International Workshop on Language Models and Programming Languages (LMPL) 2025

CLAIRE LE GOUES, Carnegie Mellon University, USA
SERGEY MECHTAEV, Peking University, China
QINGKAI SHI, Nanjing University, China
CHENGPENG WANG, Purdue University, USA
HE YE, University College London, UK
ZHUO ZHANG, Purdue University, USA

ABSTRACT

Generative artificial intelligence, exemplified by large language models (LLMs), is reshaping various aspects of programming languages, significantly simplifying many stages of software development. In recent years, many traditional areas of the programming languages (PL) community, such as program analysis and verification, program synthesis, and compiler optimization, have been profoundly influenced by LLMs, creating new opportunities for advancement. Despite their rapid progress, the inherent limitations of LLMs, particularly context length constraints, hallucinations, and high inference costs, restrict their effective application to PL tasks and other real-world problems. As two distinct computational paradigms, neural networks represented by LLMs and traditional computation models based on programming languages have unique strengths and characteristics. To encourage researchers from the PL and ML communities to collaboratively address these challenges, we propose hosting the International Workshop on Language Models and Programming Languages (LMPL) at SPLASH 2025, alongside other co-hosted PL conferences and workshops. We envision this new workshop as a platform for researchers from diverse backgrounds to engage in insightful exchanges, fostering synergy between traditional PL techniques and cutting-edge generative AI research, and driving innovation in these fields.

1 MOTIVATION AND OBJECTIVE

The rapid evolution of generative AI has profoundly influenced programming practices. Large Language Models (LLMs) have become pivotal tools extensively used in various programming language tasks [1], encompassing static analysis [2–4], program synthesis [1, 5, 6], and program testing [7–11]. According to our survey [12], there is a consistent increase in the volume of studies related to LLMs within PL conferences over the past three years. Simultaneously, industry companies are directing significant resources towards enhancing PL techniques in industrial contexts through the integration of LLMs. For instance, in early 2024, Meta released a LLM tailored for compiler optimization and program decompilation [13]. Additionally, PL researchers are actively exploring the fusion of traditional PL techniques with LLMs to bolster LLM applications, including prompt engineering [14, 15] and diverse agent designs [16].

Despite the recent strides in employing LLMs for PL tasks and other applications, inherent limitations of LLMs pose substantial hurdles to the widespread adoption of existing methods. First, the context length constraint directly restricts the scalability of programs that can be processed,

Authors' addresses: Claire Le Goues, School of Computer Science, Carnegie Mellon University, USA, clegoues@cs.cmu.edu; Sergey Mechtaev, School of Computer Science, Peking University, China, mechtaev@pku.edu.cn; Qingkai Shi, State Key Laboratory for Novel Software Technology, Nanjing University, China, qingkaishi@nju.edu.cn; Chengpeng Wang, Department of Computer Science, Purdue University, USA, wang6590@purdue.edu; He Ye, School of Computer Science, University College London, UK, hey@cs.cmu.edu; Zhuo Zhang, Department of Computer Science, Purdue University, USA, zhan3299@purdue.edu.

hindering the provision of end-to-end solutions for PL tasks like program optimization [13, 17] and bug detection [18]. Second, the inherent tendency of LLMs to generate erroneous outputs, known as hallucination [19], often leads to subtle errors, such as outputs deviating from expected formats or containing information inconsistent with inputs or common knowledge, thereby undermining performance in practical settings [20, 21]. Third, despite continual enhancements to LLMs, the inference costs associated with the latest models have increased significantly, posing challenges in supporting downstream users reliant on LLM-based solutions. For instance, OpenAI o3, the latest reasoning model, incurs inference costs that can reach thousands of dollars per task [22].

In order to stimulate broader interest from researchers across various disciplines, we advocate for the establishment of a workshop aimed at facilitating interdisciplinary collaboration. This endeavor aims to unite researchers, promoting a deeper integration of programming language techniques and LLMs to effectively address core issues within the programming languages and machine learning communities.

