Automated Software Remodularization Based on Move Refactoring - A Complex Systems Approach

Ingo Scholtes, Marcelo Serrano Zanetti, Claudio Juan Tessone, Frank Schweitzer

Chair of Systems Design
ETH Zürich
CH-8092 Zürich, Switzerland
ischoltes@ethz.ch

Abstract: Modular design is a desirable characteristic that fosters the comprehensibility and thus maintainability of software systems. While many software systems are initially created in a modular way, over time modularity typically degrades. In our work, we propose an automated strategy to remodularize software based on move refactorings, i.e., moving classes between packages without changing other aspects of the source code. Taking a complex systems perspective, our approach applies network theory to the dynamics of software dependency structures. Drawing inspiration from statistical physics, we use the Potts Spin Model and turn it into a stochastic remodularization algorithm which is based on probabilistically moving classes between modules. We test our method on 39 open source JAVA software projects. Comparing the modular structure produced by developers with that optimized by our algorithm, we find that our method is able to improve modularity by an average of 166 ± 77 percent. Our work highlights the potential of interdisciplinary applications of methods from the statistical physics perspective on complex systems to software engineering.

The modular design of software systems is an important factor that contributes to the success of software engineering projects. It is enabled by a set of design principles, among which information hiding and separation of concerns are the most influential ones [PCW85]. Their application limits coordination needs and fosters the simple replacement of components, thus facilitating sustainable software engineering processes. Many software systems are initially created in a modular way. However, over time modularity often degrades as components are reused outside the context where they were created. Our work [ZTSS14] proposes an automated strategy to remodularize software based on move refactorings, i.e., the simplest possible refactoring which moves classes between packages without changing other aspects of the code. Our goal is to automatically identify move refactorings that result in a large improvement of modularity. For this, we first define a measure capturing the modular characteristic of software structures. Focusing on object-oriented programming concepts, we take a network perspective on software structures in which nodes represent classes, and links represent class dependencies such as method or field invocation, instantiation, etc. We use the Q-modularity metric which was originally developed for cluster detection in networks, and which was recently adapted to monitor the modularity of software structures [ZS12]. Drawing inspiration from physics, we then use the Potts Spin Model [Wu82] and turn it into a probabilistic remodularization algorithm. It is based on the assumption that an optimum modular design minimizes the coupling.
between modules, while the \textit{cohesion} within modules is maximized. The algorithm probabilistically moves classes between adjacent modules, the probability of moving class $X$ nonlinearly depending on both the number and module memberships of classes depending on $X$. Similar to simulated annealing, this yields a stochastic optimization algorithm which reaches a global optimization of modularity by a simple and stochastic local rule.

We validate our method using a data set, that captures the evolution of 39 Java projects. Comparing the code generated by developers with the optimized code resulting from our approach, we find that modularity improves on average by $166 \pm 77\%$. Our method is highly efficient, requiring only a small number of iterations to reach an equilibrium configuration which can be used to identify optimum move refactoring candidates. An ECLIPSE plugin implementing our method is available online\textsuperscript{1}. An illustration of the outcome of our method is shown in Figure 1. In summary, we reinterpret the $Q$-modularity measure from complex network theory as quantitative indicator for the modularity of software structures. We further highlight relations between move refactorings and the Potts model studied in complex systems, thus demonstrating interdisciplinary applications of methods from the statistical physics perspective on complex systems to software engineering.

References


\footnote{\url{http://sourceforge.net/projects/somomoto/}}