework. Magpie builds upon the capabilities of PyGGI, showcasing compatibility with any programming language and support for improving both functional and non-functional aspects of software. However, Magpie also introduces novel features such as an improved user interface, the addition of parameter configuration to complement program source code manipulation, and the support for a much broader range of local search, genetic programming, and validation algorithms. During the tutorial, participants gained insights into the framework's structure, philosophy,

Figure 4: Shin Hwei Tan Keynote "Put on Your Tester Hat"



Magpie's Architecture

```
magpie/ (← the Git repository)
├─docs/ (← Diátaxis-based - 🔗 https://diataxis.fr/)
├─examples/
│ └─triangle-c/ (← target software)
│   └─_magpie/ (← scenario files)
├─magpie/
│ ├─core/ (← base classes + scenario config ≈ 1600 loc)
│ ├─models/ (← line, xml, config, astor ≈ 1300 loc)
│ ├─algos/ (← LS, GP, validation, ablation ≈ 800 loc)
│ ├─fitness/ (← time, repair, bloat, output ≈ 200 loc)
│ ├─bin/ (← entry points ≈ 200 loc)
│ └─_main__.py (← main entry point)
└─ts/
```

Figure 5: Dr. Aymeric Blot's Magpie Tutorial [9]

key components, and newest developments, whilst also engaging with practical examples. Dr. Blot concluding by outlining future directions and challenges.

Magpie is free and open source, accessible on GitHub at `https://github.com/bloa/magpie`. The tutorial slides are available via `http://www0.cs.ucl.ac.uk/staff/a.blot/files/slides/blot_gi@icse_2025.pdf`

Paper presentations. This year, the GI workshop accepted five research papers [14, 15, 40, 58, 63] and one position paper [24]. Each having received three independent reviews from the workshop's programme committee (Section 6). The authors of the accepted research papers had 20 minutes for the presentation and 10 for questions (Figures 10–14), whilst the accepted position paper had 10 minutes for the presentation and 5 minutes for questions (Figure 15).

Figure 7 shows some of the workshop participants.

VOTING AND DISCUSSIONS
VOTE
Link on Zoom: h...cs.google.com/forms/d/1H0krragGR1e4tY1mhMhGMXSU-HjdAcUs00TjW_80dyw

Figure 6: Aymeric Blot leading online and in person discussion.

Awards. Following the GI workshop tradition, the best paper award is given to researchers for their outstanding contributions to the Genetic Improvement field, and is decided based on reviews provided by the programme committee (Section 6). Similarly the best presentation award followed tradition and was decided by a vote amongst the participants of the workshop (Figure 6).

Best research paper award: "Large Language Model based Code Completion is an Effective Genetic Improvement Mutation" by Jingyuan Wang and Carol Hanna and Justyna Petke [63] (Figure 8).

Best presentation award: Dimitrios Bouras for "LLM-Assisted Crossover in Genetic Improvement of Software" [15] (Figure 9), which was delivered via the Internet from Peking University in China.

The workshop was followed by a social event along with the other ICSE 2025 workshops. Daniel Amyot persuaded the Cafe Nostalgica (Figure 16) to open especially and exclusively for ICSE on Sunday evening.

GI and LLMs:

The GI @ ICSE 2025 workshop featured two research papers that leveraged large language models (LLMs) to enhance the genetic improvement process.

LLM-Assisted Crossover [15]: This work proposed a novel LLM-based approach to the crossover operation in genetic programming. Implemented within the MAGPIE framework, the LLM-assisted crossover uses contextual information to intelligently select and combine edits from parent variants. Across seven benchmarks, it outperformed five traditional crossover methods in producing high-quality, performant, and viable program variants. Key results included an 8.5% improvement in fitness and 4.8% more viable variants on average, demonstrating the potential of LLMs to guide search more effectively.

LLM-based Masking Mutation [63]: This paper introduced a new mutation operator for GI, based on LLM code completion. Rather than replacing entire functions, this masking mutation replaces only selected statements using LLM-generated completions. Evaluated across five Java projects, the approach produced more valid, compiling, and test-passing patches than both traditional mutations and a previous LLM-based replacement operator. It also reduced response time by up to 60.7%. However, combining it with traditional mutations yielded mixed results, pointing to future work on integration strategies.

Together, these papers exemplify how LLMs can be embedded into core GI operations—mutation and crossover—demonstrating tangible benefits in patch quality, performance, and efficiency. They signal a growing research direction at the intersection of software improvement and modern AI capabilities.

3. DISCUSSION/FUTURE TOPICS

Genetic Improvement [35, 51, 67] has been successfully applied across a wide range of software engineering tasks, including automated bug fixing [48, 65], optimising non-functional properties—such as execution time [5, 25, 42], memory usage [22, 41, 56, 68], energy consumption [2, 13, 19, 20, 26], among others [1, 12, 21]—and functionality transplantation from one system to another [6, 52]. Beyond general-purpose applications, GI has also been used in domain-specific contexts, such as bioinformatics [44, 49] and generative art [28]. This is demonstrated by the wide range of work presented in the recent editions of the GI workshop.

3.1 Future: AI for program improvement

We are already seeing great interest in using the latest artificial intelligence tools in GI. We have already mentioned that two papers presented at the workshop used large language models (LLMs) [15, 63]. Indeed LLMs for GI were first presented at an earlier workshop [33]. Other early works include [18, 54, 57]. Indeed our Keynote speaker Dr. Shin Hwei Tan, thinks AI will play and increasing role in automatic testing, Figure 3. We expect researchers to continue to explore the intersection of GI and large language models. Future directions may include: enhancing explainability of GI transformations, integrating with developer tools, and improving performance predictability.

3.2 Extension of Recent Work

Following the success of GI @ ICSE 2025, the community anticipates continued collaboration and a potential workshop at ICSE 2026. We also encourage submission of extended versions of accepted papers to a future GI special issue of Springer-Nature's Automated Software Engineering journal.

