Applied Soft Computing xxx (2015) xxx–xxx

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

A meta-heuristic solution for automated refutation of complex software systems specified through graph transformations

Vahid Rafe [a,*], Maryam Moradi [a], Rosa Yousefian [a,b], Amin Nikanjam [c]

[a] Department of Computer Engineering, Faculty of Engineering, Arak University, Arak, 38156-8-8349, Iran
[b] Sama Technical and Vocational Training College, Islamic Azad University, Khomeinishahr Branch, Esfahan, Iran
[c] Faculty of Computer Engineering, K.N.Toosi University of Technology, Tehran, 1631714191, Iran

ARTICLE INFO

Article history:
Received 21 August 2014
Received in revised form 9 April 2015
Accepted 13 April 2015
Available online xxx

Keywords:
Model checking
Refutation
PSO
GSA
Graph transformation system
State space explosion

ABSTRACT

One of the best approaches for verifying software systems (especially safety critical systems) is the model checking in which all reachable states are generated from an initial state. All of these states are searched for errors or desirable patterns. However, the drawback for many real and complex systems is the state space explosion in which model checking cannot generate all the possible states. In this situation, designers can use refutation to check refusing a property rather than proving it. In refutation, it is very important to handle the state space for finding errors efficiently. In this paper, we propose an efficient solution to implement refutation in complex systems modeled by graph transformation. Since meta-heuristic algorithms are efficient solutions for searching in the problems with very large state spaces, we use them to find errors (e.g., deadlocks) in systems which cannot be verified through existing model checking approaches due to the state space explosion. To do so, we employ a Particle Swarm Optimization (PSO) algorithm to consider only a subset of states (called population) in each step of the algorithm. To increase the accuracy, we propose a hybrid algorithm using PSO and Gravitational Search Algorithm (GSA). The proposed approach is implemented in GROOVE, a toolset for designing and model checking graph transformation systems. The experiments show improved results in terms of accuracy, speed and memory usage in comparison with other existing approaches.

1. Introduction

Nowadays, software development is a complex task due to the size and complexity of the current system's requirements. Hence, errors and bugs are common challenges in the software development. Using appropriate methods to find and solve errors is therefore essential in every developing effort. Early detection of errors may yield better results in terms of cost, marketing time and correctness. Therefore, using model-based techniques which focus on models prior to implementation is a great option and model checking is one of the best methods for verifying software and hardware systems [1]. Using model checking method requires specifying systems with a formal language at first. Then, a model checker generates all reachable states from an initial one and the generated state space is searched to find errors or desirable patterns.

There exist different formal languages with specific capabilities. Graph Transformation System (GTS) is a visual yet formal modeling language which is used to specify different systems naturally and succinctly [2]. GTS is a formalism which is not only used for system specification and verification but also is widely utilized in many software development activities such as model transformation [3], designing architectural styles [4], refinement [5], meta-modeling [6], refactoring [7], workflow modeling and analysis [8] and software architecture performance analysis [9]. We therefore considered GTS as a test bed for implementing our idea. However, our approach is independent of the language and it is also possible to consider other model checking languages and tools to implement it.

Even using a proper formal language like GTS, there is a problem for many real and complex systems called state space explosion in which the model checker cannot generate all the states due to enormous number of states. To resolve this problem, different classical approaches, like symbolic verification [10], partial order reduction

* Corresponding author. Tel.: +98 9183526780.
E-mail addresses: v-rafe@araku.ac.ir, rafe@iust.ac.ir (V. Rafe), m.65moradi@gmail.com (M. Moradi), Rosa8a81@yahoo.com (R. Yousefian), nikanjam@kntu.ac.ir (A. Nikanjam).

http://dx.doi.org/10.1016/j.asoc.2015.04.032
1568-4946/

[11], symmetry checking methods [12] and scenario-driven model checking [13] have been developed. However, almost all of these methods use an exhaustive search on the state space and they may encounter the explosion of state space.

The experiments in recent years suggest that meta-heuristic algorithms are more efficient in finding safety property violations than classical algorithms in very large and complex systems [14]. Meta-heuristic and evolutionary algorithms such as Genetic Algorithm (GA) and Ant Colony Optimization (ACO) have been used in order to cope with the state space explosion problem.