Goals: The workshop aims to achieve the following goals:

- Facilitate discussions among researchers on the primary challenges of LLM-driven solutions for PL problems and other LLM-driven applications.
- Provide a platform for researchers to exchange novel ideas and preliminary findings at the intersection of programming language techniques and language models.
- Establish a forum for academia and industry professionals to bridge the gap between academic research and industry requirements, encouraging the practical application of programming language methodologies and language models to tackle real-world challenges.

2 FORMAT AND REQUIRED SERVICES

Our workshop gives high importance on encouraging participation from both academia and industry. In this section, we will first describe the planned format of the workshop and the preliminary schedule. Then, we will include the logistics and the required services of the workshop.

2.1 Plans of Generating Discussions

Our workshop serves as a platform for researchers in the programming language community to discuss new ideas and share emerging results on the synergy of language models and program language techniques. The authors of accepted papers will give presentations on their research, and we will organize one keynote after the opening session.

For the keynote for LMPL 2025, we plan to invite several well-known researcher from the program language or machine learning community to discuss the new advancement in the intersection of language models and traditional program language techniques, such as compilers, theorem proving, program verification, and program synthesis.

2.2 Scope of Submissions

We invite submissions that discuss recent advancement in the intersection of language models and program languages. The workshop will provide an opportunity for researchers interested in the applications of language models for PL problems and also program repair to exchange ideas and discuss emerging research directions in the field. Topics of interest include, but are not limited to:

LLMs for PL tasks.

- LLMs for static analysis, such as program verification, bug detection, program optimization
- LLMs for code generation, such as program transpilation, synthesis, and repair
- LLMs for program testing, such as fuzzing and domain-specific system testing
- Other tasks in the fields of programming languages and software engineering

PL techniques for LLM applications.

- PL techniques for prompt engineering
- PL techniques for agent design
- PL techniques for model training
- PL techniques for hallucination mitigation
- Other aspects where PL techniques can contribute to LLM applications

Benchmarks and Empirical Studies

- New benchmarks for specific PL tasks and empirical studies of existing LLM-driven PL techniques.
- Empirical studies of existing benchmarks, such as the works summarizing or criticizing existing benchmarks.
- Empirical studies of explainable AI in PL tasks, such as proposing and investigating a specific hypothesis.

2.3 Workshop Duration

Our workshop will be one day long. We prefer the workshop to be allocated on the day preceding the main conferences of OOPLSA and ICFP. To ease the management of multiple workshops within the same day, we plan to adjust the workshop schedule inline with the lunchtime and the coffee breaks of other workshops.

2.4 Workshop Format

For LMPL 2025, we will follow a similar schedule to other workshops where we had several paper presentation sessions, and a discussion session. In order to encourage participants to share their latest research at the workshop, we welcome the following three formats of submissions:

- **Research paper:** Similar to research papers presented at various conferences, these should include a well-designed methodology and experimental measurements;
- **Position paper:** Presenting look-forwarding viewpoints or showcasing ideas that have not been thoroughly evaluated through experiments;
- **Talk paper:** Sharing one or more works that have already been published in other venues.

We anticipate at least 8 research papers, 4 position papers, and 4 talk papers accepted. We will divide all accepted submissions into several paper sessions based on their topics. Each session will have a mix of research papers, position papers, and talk papers. Table 1 shows a preliminary schedule of the workshop. Organizers and participants will chair the sessions. We will determine the length of each talk based on the number of accepted submissions.

2.5 Logistics

For a face-to-face workshop, we request a projector and a projector screen. Participants will be able to use their own laptops. We will ask in advance for PPT or PDF copies and bring our own laptop to serve as a backup. We would also like to request a whiteboard and markers for workshop participants. To support presenters who are unable to attend the conference in person due to uncontrollable factors, we are considering setting up a remote presentation mode. We plan to work closely with the SPLASH publicity chairs and student volunteers to arrange for online presentations.