4. WORKSHOP OUTCOMES

Following the success of the earlier Genetic Improvement special issues of the ASE journal, we anticipate that in future some authors of accepted papers will be invited to submit their extended work to future ASE special issues. (Special Issue editors: Oliver Krauss and Vesna Nowack) [23, 30, 36].

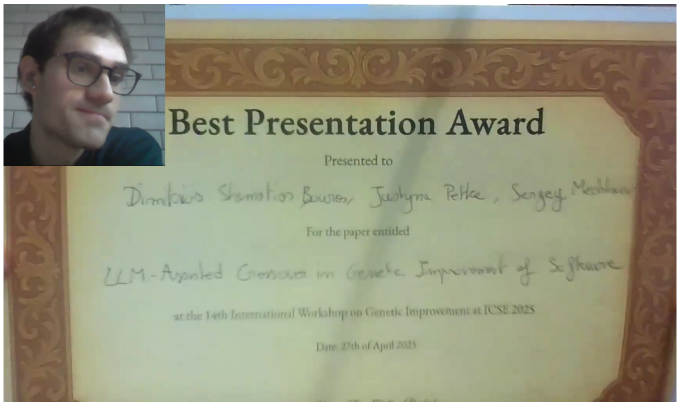
As with earlier workshops [16, 39, 43, 45, 46], there will be a short write up in the ACM SIGSOFT SEN newsletter (this document).

72
Tester Perspective: Coverage
Genetic Improvement Engine
Software Quality Standard
Maintainability
Portability
Security
Functional Suitability
ISO/IEC 25010
Reliability
Performance Efficiency
Usability
Compatibility
What are the criteria to cover?
Tester
✓ My talk m cover "Maintainability" and "Reliability"
✓ Most GI ocus on "Perfor ce Efficiency"
✓ How abo vement of oth ty aspects?
ICSE 2025
ttawa

Figure 7: Some of GI @ ICSE 2025 workshop participants: Xi Zhang, Aymeric Blot, Bill Langdon, Shin Hwei Tan, Zoe Aghababaeyan, Carol Hanna, Thanatad Songpetchmongkol, Haibo Wang.

Best Paper Award
Presented to
n Wang, Carol Hanna, and Justyna Petke
For the paper entitled
e Language Model based Code Completion
n Effective Genetic Improvement Mutation
ternational Workshop on Genetic Improvement at ICSE 2025
Date: 27th of April 2025
Me to Everyone
Link for voting for the best presentation: https://docs.google.com/forms/d/1H0krragGR1e4tY1mhMhGMXSU-HjdAcUs00TjW_80dyw
justyna to Hosts and panelists
From my experience people are not as excited about faster code than bug fixing for instance. We have issued lots of pull requests that are just left there as they don't have high priority. Even with LLMs, although people know they are expensive, people use then anyway to get the results they want (just my two cents)
I agree that GI in a way is noe more environmentally friendly and it does not require vast amounts of data, totally agree with current speaker
Who can see your messages? Recording on
To: Everyone
Type message here...
Talking: Encore Show43

Figure 8: Shin Hwei Tan presenting the Best Paper award to Carol Hanna for "Large Language Model based Code Completion is an Effective Genetic Improvement Mutation" by Jingyuan Wang and Carol Hanna and Justyna Petke [63]. Workshop organiser Aymeric Blot to left.



Best Presentation Award

Presented to

Dimitrios Stamatios Bouras, Justyna Petke, Sergey Mechtaev

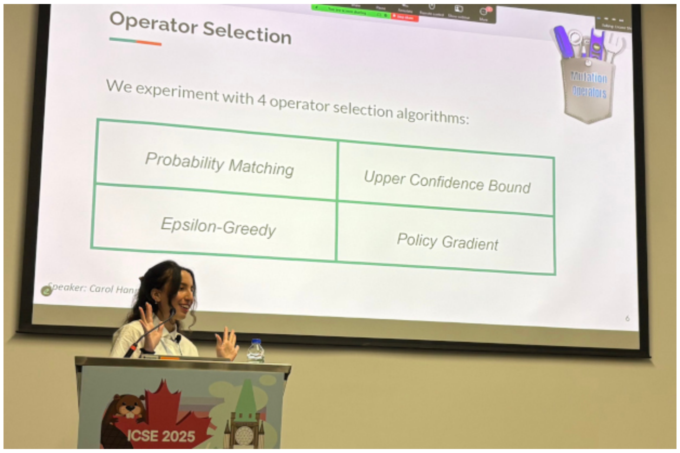
For the paper entitled

LLM-Assisted Crossover in Genetic Improvement of Software

at the 14th International Workshop on Genetic Improvement at ICSE 2025

Date: 27th of April 2025

Figure 9: Shin Hwei Tan presented the Best Presentation award via zoom to Dimitrios Stamatios Bouras for "LLM-Assisted Crossover in Genetic Improvement of Software" by Dimitrios Stamatios Bouras, Justyna Petke, and Sergey Mechtaev [15].

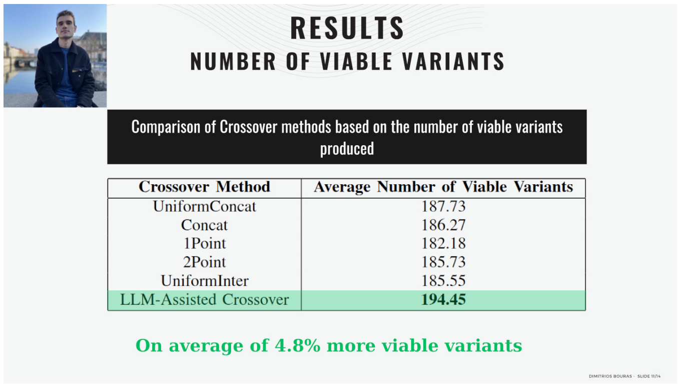


Operator Selection

We experiment with 4 operator selection algorithms:

Probability Matching	Upper Confidence Bound
Epsilon-Greedy	Policy Gradient

Figure 10: Carol Hanna presenting the Best Paper "Large Language Model based Code Completion is an Effective Genetic Improvement Mutation" by Jingyuan Wang and Carol Hanna and Justyna Petke [63].