This work is a prosecution of the work presented in [2] in which a heuristic solution was introduced to cope with the state space explosion problem using the GA. Although the GA is very efficient in many optimization problems, especially when dealing with large spaces, it suffers from the low speed of convergence and lack of accuracy in the complex and large systems [14]. Hence, our aim is to improve the previous work [2] by proposing a faster yet more accurate algorithm. The PSO [35] which is a very simple algorithm with few parameters, is one of the most efficient algorithms in artificial intelligence and computational applications. Moreover, this algorithm has an improved convergence speed [15], and can therefore be a suitable alternative. We utilize the PSO algorithm to explore the state space using model checking process. The main idea is to produce and explore only a part of the state space. However, similar to many algorithms, PSO suffers from trapping in the local optima which reduces the accuracy of search. Therefore, to solve this problem, we propose a hybrid approach using GSA which is an efficient local search algorithm to cope with the local optima problem. Some researches show that hybrids of PSO with other meta-heuristic or evolutionary algorithms provide more accurate results when compared with the original PSO itself. Additionally, we also examine different algorithms in our study: Simulated Annealing (SA), cuckoo search and bat optimization. The experimental results show that the hybrid version of PSO with GSA produces more accurate results with a lower computational burden in comparison with the mentioned algorithms. However, the hybrid approach has a slower convergence speed than PSO, but its accuracy is considerably enhanced. We use GROOVE [16,17], a tool for modeling and verifying GTS specifications, to implement our idea. As our approach considers only a part of the state space, it cannot be used to prove a property. It can however be used for software refutation, i.e. it can prove that errors exist by providing a counter example.

This paper is organized as follows: Section 2 surveys the state of the art. Section 3 briefly introduces the required background. In Section 4, we present our proposed approach using the PSO algorithm. We also introduce a hybrid algorithm in this section based on PSO and GSA. In Section 5, the implementation strategy along with the used parameters is presented. In Section 6, the experimental results on different case studies along with a discussion on the observations are presented. Finally, Section 7 concludes the paper and discusses future research.

2. Related works

There are different approaches to cope with the state space explosion problem. In the classical approaches, authors try to reduce the size of the state space [12] or memory usage [18] by some methods such as compositional verification, partial order reduction and symmetry reduction. Various algorithms are used to reduce the memory space required for storing states in the memory saving-based approaches. For example in [19], the authors presented an approach based on the bounded model checking method for GTS via SMT (Satisfiability Modulo Theories) solving. For this purpose, the authors encoded the reachability problem of a forbidden pattern in a GTS as a SMT formula. The property will always be a (forbidden) graph and the aim is to check the reachability of error states. In fact, these approaches do not use any heuristics. Therefore, the problem still exists and from this point of view, our approach is entirely different.

In [20], authors propose approaches with different heuristics such as hamming distance and approximations of the set of states which can lead to the violation of assertions and estimates the distance to error states. Using this approach reduces the number of states required to find the bug. In [21], the explicit state model checker HSF-SPIN is presented based on the model checker SPIN. The A^* algorithm is used and certain heuristics are defined to accelerate the search procedure for finding a specific failure situation. Using this approach, counter examples are found faster and the size of explored part of the state space is usually smaller in comparison with the classical approaches. In another work, an approach to formalize a framework for the application of heuristic search is presented in order to analyze structural properties of systems modeled by GTS [22]. Heuristic search is intended to reduce the analytical effort and deliver shorter solutions and paths in GTS. All of these heuristic approaches are exhaustive search methods. Hence, although these methods help to find errors faster, the state space explosion still exists.

Meta-heuristic techniques are classified as another class of methods which are used to cope with the state space explosion problem in model checking.

Strategies which are based on ACO algorithm, inspired by ant's behavior prefer shorter paths to longer ones in exploring paths to find errors. So, these strategies require less memory to store shorter paths with fewer states. Therefore, in researches based on this method, model checking can respond with optimal memory usage [23,24]. However, accuracy may be an issue. For example, in [25], the authors propose the use of a new kind of ACO, ACOhg, to refute safety properties. ACOhg (ACO for huge graphs) is the improved version of the traditional ACO. In ACOhg, the length of the paths traversed by ants in the construction phase is limited. Additionally, the ants start the path construction from different nodes during the search. They also consider deadlock to check its violation. The authors compare ACOhg with the exact and exhaustive search algorithms such as DFS, BFS and A^*. The authors extend their work to consider the liveness properties [26]. They use ACOhg to find violations of liveness property. Even though we do not consider liveness properties in this paper, however the accuracy of our proposed PSO-GSA approach is considerably improved. Moreover, the authors use a textual language, HSF-SPIN, to implement their ideas.

In another work, using a reinforcement learning agent, the authors propose an approach to optimize memory usage by providing a guided search for finding counter examples [27]. They utilize the notion of fairness to propose a heuristic reward function. Additionally, they use probabilistic model checking concept to provide a termination condition for agent's search as well as to provide an approximation measure for the correctness of the model. The authors compare their approach with random model checking. This approach is limited to LTL properties. Moreover, the results of this approach are not accurate enough (due to the approximation used for the correctness of the model). Also, the speed of this approach is slower than the existing meta-heuristic approaches. This work was implemented on a textual modeling formalism and a model checker called Modere.

Chicano et al. [28] presented a comparison of five meta-heuristic algorithms including SA, ACO, PSO and two variants of GA to solve the problem of finding deadlocks in the concurrent Java programs. Moreover, the authors have used five other classical search algorithms to compare and analyze the results of algorithms. The reported results show that meta-heuristic algorithms are more