Journal of King Saud University – Computer and Information Sciences (2017) **29**, 349–364

King Saud University

Journal of King Saud University – Computer and Information Sciences

www.ksu.edu.sa
www.sciencedirect.com

Improving package structure of object-oriented software using multi-objective optimization and weighted class connections

Amarjeet *, **Jitender Kumar Chhabra**

Department of Computer Engineering, NIT Kurukshetra, Haryana, India

Received 13 February 2015; revised 21 July 2015; accepted 3 September 2015
Available online 2 November 2015

KEYWORDS

Multi-objective optimization;
Package restructuring;
Modularization;
Weighting scheme;
Maintenance

Abstract The software maintenance activities performed without following the original design decisions about the package structure usually deteriorate the quality of software modularization, leading to decay of the quality of the system. One of the main reasons for such structural deterioration is inappropriate grouping of source code classes in software packages. To improve such grouping/modular-structure, previous researchers formulated the software remodularization problem as an optimization problem and solved it using search-based meta-heuristic techniques. These optimization approaches aimed at improving the quality metrics values of the structure without considering the original package design decisions, often resulting into a totally new software modularization. The entirely changed software modularization becomes costly to realize as well as difficult to understand for the developers/maintainers. To alleviate this issue, we propose a multi-objective optimization approach to improve the modularization quality of an object-oriented system with minimum possible movement of classes between existing packages of original software modularization. The optimization is performed using NSGA-II, a widely-accepted multi-objective evolutionary algorithm. In order to ensure minimum modification of original package structure, a new approach of computing class relations using weighted strengths has been proposed here. The weights of relations among different classes are computed on the basis of the original package structure. A new objective function has been formulated using these weighted class relations. This objective function drives the optimization process toward better modularization quality simultaneously ensuring preservation of original structure. To evaluate the results of the proposed approach, a series of experiments are conducted over four real-worlds and two random software applications. The experimental results clearly indicate the effectiveness of our approach in

* Corresponding author.
E-mail addresses: amarjeetnitkkr@gmail.com (Amarjeet), jitenderchhabra@nitkkr.ac.in (J.K. Chhabra).
Peer review under responsibility of King Saud University.

Production and hosting by Elsevier

http://dx.doi.org/10.1016/j.jksuci.2015.09.004

improving the modularization quality of existing package structure by doing very small movement of classes between packages of original software modularization.

1. Introduction

The quality of the software modularization has a major impact on many software system quality parameters such as understandability and maintainability (Tonella, 2001; Praditwong et al., 2011). For Object-oriented software systems, the modularization is largely dependent on classes. Classes are nothing but collection of data and associated methods. Generally for small software system, the classes are considered as modules, however for large and complex systems, it has been reported that a set of collaborating classes (i.e. packages) can be a better module of system organization than a class (Gupta and Chhabra, 2009) and bigger software systems are generally designed and developed by using these modularization criteria.

As the software system evolves over the time, addition, removal and modification of classes influence the original software modularization in an adverse manner. It has been observed that due to short deadlines, developers often do not follow the original package design rules during maintenance for early completion of the work leading to modular structure deterioration (Marcio et al., 2014). As a result, the original modularization structure gets modified to the extent that it looses its structural quality due to sub-optimal placement of classes in different packages (Gui and Scott, 2006). Software maintenance is a continuous ongoing phenomenon but the deteriorated structure quality makes the maintenance difficult and negatively affects the software evolution (Mitchell and Mancoridis, 2006). Hence, the re-modularization of a software system becomes essential whenever the system quality gets degraded to a point from where the further evolution is not feasible within permissible time and cost. The software remodularization problem has been solved by many researchers in the past by formulating it as a search-based optimization problem and solved it using meta-heuristic techniques (Mitchell, 2002; Mahadavi et al., 2003; Patel et al., 2009; Abdeen et al., 2009, 2013; Cui and Chae, 2011; Praditwong et al., 2011; Barros, 2012).

Most of the search-based remodularization approaches improve the software structure by optimizing coupling and cohesion criteria (Mahadavi et al., 2003; Praditwong et al., 2011; Barros, 2012). These approaches improved the coupling and cohesion in absolute terms, but the newly suggested package modularization solution usually became so complex and different from the original one, that it would be hardly acceptable to the software maintainer (Marcio et al., 2014). Such methods can be useful when system requires complete overhauling. Such a situation comes once in a while, but not during initial/regular maintenance. At initial/regular maintenance, system needs to be re-modularized with an improved modular structure with less restructuring cost. Hence, the quality criteria need to be modeled in such a way, so that it can drive optimization process with preservation of the original package modularization. In literature, some researchers (Abdeen et al., 2009, 2013; Bavota et al., 2014) have tried to address these problems by controlling the optimization process using some constraints. The authors (Abdeen et al., 2009, 2013) improved the package structure by improving the package coupling, package cohesion and package cyclic dependencies as a single and multi-objective optimization problem. They controlled the optimization process by applying the constraints on movement of the classes among packages. However, defining such constraints is not easy and maintainers must have a deep insight of the original design decisions of software modularization. In most of the cases the maintainers are not the developers of the original modularization (Bavota et al., 2014). In such situations finding the constraints that can drive the optimization process toward design decision of original modularization becomes very difficult. Instead of optimizing through constraints, another approach has been proposed recently by Bavota et al. (2014) where the involvement of end-user becomes compulsory. Their approach is based on structural and semantic dependencies. They controlled the remodularization process by putting the software end-users in the every iteration for requesting the feedback. In this method an end-user must have thorough understanding of design decision of original software modularization, which is again not feasible practically most of the times. Hence it can be stated that importance of original structure of the software plays an important role in the process of remodularization, but existing methods of its inclusion in the optimization process are highly person-specific, and availability of such persons is always a limitation of such approaches. So there is an immense need to incorporate the characteristics of original modular-structure, preferably without necessity of individuals having clear insight of original design. This paper attempts to solve this issue and the proposed approach is able to include the original structure characteristics from the source code, without any need of well-aware end-user/original developer.

This paper presents a multi-objective optimization approach for improving the existing package structure of an object-oriented system aiming at preserving the original design decision of software modularization. To this contribution, the optimization process is controlled by objective functions which are formulated in terms of newly proposed weighted relations that reflect the nature of original design decision. The weights of each type of existing relations are calculated in terms of locality (intra and inter relations) of that relation in original software modularization. Further, these weighted relations are used to calculate overall connection strength among pair of classes. This connection strength helps in keeping the optimization process around original software modularization.

The multi-objective formulation includes package cohesiveness index, package connectedness index, intra-package connection density and package size index as the objective functions. To solve the multi-objective optimization, we used Non-Dominated Sorting Genetic Algorithm (NSGA II) (Deb et al., 2002) a widely-accepted multi-objective evolutionary algorithm. We considered this algorithm, in particular, because