RESULTS

NUMBER OF VIABLE VARIANTS

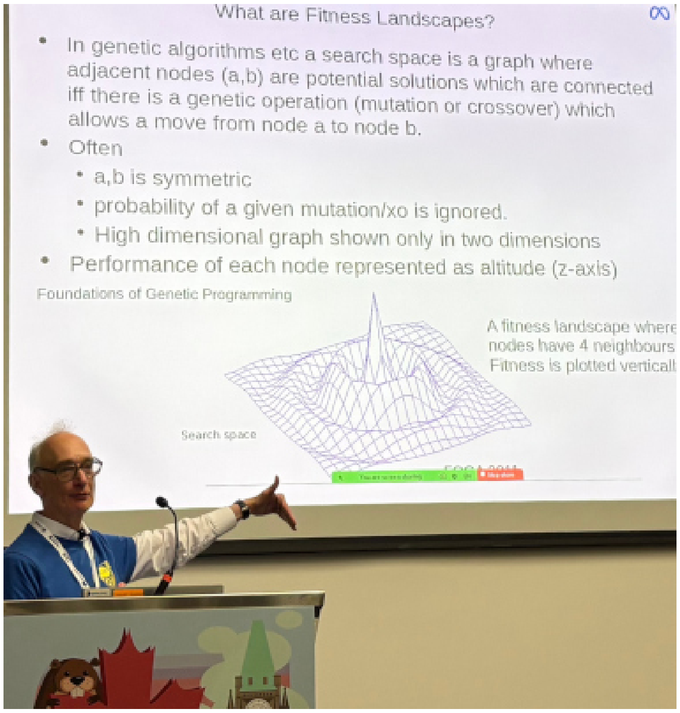
Comparison of Crossover methods based on the number of viable variants produced

Crossover Method	Average Number of Viable Variants
UniformConcat	187.73
Concat	186.27
1Point	182.18
2Point	185.73
UniformInter	185.55
LLM-Assisted Crossover	**194.45**

On average of 4.8% more viable variants

DIMITRIOS BOURAS · SLIDE 11/14

Figure 11: Dimitrios Bouras won the best presentation award: "LLM-Assisted Crossover in Genetic Improvement of Software" by Dimitrios Stamatios Bouras and Justyna Petke and Sergey Mechtaev [15].



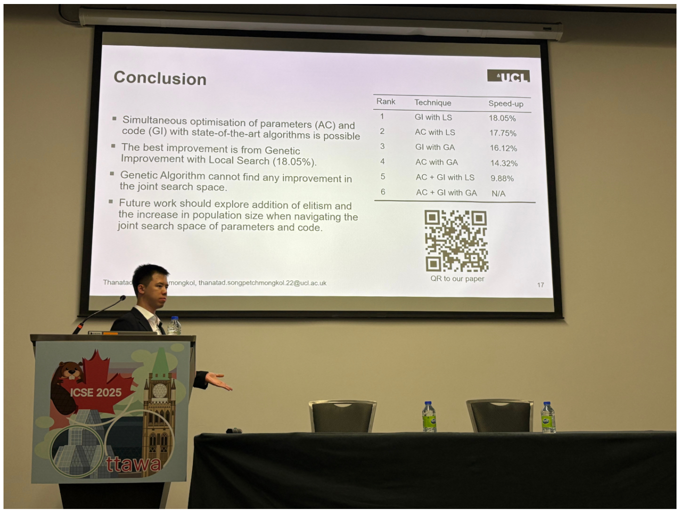
What are Fitness Landscapes?

- In genetic algorithms etc a search space is a graph where adjacent nodes (a,b) are potential solutions which are connected iff there is a genetic operation (mutation or crossover) which allows a move from node a to node b.
- Often
  - a,b is symmetric
  - probability of a given mutation/xo is ignored.
  - High dimensional graph shown only in two dimensions
- Performance of each node represented as altitude (z-axis)

Foundations of Genetic Programming

A fitness landscape where nodes have 4 neighbours Fitness is plotted verticall

Figure 12: Bill Langdon [40]



Conclusion

- Simultaneous optimisation of parameters (AC) and code (GI) with state-of-the-art algorithms is possible
- The best improvement is from Genetic Improvement with Local Search (18.05%).
- Genetic Algorithm cannot find any improvement in the joint search space.
- Future work should explore addition of elitism and the increase in population size when navigating the joint search space of parameters and code.

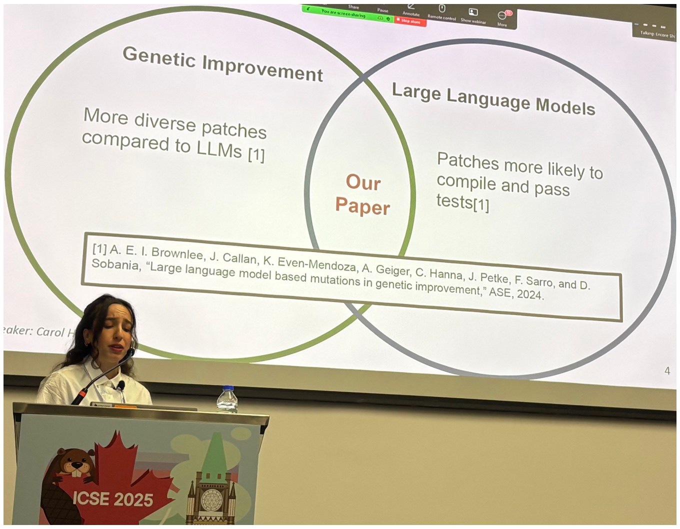
Rank	Technique	Speed-up
1	GI with LS	18.05%
2	AC with LS	17.75%
3	GI with GA	16.12%
4	AC with GA	14.32%
5	AC + GI with LS	9.88%
6	AC + GI with GA	N/A

QR to our paper

Thanatad mongkol, thanatad.songpetchmongkol.22@ucl.ac.uk

17

Figure 13: Thanatad Songpetchmongkol [58]



Genetic Improvement

More diverse patches compared to LLMs [1]

