Search-Based Software Engineering in the Era of Modern Software Systems

Federica Sarro
Department of Computer Science
University College London
London, United Kingdom
Email: f.sarro@ucl.ac.uk

***Abstract*—This short paper accompanies the keynote given by Federica Sarro at the 31st IEEE International Requirements Engineering Conference, Hanover, Germany, September 2023.**

***Index Terms*—Search-Based Software Engineering, Responsible Software Engineering, Software Fairness, Software Footprint, On-line Social Systems**

I. Introduction

This keynote introduces the use of Search-Based Software Engineering (SBSE) to tackle some of the most pressing ethical requirements posed by modern software systems. For example, in automated decision-making systems and social systems, fairness and safety properties become prominent concerns, whereas the dramatic increase in CO2 emission due to Information and Communication Technologies (ICT) generates growing concerns for software sustainability [1].

While the main requirement for more traditional software systems is to provide the user with the right functionalities (i.e., implement the functional requirements correctly), realising modern and more complex software systems involves ensuring that a software is designed, implemented, and deployed in a way that takes into account the impact it has on users, society, and the environment. This calls for more comprehensive engineering practices for developing software systems with a focus on ethical, social, and environmental considerations, dubbed as *responsible software engineering*.

However, realising these type of systems often involves finding the best or most effective solution among a vast number of possible alternatives and one cannot expect for a software engineer, even the most expert, to be able to manually find all possible opportunities. On the other hand, SBSE provides a framework to systematically explore and evaluate these alternatives, allowing engineers to identify optimal or near-optimal solutions [2].

In this keynote, we will show how SBSE can be a flexible and powerful mean to produce multiple variants of a software system empowering engineers and decision-makers to make informed choices that balance conflicting objectives and that align with their goals and priorities for achieving responsible software. We will present some recent results from my group and from others, and discuss future directions towards realising greener, fairer and safer software systems.

II. What is SBSE?

SBSE involves the application of search-based optimisation techniques to various aspects of the software engineering process. It combines principles from software engineering and optimization algorithms to automate and improve software tasks which are usually characterised by a large solution space and competing goals. These solutions can be represented as a set of variables or parameters that can be optimised to obtain the desired goal(s). The search algorithms are then used to explore this search space to find the best possible solution, guided by an objective function that measures the quality of each solution.

The benefits of SBSE include the potential for the automation, optimization of software engineering tasks, and the ability to handle complex problems which would be difficult to solve manually. SBSE has been successfully applied to various software engineering tasks [2], including but not limited to software project management [3]–[5], software defect prediction [6], software testing [7], automated program repair [8]. Overall, search-based software engineering has been an active area of research and has shown to have the potential to revolutionize various aspects of software development by providing automated and optimized solutions to challenging software engineering problems.

III. SBSE for Responsible Software Engineering

In this keynote, we argue that SBSE can play a fundamental role in achieving responsible software systems. In the following, we describe some possible areas of applications.

A. Software Footprint

The footprint caused by the usage of communication and computing technologies has been expanding globally in the last decade, with the rapid and ever growing adoption of ubiquitous mobile devices and Artificial Intelligence (AI)-enabled systems for everyday life activities [1], [9]. Software contributes to energy consumption on par of other attributes as hardware, data and network. However, software engineers were rarely trained to consider the energy consumption of the systems they create [1]. Even if engineers are nowadays more aware of the energy needs of their software, it remains difficult for them to manually cater for such a challenging non-functional requirement [10], [11]. Previous work has shown that evolutionary computing is well-suited to help developers automatically improve non-functional performance properties of software such as energy and memory consumption. Specifically, Genetic Improvement (GI) [12] is a field of SBSE that studies how to use evolutionary computing to automatically modify the source code to improve software non-functional properties. Although GI techniques for non-functional properties exist for traditional software, these are not as prevalent in other application domains such as mobile computing [11] and AI [13]. We argue that SBSE (specifically GI) can be exploited to improve the footprint of modern software systems. In fact, GI is able to automatically create multiple versions of a same software providing the same set of functionalities but yielding different energy consumption profiles. Moreover, the multi-objective nature of evolutionary computation allows for stakeholders to choose the desired trade-off between quality and efficiency.

B. Software Fairness

Decisions in several domains are increasingly assisted or taken by automated software systems mainly relying on AI components. Important ethical implications arise when such decision systems are used in sensitive contexts (e.g., healthcare, justice, loans) and several biased decisions have been found in existing systems. For example, biased automated decision systems were used to allocate fewer Black patients (who were equally sick as White individuals), to healthcare programs in US hospitals [14], or to assign a higher risk of recidivism to Black people than White people under the same conditions [15].

Therefore, algorithmic fairness has emerged as a crucial requirement to guarantee that such automated decision-making software systems do not discriminate against specific individuals or entire groups, especially minorities [16], [17]. Nevertheless, software fairness requires more than a fair prediction model. Such a model needs to be fair while maintaining a satisfactory accuracy. In fact, has been proved that designing systems to improve for fairness often decrease their accuracy. This phenomenon is known as the *fairness-accuracy trade-off* [18], [19]. Moreover it needs to be used by practitioners in real-world environments, where the notion of fairness may differ [20]–[23].

