Abstract

Method for detecting resource conflicts in multithreaded applications

Damian GIEBAS, M. Sc. Eng.

Almost each and every one of contemporarily utilised processors supports the technology of concurrency. Thanks to said technology, programmers have attained the capability of simultaneous task execution in the scope of one application. In order to enable this technology, it was required to extend the existent programming languages with proper libraries, thanks to which programmers may use the aforementioned technology. As a result, a substantial number of languages, with C included, support the creation of multithreaded applications, even despite the fact that these languages were not designed to do so. The side effect of introducing concurrency is the emergence of a new kind of errors. The errors stem from the fact that a programmer may create structures resulting in unpredicted interactions between logical units called threads. These structures cause the rise of costs involved in software development, therefore rendering the creation of multithreaded applications unprofitable. In such a situation, the answer to the following question is sought: Is given multithreaded application written in C with pthread library free from structural errors leading to resource conflicts, such as race conditions, deadlocks, atomicity violations and order violations.

Numerous methods and tools have been developed so that the localization of errors leading to resource conflicts is more attainable. A significant number of these solutions supports the localization of one to three types of resource conflicts, therefore the usage of these tools does not allow one to answer the hereabove question. The original assumptions such as the usage of simple models are among the main reasons for the fact that these tools are not agile enough to be utilised to localize four types of resource conflicts in multithreaded applications written in C with pthread library. Another important factor is the inability of any of the tools to be used online because of the time needed for the localization of the errors extending to several hours. It denotes the lack of solutions supporting C programmers in the process of development of multithreaded applications in a way enabling swift source code analysis targeted at localizing the aforementioned structural errors leading to resource conflicts.

The analysis of literature in the subject matter showed that in order to achieve the objective of creation of a method capable of localization of resource conflict in online mode, one is required to use the method of static analysis. For that end, it was essential to design a model enabling the designation of conditions not only allowing to localize errors leading to resource conflicts but also decreasing the cost of the usage of static analysis method. The designed model of multithreaded application source code oriented at localization of four types of resource conflicts assumes the utilisation of C language and pthread library in online mode. Said model enables source code analysis on a structural and temporal level.

In this dissertation the problem of localization of errors in multithreaded applications is discussed. The literature review enabled the clarification of assumptions in the area of the basis of the source code model and error localization conditions. The design of the method, its implementation in a prototype solution and conducting experiments in order that the method is properly verified were thus rendered possible.