Large Language Models

Patches more likely to compile and pass tests[1]

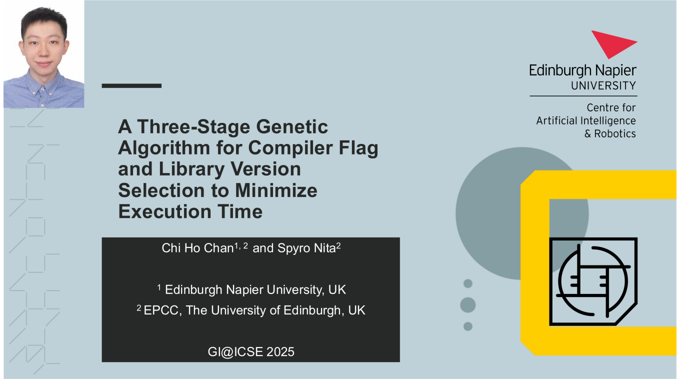
Our Paper

[1] A. E. I. Brownlee, J. Callan, K. Even-Mendoza, A. Geiger, C. Hanna, J. Petke, F. Sarro, and D. Sobania, "Large language model based mutations in genetic improvement," ASE, 2024.

eaker: Carol H

4

Figure 14: Carol Hanna presenting "Enhancing Software Runtime with Reinforcement Learning-Driven Mutation Operator Selection in Genetic Improvement" by Damien Bose, Carol Hanna and Justyna Petke [14].



Edinburgh Napier UNIVERSITY

Centre for Artificial Intelligence & Robotics

A Three-Stage Genetic Algorithm for Compiler Flag and Library Version Selection to Minimize Execution Time

Chi Ho Chan[1, 2] and Spyro Nita[2]

[1] Edinburgh Napier University, UK

[2] EPCC, The University of Edinburgh, UK

GI@ICSE 2025

Figure 15: Chi Ho Chan presented "A Three-Stage Genetic Algorithm for Compiler Flag and Library Version Selection to Minimize Execution Time" by Chi Ho Chan and Spyro Nita [24] via the Internet.

Figure 16: Social event: Cafe Nostalgica, University of Ottawa. Right: Korean Canadian dish kimchi poutine.

5. GI 2024 WORKSHOP ORGANISERS

Aymeric Blot

Vesna Nowack

Penn Faulkner Rainford

Oliver Krauss

6. GI 2025 PROGRAMME COMMITTEE

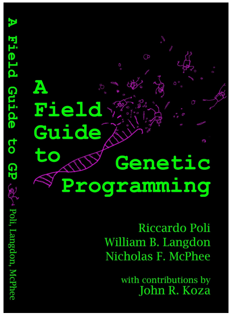
Each submission received three independent reviews from the workshop's programme committee (see Figure 17).

In addition to providing feedback to the authors and deciding which submissions to accept, the best paper award (Figure 8) was decided by the organisers using the reviewers' comments.

Acknowledgement

We would like to thank: our ICSE 2025 student volunteer, photographer "Dif" Thanatad Songpetchmongkol, and session chairs: Justyna Petke, Carol Hanna and Thanatad Songpetchmongkol. Also our thanks to Carol Hanna for additional photographs and to Daniel Amyot for organising the very enjoyable ICSE 2025 workshops' social event at Cafe Nostalgica on Sunday evening (Figure 16).

Sponsored by A Field Guide to Genetic Programming [55].



7. REFERENCES

[1] Mahmoud Al Najar, Rafael Almar, Erwin W. J. Bergsma, Jean-Marc Delvit, and Dennis G. Wilson. Genetic improvement of shoreline evolution forecasting models. In Bobby R. Bruce, Vesna Nowack, Aymeric Blot, Emily Winter, W. B. Langdon, and Justyna Petke, editors, *GI @ GECCO 2022*, pages 1916–1923, Boston, USA, 9 July 2022. Association for Computing Machinery.

[2] Akram Alofi, Mahmoud A. Bokhari, Rami Bahsoon, and Robert J. Hendley. Optimizing the energy consumption of blockchain-based systems using evolutionary algorithms: A new problem formulation. *IEEE Transactions on Sustainable Computing*, 7(4):910–922, 2022.

[3] Gabin An, Aymeric Blot, Justyna Petke, and Shin Yoo. PyGGI 2.0: Language independent genetic improvement framework. In Sven Apel and Alessandra Russo, editors, *Proceedings of the 27th Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering ESEC/FSE 2019)*, pages 1100–1104, Tallinn, Estonia, August 26–30 2019. ACM.

[4] Gabin An, Jinhan Kim, and Shin Yoo. Comparing line and AST granularity level for program repair using PyGGI. In Justyna Petke, Kathryn Stolee, William B. Langdon, and Westley Weimer, editors, *GI-2018, ICSE workshops proceedings*, pages 19–26, Gothenburg, Sweden, 2 June 2018. ACM.

[5] Andrea Arcuri, David Robert White, John Clark, and Xin Yao. Multi-objective improvement of software using co-evolution and smart seeding. In Xiaodong Li, Michael Kirley, Mengjie Zhang, David G. Green, Victor Ciesielski, Hussein A. Abbass, Zbigniew Michalewicz, Tim Hendtlass, Kalyanmoy Deb, Kay Chen Tan, Jürgen Branke, and Yuhui Shi, editors, *Proceedings of the 7th International Conference on Simulated Evolution And Learning (SEAL '08)*, volume 5361 of *Lecture Notes in Computer Science*, pages 61–70, Melbourne, Australia, December 7-10 2008. Springer.

[6] Earl T. Barr, Mark Harman, Yue Jia, Alexandru Marginean, and Justyna Petke. Automated software transplantation. In Tao Xie and Michal Young, editors, *International Symposium on Software Testing and Analysis, ISSTA 2015*, pages 257–269, Baltimore, Maryland, USA, 14-17 July 2015. ACM. ACM SIGSOFT Distinguished Paper Award.

