Interpretable Program Synthesis for Text Transformations

Supervisor: Prof Alan Blackwell (afb21@cam.ac.uk)

The “FlashFill” feature in Microsoft Excel is the best known commercial example of program synthesis - using machine learning methods to automatically create simple programs that transform arbitrary input strings to output strings. However, the behaviour of FlashFill is a black box - although it sometimes works exactly as expected, when it doesn’t work, the user has to resort to trial and error, providing different sets of input/output pairs until the required functionality emerges. Even when the initial results appear to be as expected, a program synthesised by trial and error is very likely to contain bugs or produce unexpected behaviours. This project will build on previous research in the Rainbow Group that created a novel visualisation to specify transformations via underlying algorithms from the research behind FlashFill. The goal of this research will be to produce interpretable and editable program synthesis methods for text transformations.

Gulwani, S., Hernández-Orallo, J., Kitzelmann, E., Muggleton, S. H., Schmid, U., & Zorn, B. (2015). Inductive programming meets the real world. Communications of the ACM, 58(11), 90-99.

Gorinova, M.I., Blackwell, A.F., Sarkar, A. and Prince, K. (2016). Transforming spreadsheets with Data Noodles. In Proceedings of IEEE Visual Languages and Human-Centric Computing (VL/HCC) 2016.

Validated Design Process for No-Code/Low-Code DSLs

Supervisor: Prof Alan Blackwell (afb21@cam.ac.uk)

Domain-Specific Programming Languages (DSLs) are often created for specific application communities, usually with a syntax layer and application-specific semantics implemented on top of existing metaprogramming techniques. With rapidly increasing commercial interest in no-code / low-code development platforms, there is a serious methodology gap in replicable processes for the design of new DSLs that draw on a range of interaction and visualisation methods. The goal of this project is to develop a rigorous design method that builds on human factors frameworks such as Green's Cognitive Dimensions of Notations, Moody's Physics of Notation and Blackwell and Fincher's Patterns of User Experience, integrating human factors insights within an engineering process. In addition to systematic analysis of this human factors literature, the project will include validation of the derived method with candidate DSL designers from several different application areas, using evaluation interviews and case study analysis.

Green, T. R. G., & Petre, M. (1996). Usability analysis of visual programming environments: a 'cognitive dimensions' framework. Journal of Visual Languages & Computing, 7(2), 131-174.

Moody, D. (2009). The "physics" of notations: toward a scientific basis for constructing visual notations in software engineering. IEEE Transactions on software engineering, 35(6), 756-779.

Blackwell, A.F. & Fincher, S. (2010). PUX: Patterns of User Experience. interactions 17(2), 27-31.

Quantifying the World Population of Programmers

Supervisor: Prof Alan Blackwell (afb21@cam.ac.uk)

How many programmers are there in the world? This seems like a simple question, but presents a serious opportunity for research. In previous decades, major advances in End-User Programming research were introduced by empirical studies which quantified the level of demand for programming tools that were accessible outside of the educational and professional software development contexts. Classic papers by Boehm at al in 1995 (since cited 949 times) and Scaffidi et al in 2005 (since cited 429 times) have not been updated for the past 15 years. In that time, there have been radical changes in the landscape of digital services, meaning that it is now urgent to direct future research with better understanding of who, how, and why people have to carry out programming tasks. This project will involve identifying appropriate data sources and creating data mining and statistical tools to develop an evidence base for the actual distribution of programming, extending beyond the commercial and office environments that were the focus of work by Boehm and Scaffidi, to include social media applications and IoT environments in the home and industry.

Boehm, B., et al. Cost Models for Future Software Life Cycle Processes: COCOMO 2.0. Annals of Software Engineering Special Volume on Software Process and Product Measurement, J.C. Baltzer AG Science Publishers, Amsterdam, The Netherlands, 1995.

C. Scaffidi, M. Shaw and B. Myers, "Estimating the numbers of end users and end user programmers," 2005 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC'05), Dallas, TX, USA, 2005, pp. 207-214, doi: 10.1109/VLHCC.2005.34.