We argue that SBSE can aid to strike an optimal fairness-accuracy trade-off as well as to bridge the gap between the two roles of *prediction-modeler* (usually taken by data scientists, engineers, or computer scientists) and the *decision-maker* (usually taken by product managers, business strategists, doctors, depending on the application domain).

Chakraborty et al. [24] and Hort et al. [25] were the first to propose the use of multi-objective search to simultaneously optimise for fairness and accuracy/correctness of machine learning and word-embedding models, respectively. Perera et al. [26] showed that using search-based fairness testing outperforms existing fairness testing approaches. Hort et al. [27] showed that search-based approaches can be used to automatically repair fairness and accuracy in decision-making software.

Moreover, SBSE can enable the decision-maker to explicitly specify fairness requirements and constraints, as well as support the job of the prediction-modeler by automating the development and testing of several alternative optimal solutions based on the given requirements, from which the decision-maker can choose from.

SBSE can also help engineers tackle a more challenging notion of fairness, called intersectional fairness, which encompasses multiple sensitive attributes, such as race and gender, simultaneously [28], [29].

C. Safety In On-line Social Systems

Social system uptake has reached the point where they have become crucial for interpersonal communication, business-to-customer communication, and government-to-citizen communication. They have become a driver for innovation, a vehicle for businesses to reach their customers, and a way for families and friends to stay connected. Nevertheless, a number of malicious users can create harm by misusing well-intentioned software platforms as a tool to attack innocent users.

Keeping users safe in on-line social systems has a multi objective character [30]. Ideally one would like to automatically improve existing social systems so that they prevent malicious users, while maintaining or enhancing the user experience for innocent users. This is a challenging problem, where the two objectives are conflicted with each other.

SBSE techniques have been repeatedly proven to be ideal for tackling enormous problem spaces with tight constraints and complex feature interactions, and we envision that they can be used to optimsise social systems to maximise the potential for good while simultaneously minimising the risk of harm. We conjecture that SBSE can provide a single unified way to tackle all forms of online harm including, scamming, spamming, bullying, harassment, hate speech, misinformation, and grooming. Each harm type requires its own set of fitness functions to guide the optimisation of the social system by reducing that particular form of harm, but all such fitness functions fit into the overall general framework of search based optimisation.

BIOGRAPHY

Federica Sarro is a Professor of Software Engineering in the Department of Computer Science at UCL, where she is the Head of the Software Systems Engineering group and where she has established the SOLAR team within CREST. Prof. Sarro has extensive academic and industrial expertise in Search-Based Software Engineering, Empirical Software Engineering and Software Analytics, with a focus on automated software management, optimisation, testing and repair. On these topics she has published over 100 peer-reviewed scholarly articles, and given several invited talks at academic and industrial international events. She has also worked in collaboration with several companies including Meta, Google and Microsoft. Prof. Sarro has obtained numerous awards and generous funding for her research. In 2021, she received the Rising Star Award by the IEEE Technical Community on Software Engineering in recognition of her "excellence in Software Engineering research with scholarly and real-world impact".

Acknowledgment

This work is supported by the ERC grant no. 741278. I would like to thank the members of my group at UCL and the many international colleagues who I have had the pleasure to collaborate with. A special thanks to the Requirements Engineering community, from which I always learn so much. *To our dearest and deeply missed Günther Ruhe, a pioneer of search-based requirements engineering.*

