Approach to Learning Production Rules for Grammar-Based Functional Design

Julian R. Eichhoff, Dieter Roller

Institute of Computer-aided Product Development Systems
University of Stuttgart
Universitätsstr. 38
70569 Stuttgart, Germany
julian.eichhoff@informatik.uni-stuttgart.de
dieter.roller@informatik.uni-stuttgart.de

Abstract: This paper outlines the problem of efficiently formalizing functional design rationale in form of graph-rewriting production rules. It is argued that genetic programming provides a feasible machine learning approach to this.

1 Introduction

According to the VDI 2221 design guideline functional design is a crucial step in conceptual product design [VDI93]. Its goal is to define the purpose of the product in terms of input-output-relations on forms of energy, materials and signals. These so called functions are composed in a graph-based model termed function structure. Graph grammars and graph-rewriting systems in general have been used to automatically derive fully evolved function structures from simplified black-box models, e.g. [SC04]. These systems rely on a set of production rules for modifying function structures. Iterative application of rules results in step-wise model changes, where some sequences may lead to the desired results. However, since rules are formal representations of human design rationale their elicitation can become costly. E.g. [SC04] put a great effort in analyzing commonalities among function structures to acquire their rule set. In order to address this drawback we propose a machine learning (ML) approach to the acquisition of rules that grounds on the following hypothesis: Production rules can be efficiently learned from sample function structures using genetic programming (GP).

2 Counter Argument

One may argue that production rules cannot be acquired by ML since they resemble human design rationale that can be only accessed asking knowledgeable experts. Nonetheless, suppose sample data is accessible, i.e. there are pairs of black boxes and resulting function structures. Then hypotheses – here in the form of rule sets – can be formulated on how black boxes were transformed into function structures given that the space of possible hypotheses is known. ML methods can be used to automatically search this space for hypotheses that best describe the sample data [Sa11].

3 Support for Learning Rules with Genetic Programming

Quite similar to our goal [Ma06] used GP to discover relationships among the set of design parameters and objectives. [YSA10] employed a genetic algorithm to find rules and rule sequences for growing shapes, whereas [OCB08] took a partial component analysis approach to discover shape grammar rules. Further support is also given by preliminary successes in the implementation of a prototype system. In this system a rule is represented by a tree consisting of elementary graph operations (find/add/link functions, etc.). The GP framework maintains a set of alternatives for the next best rule. This so called population is iteratively refined using the evolutionary principles of selection, mutation and recombination [Ko92]. To assess the fitness of rule candidates we consider the validity and applicability of rules as well as the validity of function structures produced by the current rule set and their similarity with sample function structures. First experiments show that simple rules can be learned with this prototype.

4 Conclusion

Using GP to learn production rules from sample function structures is feasible since: The suggested method has been used before in order to explicate design rationale from sample designs; other works specifically addressed the task of learning production rules; and proof of concept is given by a prototype implementation. In our future work we will extend the prototype to learn complex rules and complete rule sets.

References

[VDI93] Verein Deutscher Ingenieure: Methodik zum Entwickeln und Konstruieren technischer Systeme und Produkte (VDI 2221). Beuth, Berlin, 1993.

[SC04] Sridharan, P.; Campbell, M. I.: A Grammar for Function Structures. In Proc. of the ASME 2004 IDETC/CIE Conferences, Salt Lake City, 2004; pp. 41-55.

[Sa11] Sammut, C.: Learning as Search. In (Sammut, C.; Webb, G.I. Eds.): Encyclopedia of Machine Learning. Springer, New York, 2011; pp. 572-576.

[Ma06] Matthews, P.C.; Standingford, D.W.F.; Holden, C.M.E.; Wallace, K.M.: Learning inexpensive parametric design models using an augmented genetic programming technique, AIEDAM 20(1), 2006; pp. 1-18.

[YSA10] Yogev, O.; Shapiro, A.; Antonsson, E. K.: Computational Evolutionary Embryogeny, IEEE T. Evolut. Comput. 14(2), 2010; pp. 301-325.

[OCB08] Orsborn, S.; Cagan, J.; Boatwright, P.: A methodology for creating a statistically derived shape grammar composed of non-obvious shape chunks, Res. Eng. Des. 18(4), 2008; pp. 181-196.

[Ko92] Koza, J. R.: Genetic Programming: On the Programming of Computers by Means of Natural Selection, MIT Press, Cambridge, 1992.