[7] Michail Basios, Lingbo Li, Fan Wu, Leslie Kanthan, and Earl T. Barr. Darwinian data structure selection. In Gary T. Leavens, Alessandro Garcia, and Corina S. Pasareanu, editors, *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering, ESEC/FSE 2018*, pages 118–128, Lake Buena Vista, FL, USA, 4-9 November 2018. ACM.

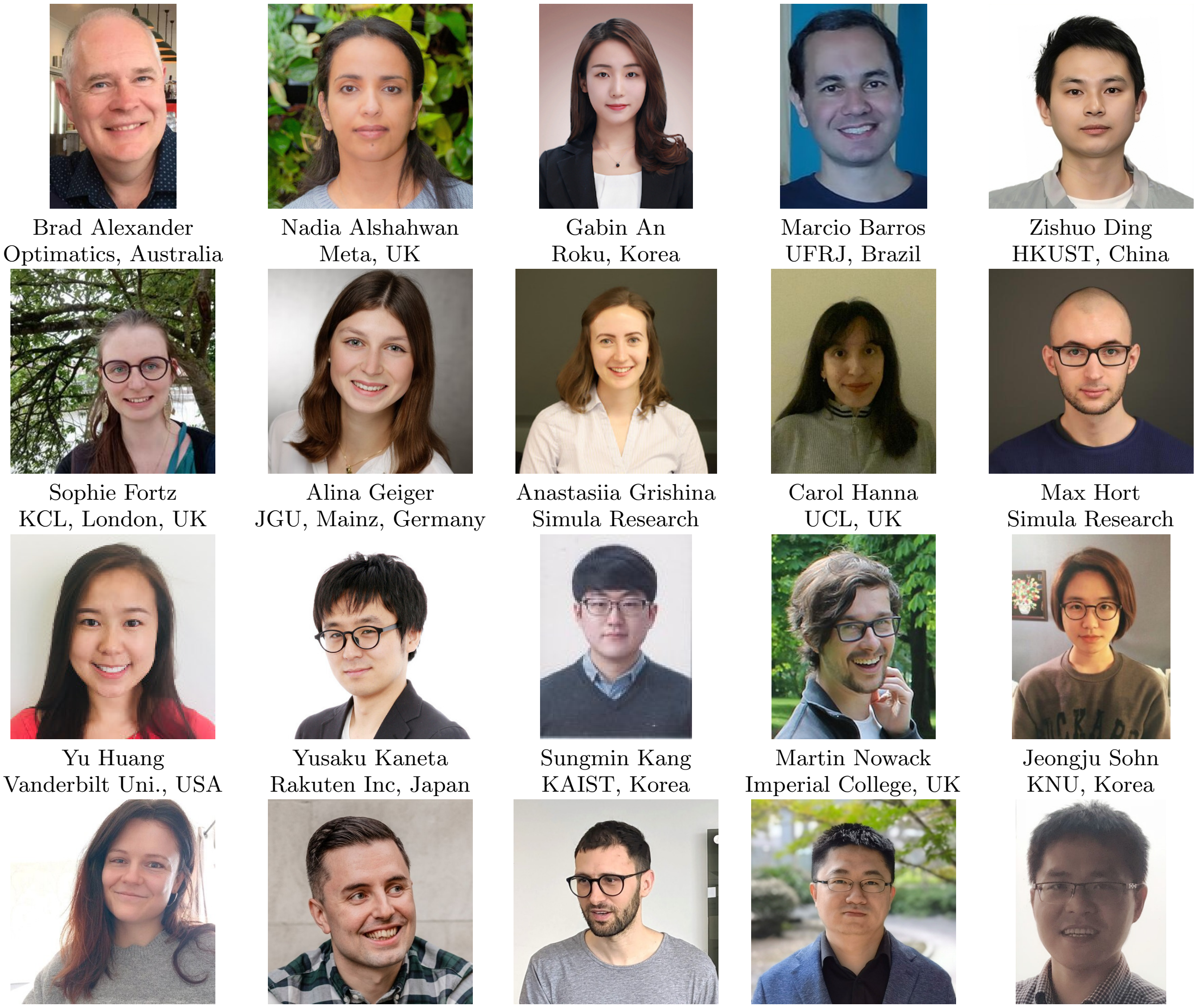
[8] Hunter Baxter, Yu Huang, and Kevin Leach. Genetic improvement for DNN security. In Gabin An, Aymeric Blot, Vesna Nowack, Oliver Krauss, and Justyna Petke, editors, *13th International Workshop on Genetic Improvement @ICSE 2024*, pages 11–12, Lisbon, 16 April 2024. ACM. Best Presentation.

[9] Aymeric Blot. Automated software performance improvement with Magpie. In Aymeric Blot, Vesna Nowack, Penn Faulkner Rainford, and Oliver Krauss, editors, *14th International Workshop on Genetic Improvement @ICSE 2025*, page vi, Ottawa, 27 April 2025. IEEE. Invited tutorial.

[10] Aymeric Blot, Vesna Nowack, Penn Faulkner Rainford, and Oliver Krauss, editors. *14th International Workshop on Genetic Improvement @ICSE 2025*, Ottawa, 27 April 2025. IEEE.

[11] Aymeric Blot and Justyna Petke. MAGPIE: Machine automated general performance improvement via evolution of software. arXiv, 4 August 2022.

[12] Aymeric Blot and Justyna Petke. Using genetic improvement to optimise optimisation algorithm implementations. In Khaled Hadj-Hamou, editor, *23ème congrès annuel de la Société Française de Recherche Opérationnelle et d'Aide à la Décision, ROADEF'2022*, Villeurbanne - Lyon, France, 23–25 February 2022. INSA Lyon.



