Modelling Robustness with Conjunctive Grammars

(work under development)

Vadim Zaytsev, vadim@grammarware.net

Software Analysis & Transformation Team (SWAT),
Centrum Wiskunde & Informatica (CWI), The Netherlands

Grammars in a broad sense [2] are used to specify structural commitment in software systems: concrete syntax definitions determine how source code is turned into a parse tree; library interfaces define the signatures of the functions exposed and used by third parties; document schemata fix the structure of the XML documents that are considered valid; type definitions influence assertions on the variable values and function bodies; etc. In the context of software evolution, one frequently faces a challenge of consistency enforcement and change propagation: when the source code is updated, its structural commitments must be reevaluated — commonly by coevolving a corresponding grammar.

There exist numerous techniques to address robustness issues: tolerant [4], agile [1] and island [5] variations of parsing; negotiated [8] form of grammar transformation; notation-parametric [9] heuristics of grammar recovery; and many others. These techniques can be considered to share one important property — they specify two kinds of structural commitment at the same time: a precise one and a tolerant one. On one extreme, a precise commitment relation is unknown or missing, so tolerance is the only way to ensure robustness. On the other extreme, a tolerant kind of commitment is never needed. In this presentation, it is proposed to formally specify such double commitments with conjunction [6] — an operation commonly found in set theory, but much more rare in grammarware. This approach may be seen as distantly related to quasi-synchronous grammars [7] known in machine translation.

By using a conjunctive grammar to specify both precise and tolerant structural commitments, we create a setup where one entity specifies many ways of jeopardising or weakening of existing contracts when the base software evolves. For instance, when parsing, any failure of a conjunctive clause of a particular nonterminal can be noted and reported, but does not necessarily prevent delivering a parse tree to the next tool in the pipeline (such as a fact extractor).

During the presentation, there will be a demonstration showing conjunctive grammars to be useful for specifying derived grammars in the islands-and-lakes paradigm, which is a fuzzy generalisation of context-free grammars that allows insignificant fragments of "water" to be recognised along the detailed "islands" for the sake of performance or robustness. We will use Rascal [3] language workbench, and all the code will be made publicly available through the Software Language Processing Suite repository [10].

References

1. T. R. Dean, J. R. Cordy, A. J. Malton, and K. A. Schneider. Agile Parsing in TXL. *Journal of Automated Software Engineering*, 10(4):311–336, 2003.
2. P. Klint, R. Lämmel, and C. Verhoef. Toward an Engineering Discipline for Grammarware. *ACM Transactions on Software Engineering Methodology (ToSEM)*, 14(3):331–380, 2005.
3. P. Klint, T. van der Storm, and J. Vinju. EASY Meta-programming with Rascal. In J. M. Fernandes, R. Lämmel, J. Visser, and J. Saraiva, editors, *Post-proceedings of GTTSE 2009*, volume 6491 of *LNCS*, pages 222–289, Berlin, Heidelberg, January 2011. Springer-Verlag.
4. S. Klusener and R. Lämmel. Deriving Tolerant Grammars from a Base-line Grammar. In *Proceedings of ICSM 2003*, pages 179–188, Los Alamitos, CA, USA, Sept. 2003. IEEE Computer Society.
5. L. Moonen. Generating Robust Parsers using Island Grammars. In *Proceedings of WCRE 2001*, pages 13–22. IEEE Computer Society Press, Oct. 2001.
6. A. Okhotin. Conjunctive Grammars. *Journal of Automata, Languages and Combinatorics*, 6(4):519–535, 2001. http://users.utu.fi/aleokh/papers/conjunctive.pdf.
7. D. A. Smith and J. Eisner. Quasi-Synchronous Grammars: Alignment by Soft Projection of Syntactic Dependencies. In *Proceedings of the Workshop on Statistical Machine Translation*, StatMT '06, pages 23–30, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.
8. V. Zaytsev. Negotiated Grammar Transformation. In J. De Lara, D. Di Ruscio, and A. Pierantonio, editors, *Post-proceedings of the Extreme Modeling Workshop (XM 2012)*. ACM Digital Library, Nov. 2012. In print. An extended version submitted to the Journal of Object Technology (JOT).
9. V. Zaytsev. Notation-Parametric Grammar Recovery. In A. Sloane and S. Andova, editors, *Post-proceedings of the 12th International Workshop on Language Descriptions, Tools, and Applications (LDTA 2012)*. ACM Digital Library, June 2012.
10. V. Zaytsev, R. Lämmel, T. van der Storm, L. Renggli, and G. Wachsmuth. Software Language Processing Suite[1], 2008–2013. http://slps.github.io.

[1] The authors are given according to the list of contributors at http://github.com/grammarware/slps/graphs/contributors.