3 TARGET AUDIENCE

We plan to hold an open workshop, where any attendee can register and attend if one is interested. We plan to promote the workshop through relevant mailing lists (e.g., SEWORLD) and social media

Table 1. The preliminary schedule of PLML 2025

Time Slot	Event
9:00 - 9:10	Welcome session
9:10 - 10:00	Keynote
10:00 - 10:20	Tea break
10:20 - 11:50	Paper session
11:50 - 13:00	Lunch break
13:00 - 13:30	Discussion
13:30 - 15:00	Paper session
15:00 - 15:20	Tea break
15:20 - 16:50	Paper session
16:50 - 17:00	Closing

platforms (e.g., posting tweets on X). We expect to have around 40–60 workshop participants. Due to the close connection between the organizers and the industry, including experience working full-time in the industry, we expect to attract participants from diverse backgrounds in both industry and academia. As the previous and current institutions cover different continents, including North America, Europe, and Asia, we will solicit workshop attendees from different continents and countries to ensure diversity among the attendees. We believe that the workshop should be attractive to junior researchers and students looking for important and emerging research directions in the intersection of language models and programming languages. We also hope that practitioners from the industry can raise more needs and problems from real-world scenarios, thus encouraging practitioners from academia to create real impacts by solving these real-world problems.

4 PROCEEDINGS

Submissions should use ACM SIGPLAN Conference acmart Format, sigplan sub-format and 10 point font. All submissions should be in PDF format. We expect to have at least 16 submissions of non-talk papers. We anticipate 10–12 to be research papers, while the rest will be position papers. Research papers will be in the range of 8–10 pages (not including references), while position papers are expected to be 2–4 pages. Besides, a talk paper should offer the following two kinds of information in one page: (1) The abstract of the talk; (2) the references of introduced papers. In principle, there is no limit on the number of works to be introduced, but due to time limits for the talk, it is recommended not to exceed 3 papers. Please note that research papers will be included in the proceedings of SPLASH 2025 and follow the SPLASH publication guidelines, while the authors of position papers can choose whether to publish their position papers in the proceedings. All the talk papers will not be included in the proceedings.

Review Process. We will use HotCRP as the online submission system because (1) we would like to be consistent with the review process of OOPLSA and other co-hosted conferences / workshops in SPLASH 2025, and (2) several organizers have used HotCRP for other previous workshops. Our selection of PC members aims to follow the principles of diversity and inclusion. We plan to derive the tentative list of PC members by collecting the researchers who have published papers in programming language venues (e.g., PLDI, POPL, OOPSLA, ICFP, ECOOP) and have also contributed to machine learning publications in the past five years. For the review process, we will follow the same review process as other conferences where each paper receives at least 3 reviews and the double-blind reviewing model is employed. The paper acceptance will depend on the quality of the submission and the significance of the contribution. The proceeding chair will be He Ye, who will be responsible for sending the list of accepted papers to ACM.

REFERENCES

[1] Shraddha Barke, Michael B James, and Nadia Polikarpova. Grounded copilot: How programmers interact with code-generating models. *Proceedings of the ACM on Programming Languages*, 7(OOPSLA1):85–111, 2023.

[2] Qian Chen, Chenyang Yu, Ruyan Liu, Chi Zhang, Yu Wang, Ke Wang, Ting Su, and Linzhang Wang. Evaluating the effectiveness of deep learning models for foundational program analysis tasks. *Proceedings of the ACM on Programming Languages*, 8(OOPSLA1):500–528, 2024.

[3] Aashish Yadavally, Yi Li, Shaohua Wang, and Tien N Nguyen. A learning-based approach to static program slicing. *Proceedings of the ACM on Programming Languages*, 8(OOPSLA1):83–109, 2024.

[4] Haonan Li, Yu Hao, Yizhuo Zhai, and Zhiyun Qian. Enhancing static analysis for practical bug detection: An llm-integrated approach. *Proceedings of the ACM on Programming Languages*, 8(OOPSLA1):474–499, 2024.

[5] Jialu Zhang, José Pablo Cambronero, Sumit Gulwani, Vu Le, Ruzica Piskac, Gustavo Soares, and Gust Verbruggen. Pydex: Repairing bugs in introductory python assignments using llms. *Proceedings of the ACM on Programming Languages*, 8(OOPSLA1):1100–1124, 2024.

[6] Qiaochu Chen, Arko Banerjee, Çağatay Demiralp, Greg Durrett, and Işıl Dillig. Data extraction via semantic regular expression synthesis. *Proceedings of the ACM on Programming Languages*, 7(OOPSLA2):1848–1877, 2023.