Brad Alexander
Optimatics, Australia

Nadia Alshahwan
Meta, UK

Gabin An
Roku, Korea

Marcio Barros
UFRJ, Brazil

Zishuo Ding
HKUST, China

Sophie Fortz
KCL, London, UK

Alina Geiger
JGU, Mainz, Germany

Anastasiia Grishina
Simula Research

Carol Hanna
UCL, UK

Max Hort
Simula Research

Yu Huang
Vanderbilt Uni., USA

Yusaku Kaneta
Rakuten Inc, Japan

Sungmin Kang
KAIST, Korea

Martin Nowack
Imperial College, UK

Jeongju Sohn
KNU, Korea

Sarah Thomson
Napier University, UK

Christopher Timperley
CMU

Michele Tufano
Microsoft, USA

Jifeng Xuan
Wuhan Uni., China

Yuan Yuan
MSU, USA

Figure 17: GI @ ICSE 2025 Reviewers

[13] Mahmoud Bokhari and Markus Wagner. Optimising energy consumption heuristically on Android mobile phones. In Justyna Petke, David R. White, and Westley Weimer, editors, *Genetic Improvement 2016 Workshop*, pages 1139–1140, Denver, July 20-24 2016. ACM.

[14] Damien Bose, Carol Hanna, and Justyna Petke. Enhancing software runtime with reinforcement learning-driven mutation operator selection in genetic improvement. In Aymeric Blot, Vesna Nowack, Penn Faulkner Rainford, and Oliver Krauss, editors, *14th International Workshop on Genetic Improvement @ICSE 2025*, Ottawa, 27 April 2025. forthcoming.

[15] Dimitrios Stamatios Bouras, Justyna Petke, and Sergey Mechtaev. Llm-assisted crossover in genetic improvement of software. In Aymeric Blot, Vesna Nowack, Penn Faulkner Rainford, and Oliver Krauss, editors, *14th International Workshop on Genetic Improvement @ICSE 2025*, Ottawa, 27 April 2025. Best presentation.

[16] Alexander E. I. Brownlee. Genetic Improvement @ ICSE 2021: Personal reflection of a workshop participant. *SIGSOFT Software Engineering Notes*, 46(4):28–30, October 2021.

[17] 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.

[18] Alexander Edward Ian Brownlee, James Callan, Karine Even-Mendoza, Alina Geiger, Carol Hanna, Justyna Petke, Federica Sarro, and Dominik Sobania. Large language model based mutations in genetic improvement. *Automated Software Engineering*, 15:article number 15, 2025. Special Issue on Advances in Search-Based Software.

[19] Bobby R. Bruce. *The Blind Software Engineer: Improving the Non-Functional Properties of Software by Means of Genetic Improvement.* PhD thesis, Computer Science, University College, London, UK, 12 July 2018.

[20] Bobby R. Bruce, Justyna Petke, Mark Harman, and Earl T. Barr. Approximate oracles and synergy in software energy search spaces. *IEEE Transactions on Software Engineering*, 45(11):1150–1169, November 2019.

[21] James Callan, William B. Langdon, and Justyna Petke. On reducing network usage with genetic improvement. In Gabin An, Aymeric Blot, Vesna Nowack, Oliver Krauss, and Justyna Petke, editors, *13th International Workshop on Genetic Improvement @ICSE 2024*, pages 23–30, Lisbon, 16 April 2024. ACM.

[22] James Callan and Justyna Petke. Multi-objective genetic improvement: A case study with EvoSuite. In Mike Papadakis and Silvia Regina Vergilio, editors, *14th International Symposium on Search Based Software Engineering SSBSE 2020*, volume 13711 of *LNCS*, pages 111–117, Singapore, 17-18 November 2022. Springer.

[23] James Callan and Justyna Petke. Multi-objective improvement of android applications. *Automated Software Engineering*, 32:article number 2, 2025. Accepted 04 Nov 2024.

[24] Chi Ho Chan and Spyro Nita. A three-stage genetic algorithm for compiler flag and library version selection to minimize execution time. In Aymeric Blot, Vesna Nowack, Penn Faulkner Rainford, and Oliver Krauss, editors, *14th International Workshop on Genetic Improvement @ICSE 2025*, Ottawa, 27 April 2025. forthcoming.

[25] Fabio de A. Farzat, Marcio de O. Barros, and Guilherme H. Travassos. Evolving JavaScript code to reduce load time. *IEEE Transactions on Software Engineering*, 47(8):1544–1558, August 2021.

[26] Jonathan Dorn, Jeremy Lacomis, Westley Weimer, and Stephanie Forrest. Automatically exploring tradeoffs between software output fidelity and energy costs. *IEEE Transactions on Software Engineering*, 45(3):219–236, March 2019.

[27] 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.

[28] Erik M. Fredericks, Denton Bobeldyk, and Jared M. Moore. Crafting generative art through genetic improvement: Managing creative outputs in diverse fitness landscapes. In Stephan M. Winkler, Wolfgang Banzhaf, Ting Hu, and Alexander Lalejini, editors, *Genetic Programming Theory and Practice XXI*, Genetic and Evolutionary Computation, pages 321–335, University of Michigan, USA, June 6-8 2024. Springer.

[29] Erik M. Fredericks, Abigail C. Diller, and Jared M. Moore. Generative art via grammatical evolution. In Vesna Nowack, Markus Wagner, Gabin An, Aymeric Blot, and Justyna Petke, editors, *12th International Workshop on Genetic Improvement @ICSE 2023*, pages 1–8, Melbourne, Australia, 20 May 2023. IEEE. Best paper.

[30] Erik M. Fredericks, Jared M. Moore, and Abigail C. Diller. GenerativeGI: creating generative art with genetic improvement. *Automated Software Engineering*, 31:23, 2024.

[31] David Griffin, Susan Stepney, and Ian Vidamour. DebugNS: Novelty search for finding bugs in simulators. In Vesna Nowack, Markus Wagner, Gabin An, Aymeric Blot, and Justyna Petke, editors, *12th International Workshop on Genetic Improvement @ICSE 2023*, pages 17–18, Melbourne, Australia, 20 May 2023. IEEE.

[32] 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.

[33] Sungmin Kang and Shin Yoo. Towards objective-tailored genetic improvement through large language models. In Vesna Nowack, Markus Wagner, Gabin An, Aymeric Blot, and Justyna Petke, editors, *12th International Workshop on Genetic Improvement @ICSE 2023*, pages 19–20, Melbourne, Australia, 20 May 2023. IEEE. Best position paper.