References

[1] A. Fonseca, R. Kazman, and P. Lago, "A manifesto for energy-aware software," *IEEE Software*, vol. 36, no. 6, pp. 79–82, 2019.
[2] M. Harman, S. A. Mansouri, and Y. Zhang, "Search-based software engineering: Trends, techniques and applications," *ACM Comput. Surv.*, vol. 45, no. 1, dec 2012.
[3] F. Ferrucci, M. Harman, and F. Sarro, *Search-Based Software Project Management*. Springer Berlin Heidelberg, 2014, pp. 373–399.
[4] F. Sarro, F. Ferrucci, M. Harman, A. Manna, and J. Ren, "Adaptive multi-objective evolutionary algorithms for overtime planning in software projects," *IEEE Trans. Softw. Eng. (TSE)*, vol. 43, no. 10, pp. 898–917, 2017.
[5] F. Sarro, A. Petrozziello, and M. Harman, "Multi-objective software effort estimation," in *Procs. of the 38th ICSE*, 2016, pp. 619–630.
[6] R. Moussa, G. Guizzo, and F. Sarro, "Meg: Multi-objective ensemble generation for software defect prediction," in *Procs. of the 16th ACM/IEEE ESEM*, 2022.
[7] S. Ali, L. C. Briand, H. Hemmati, and R. K. Panesar-Walawege, "A systematic review of the application and empirical investigation of search-based test case generation," *IEEE Trans. Softw. Eng.*, vol. 36, no. 6, pp. 742–762, 2010.
[8] L. Gazzola, D. Micucci, and L. Mariani, "Automatic software repair: A survey," *IEEE Trans. Softw. Eng.*, vol. 45, no. 1, pp. 34–67, 2019.
[9] R. Schwartz, J. Dodge, N. A. Smith, and O. Etzioni, "Green ai," *Commun. ACM*, vol. 63, no. 12, p. 54–63, Nov. 2020.
[10] M. Hort, M. Kechagia, F. Sarro, and M. Harman, "A survey of performance optimization for mobile applications," *IEEE Trans. Softw. Eng. (TSE)*, 2021.
[11] J. Callan, O. Krauss, J. Petke, and F. Sarro, "How do android developers improve non-functional properties of software?" *Empir. Softw. Eng.*, vol. 27, no. 5, p. 113, 2022.
[12] J. Petke, S. O. Haraldsson, M. Harman, W. B. Langdon, D. R. White, and J. R. Woodward, "Genetic improvement of software: A comprehensive survey," *IEEE Trans. on Evol. Comp.*, vol. 22, no. 3, pp. 415–432, 2018.
[13] S. Georgiou, M. Kechagia, T. Sharma, F. Sarro, and Y. Zou, "Green ai: Do deep learning frameworks have different costs?" in *Procs. of the 44th ICSE*. ACM, 2022, p. 1082–1094.
[14] Z. Obermeyer, B. Powers, C. Vogeli, and S. Mullainathan, "Dissecting racial bias in an algorithm used to manage the health of populations," *Science*, vol. 366, no. 6464, pp. 447–453, 2019.
[15] "How we analyzed the compas recidivism algorithm," May 2016. [Online]. Available: https://www.propublica.org/article/how-we-analyzed-the-compas-recidivism-algorithm
[16] Z. Chen, J. M. Zhang, M. Hort, F. Sarro, and M. Harman, "Fairness testing: A comprehensive survey and analysis of trends," 2022.
[17] M. Hort, Z. Chen, J. M. Zhang, F. Sarro, and M. Harman, "Bias mitigation for machine learning classifiers: A comprehensive survey," 2022.
[18] M. Hort, J. Zhang, F. Sarro, and M. Harman, "Fairea: A model behaviour mutation approach to benchmarking bias mitigation methods," in *Procs. of the 29th ACM ESEC/FSE*, 2021.
[19] Z. Chen, J. M. Zhang, F. Sarro, and M. Harman, "A comprehensive empirical study of bias mitigation methods for machine learning classifiers," *ACM Trans. Softw. Eng. Methodol.*, vol. 32, no. 4, may 2023.
[20] K. Makhlouf, S. Zhioua, and C. Palamidessi, "Machine learning fairness notions: Bridging the gap with real-world applications," *Information Processing & Management*, vol. 58, no. 5, p. 102642, 2021.
[21] L. Baresi, C. Criscuolo, and C. Ghezzi, "Understanding fairness requirements for ml-based software," in *Procs. of the 31st IEEE RE*. IEEE, 2023.
[22] M. Hort and F. Sarro, "Privileged and unprivileged groups: An empirical study on the impact of the age attribute on fairness," in *Procs. of the ACM FairWare'22 (ICSE companion)*. ACM, 2022.
[23] ——, "Did you do your homework? raising awareness on software fairness and discrimination," in *Procs. of the 36th IEEE/ACM ASE*. IEEE, 2021.
[24] J. Chakraborty, S. Majumder, Z. Yu, and T. Menzies, "Fairway: A way to build fair ml software," in *Procs. of the 28th ACM ESEC/FSE*. ACM, 2020, p. 654–665.
[25] M. Hort, R. Moussa, and F. Sarro, "Multi-objective search for gender-fair and semantically correct word embeddings," in *Applied Soft Computing Journal*. IEEE, 2022.
[26] A. Perera, A. Aleti, C. Tantithamthavorn, J. Jiarpakdee, B. Turhan, L. Kuhn, and K. Walker, "Search-based fairness testing for regression-based machine learning systems," *Empir. Softw. Eng.*, vol. 27, no. 3.
[27] M. Hort, J. Zhang, F. Sarro, and M. Harman, "Search-based automatic repair for fairness and accuracy in decision-making software," *Empir. Softw. Eng.*, 2023.
[28] U. Gohar and L. Cheng, "A survey on intersectional fairness in machine learning: Notions, mitigation, and challenges," 2023.
[29] Z. Chen, J. M. Zhang, F. Sarro, and M. Harman, "Fairness improvement with multiple protected attributes: How far are we," 2023.
[30] J. Ahlgren, M. E. Berezin, K. Bojarczuk, E. Dulskyte, I. Dvortsova, J. George, N. Gucevska, M. Harman, R. Lämmel, E. Meijer, S. Sapora, and J. Spahr-Summers, "Wes: Agent-based user interaction simulation on real infrastructure," in *Procs. of the 42nd IEEE/ACM ICSEW*. ACM, 2020, p. 276–284.