A Source Code Change Impact Analysis Algorithm for Iterative Software Development

Jerod W. Wilkerson, Department of Information Systems, Sam and Irene Black School of Business, Sam and Irene Black School of Business, Penn State Erie, Erie, PA, USA

ABSTRACT

CHA-AS is a source code change impact analysis algorithm for Java programs. CHA-AS differs from other algorithms in that it does not require the program versions it compares to be whole programs with a well-defined program entry point. The need for such an algorithm is evident in iterative software development projects and projects involving the development of code libraries and frameworks—all of which may not have a well-defined program entry point at the time when change impact analysis needs to be performed. The CHA-AS algorithm supports the development of Decision Support Systems for software development managers and programmers working on iterative software development projects, or projects to develop source code libraries and frameworks. This paper describes the CHA-AS algorithm and demonstrates it to be efficient and effective in calculating source code change impact.

Keywords: Algorithm, Change Impact Analysis, Change Impact Analysis Algorithm, Iterative Software Development, Source Code Analysis, Static Analysis

INTRODUCTION

Software change impact analysis has been defined as “identifying the potential consequences of a change, or estimating what needs to be modified to accomplish a change” (Bohner & Arnold, 1996). Source code change impact analysis involves the identification of source code changes and the “ripple effects” (Stevens, Meyers, & Constantine, 1974) of these changes on other unchanged code. For example, methods that invoke or are invoked by a changed method may be impacted by the changed method, and these impacted methods may in turn impact any methods they invoke or are invoked by.

Several source code, change impact analysis algorithms have been developed to identify the impact of a set of source code edits, but most rely on programs having a well-defined program entry point. For example, the RTA algorithm (Bacon, 1997) uses information about which classes have been instantiated between the program entry point and each method call site, to reduce the potential targets of method
calls. Although fully-functional software programs can reasonably be expected to have such a program entry point, other software cannot. Software being developed using iterative software development methods, such as Test-Driven Development (TDD) (Beck, 2002), may not have a well-defined program entry point at the time when change impact analysis needs to be performed; and code libraries and software development frameworks may never have such an entry point.

Source code change impact analysis algorithms may be used to form the foundation of Decision Support Systems that provide critical information to software development managers and programmers. The most obvious use of these change impact based Decision Support Systems is to obtain information about how project schedules and budgets will be affected by software change requests. This information can be used as input for project scheduling, feasibility analysis, and client negotiation decisions. These Decision Support Systems can also be used by software developers to evaluate different alternatives for implementing change requests in existing software. Other decisions that these systems can support include determining which if any unit tests need to be re-executed after development of a new iteration (Parashar, Bhatia, & Kalia, 2011; Ren, Shah, Tip, Ryder, & Chesley, 2004), which code needs to be re-analyzed when using automated test data generation (DeMillo & Offutt, 1991) on iterative projects, and which software mutants need to be generated and executed when using mutation analysis (DeMillo, Krauser, & Mathur, 1991) on iterative projects.

This paper promotes the development of change impact based Decision Support Systems by describing and evaluating the Class Hierarchy Analysis, Arbitrary Start (CHA-AS) change impact analysis algorithm. CHA-AS is a source code change impact analysis algorithm for Java programs, which can identify the impact of a set of source code changes in situations where a clearly defined program entry point does not exist at the time when change impact analysis needs to be performed. A prior version of the algorithm was presented at the 2010 Annual Meeting of the Decision Sciences Institute (Wilkerson, 2010).

RELATED RESEARCH

Software change impact analysis can be divided into two broad categories: traceability analysis and dependency analysis. Traceability analysis is concerned with the ability to trace relationships between different software artifacts—such as the ability to trace requirements to the source code that implements them.

Dependency analysis examines detailed relationships between source code level entities such as variables, modules, functions, and methods. Dependency analysis is focused on code-level relationships that are more specific than the broad relationships analyzed by traceability analysis. Dependency analysis is the type of change impact analysis performed by the CHA-AS algorithm. Two broad categories of dependency analysis include data dependency and control dependency (Bohner & Arnold, 1996). Data dependency analysis involves the analysis of program statements that define or modify data (variables or constants) in a program, and the other parts of the program affected by these statements. Control dependency analysis consists of the statements in a program controlling the flow of execution (such as function and method calls) and the parts of the program affected by these statements. Most applications of dependency analysis consist of the following three steps for both data and control dependencies:

1. Identify Program Changes.
2. Create Call Graph.
3. Use Call Graph to Identify Change Impact.

The relevant literature relating to the first two steps is described below. The use of the call graph to identify the change impact varies depending on the problem domain, and is the main contribution of this research.
Predicting Impulsive Buyers: A Comparative Study of Binary Classifiers’ Discriminative Ability
www.igi-global.com/article/predicting-impulsive-buyers/163960?camid=4v1a

Identifying Crucial Know-How and Knowing-That for Medical Decision Support
www.igi-global.com/article/identifying-crucial-know-how-and-knowing-that-for-medical-decision-support/164439?camid=4v1a