Platform Independent Analysis of Probabilities on Multithreaded Programs

Yuting Chen, School of Software, Shanghai Jiao Tong University, Shanghai, China & Shanghai Key Laboratory of Computer Software Testing & Evaluating, Shanghai, China

ABSTRACT

A concurrent program is intuitively associated with probability: the executions of the program can produce nondeterministic execution program paths due to the interleavings of threads, whereas some paths can always be executed more frequently than the others. An exploration of the probabilities on the execution paths is expected to provide engineers or compilers with support in helping, either at coding phase or at compile time, to optimize some hottest paths. However, it is not easy to take a static analysis of the probabilities on a concurrent program in that the scheduling of threads of a concurrent program usually depends on the operating system and hardware (e.g., processor) on which the program is executed, which may be vary from machine to machine. In this paper the authors propose a platform independent approach, called ProbPP, to analyzing probabilities on the execution paths of the multithreaded programs. The main idea of ProbPP is to calculate the probabilities on the basis of two kinds of probabilities: Primitive Dependent Probabilities (PDPs) representing the control dependent probabilities among the program statements and Thread Execution Probabilities (TEPs) representing the probabilities of threads being scheduled to execute. The authors have also conducted two preliminary experiments to evaluate the effectiveness and performance of ProbPP, and the experimental results show that ProbPP can provide engineers with acceptable accuracy.

Keywords: Control Dependent Probability, Execution Path, Scheduling, Thread Execution Probability, Threads

INTRODUCTION

A concurrent program with nondeterminism is intuitively associated with probability. A concurrent program can be nondeterministic in that races may exist in the program, each of which usually arises when separate threads are scheduled to execute sequentially on a single processor. When it happens, the execution steps of threads can be interleaved. Thus using an input to repeatedly execute the program can produce various program paths, each of which represents a sequence of program instructions executed.

A weak assumption is that the frequencies of all potential paths are uniformly distributed. For example, in the sample program shown in Figure 1, the main thread spawns the threads

DOI: 10.4018/ijsi.2013070104
Let $t_1$ and $t_2$. Note that a data race exists in the program in the sense that after the program is executed, $x$ can either be 6 or 4, and $y$ can either be 2 or 16. Let the two threads $t_1$ and $t_2$ be executed in an interleaving manner, and each program statement be executed in an atomic manner. Six potential orders of the statements can be enumerated to represent six execution paths, each of which is denoted by a sequence of pairs. A pair, say $(t, s)$, is composed of a thread $t$ and a statement $s$.

Let $n$ be the total number of program paths in the program, and $\text{Prob}(path_i)$ be the probability of a path $path_{0 \leq i \leq n-1}$. The weak assumption denotes that the frequencies of all execution paths are uniformly distributed, i.e.,

$$\text{Prob}(path_i) = \frac{1}{n}.$$  

In this example, we have $\text{Prob}(path_{0 \leq i \leq 5}) = 0.167$. However, the assumption is usually not true in practice. Running of the program for a number of times can inevitably find a fact that some program paths can be executed more frequently than the others.
Related Content

Evaluating Performance of Software Architecture Models with the Palladio Component Model
www.igi-global.com/chapter/evaluating-performance-software-architecture-models/29436?camid=4v1a

Model-Based Testing of Embedded Systems Exemplified for the Automotive Domain
www.igi-global.com/chapter/model-based-testing-embedded-systems/36350?camid=4v1a
Acoustic Modeling of Speech Signal using Artificial Neural Network: A Review of Techniques and Current Trends
www.igi-global.com/chapter/acoustic-modeling-of-speech-signal-using-artificial-neural-network/135890?camid=4v1a

Classification of Bug Injected and Fixed Changes Using a Text Discriminator
www.igi-global.com/article/classification-of-bug-injected-and-fixed-changes-using-a-text-discriminator/121547?camid=4v1a