[34] Joel Kuepper, Andres Erbsen, Jason Gross, Owen Conoly, Chuyue Sun, Samuel Tian, David Wu, Adam Chlipala, Chitchanok Chuengsatiansup, Daniel Genkin, Markus Wagner, and Yuval Yarom. CryptOpt: Verified compilation with randomized program search for cryptographic primitives. In Nate Foster, editor, *44th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2023*, page article no. 158, Orlando, Florida, 17-21 June 2023. Association for Computing Machinery. Gold winner 2023 HUMIES, PLDI Distinguished Paper.

[35] W. B. Langdon. Genetic improvement of programs. In Radomil Matousek, editor, *18th International Conference on Soft Computing, MENDEL 2012*, Brno, Czech Republic,

27-29 June 2012. Brno University of Technology. Invited keynote.

[36] W. B. Langdon and David Clark. Deep imperative mutations have less impact. *Automated Software Engineering*, 32:article number 6, 2025.

[37] W. B. Langdon and Riccardo Poli. Removal of the man-machine interface bottleneck "Do what I ment not what I said". In *Grand Challenges for Computing*, Edinburgh, 24-26 November 2002. Discussion paper.

[38] William B. Langdon and Bradley J. Alexander. Genetic improvement of OLC and H3 with Magpie. In Vesna Nowack, Markus Wagner, Gabin An, Aymeric Blot, and Justyna Petke, editors, *12th International Workshop on Genetic Improvement @ICSE 2023*, pages 9–16, Melbourne, Australia, 20 May 2023. IEEE.

[39] William B. Langdon, Gabin An, Aymeric Blot, Vesna Nowack, Justyna Petke, Shin Yoo, Oliver Krauss, Erik M. Fredericks, and Daniel Blackwell. The 13th international workshop on genetic improvement (GI @ ICSE 2024). *SIGSOFT Software Engineering Notes*, 49(3):42–50, July 2024.

[40] William B. Langdon and Bobby R. Bruce. The gem5 C++ glibc heap fitness landscape. In Aymeric Blot, Vesna Nowack, Penn Faulkner Rainford, and Oliver Krauss, editors, *14th International Workshop on Genetic Improvement @ICSE 2025*, Ottawa, 27 April 2025. forthcoming.

[41] William B. Langdon and David Clark. Genetic improvement of last level cache. In Mario Giacobini, Bing Xue, and Luca Manzoni, editors, *EuroGP 2024: Proceedings of the 27th European Conference on Genetic Programming*, volume 14631 of *LNCS*, pages 209–226, Aberystwyth, 3-5 April 2024. Springer Verlag.

[42] William B. Langdon and Mark Harman. Optimising existing software with genetic programming. *IEEE Transactions on Evolutionary Computation*, 19(1):118–135, February 2015.

[43] William B. Langdon, Vesna Nowack, Justyna Petke, Erik M. Fredericks, Gabin An, Aymeric Blot, Markus Wagner, and Hyeonseok Lee. Genetic Improvement @ ICSE 2023. *SIGSOFT Software Engineering Notes*, 48(4):51–59, October 2023.

[44] William B. Langdon, Justyna Petke, and Ronny Lorenz. Evolving better RNAfold structure prediction. In Mauro Castelli, Lukas Sekanina, and Mengjie Zhang, editors, *EuroGP 2018: Proceedings of the 21st European Conference on Genetic Programming*, volume 10781 of *LNCS*, pages 220–236, Parma, Italy, 4-6 April 2018. Springer Verlag.

[45] 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.

[46] William B. Langdon, Westley Weimer, Christopher Timperley, Oliver Krauss, Zhen Yu Ding, Yiwei Lyu, Nicolas Chausseau, Eric Schulte, Shin Hwei Tan, Kevin Leach, Yu Huang, and Gabin An. The state and future of genetic improvement. *SIGSOFT Software Engineering Notes*, 44(3):25–29, July 2019.

[47] William B. Langdon, Shin Yoo, and Mark Harman. Inferring automatic test oracles. In Juan P. Galeotti and Justyna Petke, editors, *Search-Based Software Testing*, pages 5–6, Buenos Aires, Argentina, 22-23 May 2017.

[48] Claire Le Goues, Michael Pradel, and Abhik Roychoudhury. Automated program repair. *Communications of the ACM*, 62(12):56–65, December 2019.

[49] Jhe-Yu Liou, Muaaz Awan, Kirtus Leyba, Petr Sulc, Steven Hofmeyr, Carole-Jean Wu, and Stephanie Forrest. Evolving to find optimizations humans miss: using evolutionary computation to improve GPU code for bioinformatics applications. *ACM Transactions on Evolutionary Learning and Optimization*, 4(4), December 2024.

[50] 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. best paper.

[51] Justyna Petke, Saemundur O. Haraldsson, Mark Harman, William B. Langdon, David R. White, and John R. Woodward. Genetic improvement of software: a comprehensive survey. *IEEE Transactions on Evolutionary Computation*, 22(3):415–432, June 2018.

[52] Justyna Petke, Mark Harman, William B. Langdon, and Westley Weimer. Specialising software for different downstream applications using genetic improvement and code transplantation. *IEEE Transactions on Software Engineering*, 44(6):574–594, June 2018.

[53] Justyna Petke, Shin Hwei Tan, William B. Langdon, and Westley Weimer, editors. *Proceedings 2019 ACM/IEEE 6th International Genetic Improvement Workshop, GI 2019*, Montreal, 28 May 2019.

[54] Giovanni Pinna, Damiano Ravalico, Luigi Rovito, Luca Manzoni, and Andrea De Lorenzo. Enhancing large language models-based code generation by leveraging genetic improvement. In Mario Giacobini, Bing Xue, and Luca Manzoni, editors, *EuroGP 2024: Proceedings of the 27th European Conference on Genetic Programming*, volume 14631 of *LNCS*, pages 108–124, Aberystwyth, 3-5 April 2024. Springer.

[55] Riccardo Poli, William B. Langdon, and Nicholas Freitag McPhee. *A field guide to genetic programming.* Published via `http://lulu.com` and freely available at `http://www.gp-field-guide.org.uk`, 2008. (With contributions by J. R. Koza).

