COMP-604 Written Report:
Program Repair as Tensor Completion

Breandan Considine
Advisors: Jin Guo, Xujie Si

April 12, 2023

Short summary: We introduce a mathematical framework for parsing, code completion and program repair based on multilinear algebra over finite fields, unifying probabilistic programming and formal language theory.

Medium summary: Our work is motivated by the observation that many problems which arise in programming can be cast as matrix multiplication with semiring algebras. For example, code completion can be viewed as a fixpoint search over the space of syntactically admissible edits. In our work, we introduce a new formalism for context-free parsing, code completion, and error correction based on multilinear algebra over finite fields which can be used to parse, complete code and repair broken syntax fragments in nearly optimal time. Our approach scales to a variety of practical program repair scenarios, allowing users to flexibly incorporate domain-specific constraints.

Long summary: Over the last few years, there has been a surge of progress in applying language models to write programs. That work is primarily based on methods from differential calculus and continuous optimization, leading to the so-called *naturalness hypothesis*, which suggests programming languages are not so different from natural ones. In contrast, programming language theory takes the view that languages are essentially discrete and finitely-generated sets, although perhaps uncomputably large ones, governed by logical calculi. Programming, thus viewed, is more like a mathematical exercise in constraint satisfaction, an oft-overlooked but unavoidable aspect of translating abstract ideas into computing machinery. These constraints naturally arise at various stages of syntax validation, type-checking and run-time verification, and help to ensure programs fulfill their intended purpose.

Our work shows these two approaches are not mutually exclusive, but complementary and reciprocal. Borrowing analysis techniques from multilinear algebra and tensor completion in the machine learning setting, we develop an equational theory that allows us to translate various decision problems on formal languages into a system of inequalities over finite fields. As a practical consequence, this means we can efficiently encode a number of problems in parsing, code completion and program repair using SAT solvers, which are known to be highly efficient, scalable and flexible to domain-specific constraints. We demonstrate the effectiveness of our approach in a variety of practical scenarios, including code completion and program repair for linear conjunctive languages, and show that our approach is competitive with state-of-the-art methods in terms of both accuracy and efficiency.

Furthermore, we implement our framework in a new developer tool called Tidyparse, which allows users to define grammars and incorporate domain-specific constraints during code completion and program repair. Tidyparse accepts a linear conjunctive language and a string. If the string is valid, it returns the parse forest, otherwise, it returns a set of repairs, ordered by their Levenshtein edit distance to the invalid string. Tidyparse compiles the grammar and candidate string onto a discrete dynamical system using an extended version of Valiant's algorithm and solves for its fixedpoints using an incremental SAT solver. By allowing the string to contain holes, repairs may contain either concrete tokens or nonterminals, which can be expanded by the user or a neural-guided search procedure to produce a valid program. This approach to parsing has many advantages, enabling us to repair syntax errors, correct typos and recover from errors in real time, as well as being provably sound and complete with respect to the grammatical specification.

This work is in collaboration with Jin Guo, Xujie Si and Nghi Bui. We would like to thank Qirun Zhang, Brigitte Pientka, Semyon Grigorev, Jürgen Cito, Michael Schröder, Ori Roth, Younesse Kaddar, and Kiran Gopinathan for their invaluable comments and feedback during its development.