[7] Chunqiu Steven Xia, Matteo Paltenghi, Jia Le Tian, Michael Pradel, and Lingming Zhang. Fuzz4all: Universal fuzzing with large language models. In *Proceedings of the IEEE/ACM 46th International Conference on Software Engineering*, pages 1–13, 2024.

[8] Gabriel Ryan, Siddhartha Jain, Mingyue Shang, Shiqi Wang, Xiaofei Ma, Murali Krishna Ramanathan, and Baishakhi Ray. Code-aware prompting: A study of coverage-guided test generation in regression setting using llm. *Proceedings of the ACM on Software Engineering*, 1(FSE):951–971, 2024.

[9] Caroline Lemieux, Jeevana Priya Inala, Shuvendu K Lahiri, and Siddhartha Sen. Codamosa: Escaping coverage plateaus in test generation with pre-trained large language models. In *2023 IEEE/ACM 45th International Conference on Software Engineering (ICSE)*, pages 919–931. IEEE, 2023.

[10] Xianfei Ou, Cong Li, Yanyan Jiang, and Chang Xu. The mutators reloaded: Fuzzing compilers with large language model generated mutation operators. ASPLOS, 2024.

[11] Chenyuan Yang, Yinlin Deng, Runyu Lu, Jiayi Yao, Jiawei Liu, Reyhaneh Jabbarvand, and Lingming Zhang. Whitefox: White-box compiler fuzzing empowered by large language models. *Proceedings of the ACM on Programming Languages*, 8(OOPSLA2):709–735, 2024.

[12] PurCL. A continuously updated collection of codellm papers, 2024. URL https://github.com/PurCL/CodeLLMPaper.

[13] Chris Cummins, Volker Seeker, Dejan Grubisic, Baptiste Roziere, Jonas Gehring, Gabriel Synnaeve, and Hugh Leather. Meta large language model compiler: Foundation models of compiler optimization. *arXiv preprint arXiv:2407.02524*, 2024.

[14] Luca Beurer-Kellner, Marc Fischer, and Martin Vechev. Prompting is programming: A query language for large language models. *Proceedings of the ACM on Programming Languages*, 7(PLDI):1946–1969, 2023.

[15] Ningke Li, Yuekang Li, Yi Liu, Ling Shi, Kailong Wang, and Haoyu Wang. Drowzee: Metamorphic testing for fact-conflicting hallucination detection in large language models. *Proceedings of the ACM on Programming Languages*, 8 (OOPSLA2):1843–1872, 2024.

[16] Ziyang Li, Jiani Huang, and Mayur Naik. Scallop: A language for neurosymbolic programming. *Proceedings of the ACM on Programming Languages*, 7(PLDI):1463–1487, 2023.

[17] Chris Cummins, Bram Wasti, Jiadong Guo, Brandon Cui, Jason Ansel, Sahir Gomez, Somya Jain, Jia Liu, Olivier Teytaud, Benoit Steiner, et al. Compilergym: Robust, performant compiler optimization environments for ai research. In *2022 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*, pages 92–105. IEEE, 2022.

[18] Chongzhou Fang, Ning Miao, Shaurya Srivastav, Jialin Liu, Ruoyu Zhang, Ruijie Fang, Ryan Tsang, Najmeh Nazari, Han Wang, Houman Homayoun, et al. Large language models for code analysis: Do {LLMs} really do their job? In *33rd USENIX Security Symposium (USENIX Security 24)*, pages 829–846, 2024.

[19] Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 2023.

[20] Chengpeng Wang, Wuqi Zhang, Zian Su, Xiangzhe Xu, and Xiangyu Zhang. Sanitizing large language models in bug detection with data-flow. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 3790–3805, 2024.

[21] Niels Mündler, Jingxuan He, Slobodan Jenko, and Martin Vechev. Self-contradictory hallucinations of large language models: Evaluation, detection and mitigation. In *The Twelfth International Conference on Learning Representations*.

[22] Sherwyn. O3 model compute cost seems high - chatgpt - openai developer forum, 2024. URL https://community.openai.com/t/o3-model-compute-cost-seems-high/1066145.