[56] Jose L. Risco-Martin, J. Manuel Colmenar, J. Ignacio Hidalgo, Juan Lanchares, and Josefa Diaz. A methodology to automatically optimize dynamic memory managers applying grammatical evolution. *Journal of Systems and Software*, 91:109–123, 2014.

[57] Gudny Birna Saemundsdottir and Saemundur Oskar Haraldsson. Large language models as all-in-one operators for genetic improvement. In Dominik Sobania and Aymeric Blot, editors, *Proceedings of the 2024 Genetic and Evolutionary Computation Conference Companion*, GECCO '24, pages 727–730, Melbourne, Australia, 14-18 July 2024. Association for Computing Machinery.

[58] Thanatad Songpetchmongkol, Aymeric Blot, and Justyna Petke. Empirical comparison of runtime improvement approaches: Genetic improvement, parameter tuning, and their combination. In Aymeric Blot, Vesna Nowack, Penn Faulkner Rainford, and Oliver Krauss, editors, *14th International Workshop on Genetic Improvement @ICSE 2025*, Ottawa, 27 April 2025. forthcoming.

[59] Shin Hwei Tan. Put on your tester hat: Improving programs for automated program generation. In Aymeric

Blot, Vesna Nowack, Penn Faulkner Rainford, and Oliver Krauss, editors, *14th International Workshop on Genetic Improvement @ICSE 2025*, page v, Ottawa, 27 April 2025. IEEE. Invited Keynote.

[60] Shin Hwei Tan, Chunfeng Hu, Ziqiang Li, Xiaowen Zhang, and Ying Zhou. Github-oss fixit: Fixing bugs at scale in a software engineering course. In *2021 IEEE/ACM 43rd International Conference on Software Engineering: Software Engineering Education and Training (ICSE-SEET)*, pages 1–10, 2021.

[61] Shin Hwei Tan, Jooyong Yi, Sergey Mechtaev, Abhik Roychoudhury, et al. Codeflaws: a programming competition benchmark for evaluating automated program repair tools. In *Proceedings of the 39th International Conference on Software Engineering Companion*, pages 180–182. IEEE Press, 2017.

[62] Huanting Wang, Guixin Ye, Zhanyong Tang, Shin Hwei Tan, Songfang Huang, Dingyi Fang, Yansong Feng, Lizhong Bian, and Zheng Wang. Combining graph-based learning with automated data collection for code vulnerability detection. *IEEE Transactions on Information Forensics and Security*, 16:1943–1958, 2021.

[63] Jingyuan Wang, Carol Hanna, and Justyna Petke. Large language model based code completion is an effective genetic improvement mutation. In Aymeric Blot, Vesna Nowack, Penn Faulkner Rainford, and Oliver Krauss, editors, *14th International Workshop on Genetic Improvement @ICSE 2025*, Ottawa, 27 April 2025. Best paper.

[64] Ying Wang, Ming Wen, Rongxin Wu, Zhenwei Liu, Shin Hwei Tan, Zhiliang Zhu, Hai Yu, and Shing-Chi Cheung. Could i have a stack trace to examine the dependency conflict issue? In *Proceedings of the 41st International Conference on Software Engineering*, ICSE '19, pages 572–583, Piscataway, NJ, USA, 2019. IEEE Press.

[65] Westley Weimer, ThanhVu Nguyen, Claire Le Goues, and Stephanie Forrest. Automatically finding patches using genetic programming. In Stephen Fickas, editor, *International Conference on Software Engineering (ICSE) 2009*, pages 364–374, Vancouver, May 16-24 2009. Winner ACM SIGSOFT Distinguished Paper Award. Gold medal at 2009 HUMIES. Ten-Year Most Influential Paper [66].

[66] Westley Weimer, ThanhVu Nguyen, Claire Le Goues, and Stephanie Forrest. It does what you say, not what you mean: Lessons from a decade of program repair. ICSE 2019 Plenary Most Inflential Paper, 30 May 2019.

[67] David R. White, Andrea Arcuri, and John A. Clark. Evolutionary improvement of programs. *IEEE Transactions on Evolutionary Computation*, 15(4):515–538, August 2011.

[68] David R. White, John Clark, Jeremy Jacob, and Simon M. Poulding. Searching for resource-efficient programs: low-power pseudorandom number generators. In Maarten Keijzer, Giuliano Antoniol, Clare Bates Congdon, Kalyanmoy Deb, Benjamin Doerr, Nikolaus Hansen, John H. Holmes, Gregory S. Hornby, Daniel Howard, James Kennedy, Sanjeev Kumar, Fernando G. Lobo, Julian Francis Miller, Jason Moore, Frank Neumann, Martin Pelikan, Jordan Pollack, Kumara Sastry, Kenneth Stanley, Adrian Stoica, El-Ghazali Talbi, and Ingo Wegener, editors, *GECCO '08: Proceedings of the 10th annual conference on Genetic and evolutionary computation*, pages 1775–1782, Atlanta, GA, USA, 12-16 July 2008. ACM.

[69] Jooyong Yi, Umair Z Ahmed, Amey Karkare, Shin Hwei Tan, and Abhik Roychoudhury. A feasibility study of using automated program repair for introductory programming assignments. In *Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering*, pages 740–751. ACM, 2017.

[70] Shin Yoo. Executing one's way out of the Chinese Room. In Gabin An, Aymeric Blot, Vesna Nowack, Oliver Krauss, and Justyna Petke, editors, *13th International Workshop on Genetic Improvement @ICSE 2024*, page viii, Lisbon, 16 April 2024. ACM. Invited Keynote.

[71] James Zhong, Max Hort, and Federica Sarro. Py2Cy: A genetic improvement tool to speed up python. In Bobby R. Bruce, Vesna Nowack, Aymeric Blot, Emily Winter, W. B. Langdon, and Justyna Petke, editors, *GI @ GECCO 2022*, pages 1950–1955, Boston, USA, 9 July 2022. Association for Computing Machinery.