Автоматизация в промышленности 2021. №6
www.avtprom.ru

DOI: 10.25728/avtprom.2021.06.12

В.А. Петухов (Университет ИТМО)

Генерация кода для тестирования компиляторов с использованием генеративно-состязательных сетей

Рассматривается задача генерации кода на произвольном языке программирования, анализируются существующие достижения в этой области, их достоинства и недостатки. Выделяется ряд проблем, которые остаются актуальными, в частности, перекладывание на алгоритмы машинного обучения правил языка (синтаксиса и некоторых правил семантики) вместо их формального описания в архитектуре модели. Для решения обозначенных проблем предлагается применить алгоритмы генеративно-состязательных сетей.

Ключевые слова: машинное обучение, генерация кода, генеративно-состязательные сети, тестирование компиляторов.

Петухов Виктор Алексеевич *– аспирант факультета «Информационных технологий и программирования» Университет ИТМО.*

.

Список литературы

1. Neelakantan, K., et al. The Role of Compilers in Computer-System Performance // Current Science, vol. 60, no. 11, 1991, pp. 650–652
2. Xuejun Yang, Yang Chen, Eric Eide, and John Regehr. Finding and understanding bugs in C compilers // 32th ACM SIGPLAN Conference on Programming Language Design and Implementation. 2011. Pp. 283–294.
3. Junjie Chen, Jibesh Patra, Michael Pradel, Yingfei Xiong, Hongyu Zhang, Dan Hao, and Lu Zhang. A Survey of Compiler Testing // ACM Comput. Surv. 53, 1, Article 4 (May 2020), 36 pages
4. Andrew W. Appel. Semantics-directed code generation // 12th ACM SIGACT-SIGPLAN symposium on Principles of programming languages. 1985. Pp. 315–324.
5. Химонин Ю. И. Сбор и анализ требований к программному продукту. http://mastefanov.com 6. A.S. Boujarwah, K. Saleh. Compiler test case generation methods: a survey and assessment // Information and Software Technology. Vol. 39. Is. 9. 1997. Pp. 617-625
6. A.S. Boujarwah, K. Saleh. Compiler test case generation methods: a survey and assessment // Information and Software Technology. Vol. 39. Is. 9. 1997. Pp. 617-625
7. Chen, Junjie & Hu, Wenxiang & Hao, Dan & Xiong, Yingfei & Zhang, Hongyu & Zhang, Lu & Xie, Bing. An empirical comparison of compiler testing techniques. 2016. pp. 180-190.
8. Rohan Padhye, Caroline Lemieux, Koushik Sen, Mike Papadakis, and Yves Le Traon. Semantic fuzzing with zest // 28th ACM SIGSOFT International Symposium on Software Testing and Analysis. 2019. Pp. 329–340
9. Godefroid, Patrice & Kiezun, Adam & Levin, Michael. Grammar-based Whitebox Fuzzing // Proceedings of the

ACM SIGPLAN Conference on Programming Language Design and Implementation (PLDI). 2008. No 43. P. 206-215.
10. Chris Cummins, Pavlos Petoumenos, Alastair Murray, and Hugh Leather. Compiler fuzzing through deep learning // 27th ACM SIGSOFT International Symposium on Software Testing and Analysis. 2018. pp. 95–105
11. X. Liu, X. Kong, L. Liu and K. Chiang. TreeGAN: Syntax-Aware Sequence Generation with Generative Adversarial Networks // IEEE International Conference on Data Mining (ICDM). 2018. pp. 1140-11459. Brin S., Page L.The Anatomy of a Large-Scale Hypertextual Web Search Engine // URL: http://infolab.stanford.edu/pub/papers/google.pdf

Petukhov V.A. Compiler testing code generation using generative adversarial networks

The problem of code generation in an arbitrary programming language is discussed, the present-day results in this field, their merits and drawbacks are examined. Urgent problems are identified, such as transposing language rules (syntaxis and some semantic rules) into machine learning algorithms instead of their formal description in the model architecture. Generative adversarial network algorithms are proposed for rsolving these problems.

Keywords: machine learning, code generation, generative adversarial networks, compiler testing.