Karsten Schmidt

Automated generation of a progress measure for the sweep-line method

Abstract In the context of Petri nets, we propose an automated construction of a progress measure which is an important pre-requisite for a state space reduction technique called the sweep-line method. Our construction is based on linear-algebraic techniques concerning the transition vectors of the Petri net under consideration. We further discuss the possible combination of the sweep-line method with other state space reduction techniques (partial order reduction, the symmetry method).

Keywords Petri nets · Explicit state space verification · Sweep · Line method

1 Introduction

The sweep-line method [1] is a recently proposed reduction technique for explicit state space verification. Using this method, it is possible to verify reachability properties of finite state systems.

In its basic shape, it deletes previously constructed states that cannot serve as successors of states not yet explored. The key concept for this method is a so-called progress measure that assigns values to states which are non-decreasing w.r.t. the successor state relation. The sweep-line method was later generalized such that progress measures can be used which are non-monotonous w.r.t. the successor relation. In that case, states that have a predecessor with larger progress value are stored permanently. Thus, a good non-monotonous progress measure should be designed such that value decrease by transition occurrence happens as seldom as possible. In the original papers, [2, 3], it is left to the user to provide a progress measure, assuming that the user knows about some concept of progress in the modeled system.

We propose an automated generation of a progress measure for the generalized sweep-line method. It works for place/transition Petri nets, where convenient concepts for describing progress measures cannot be found within the formalism itself (in contrast to high-level nets where the language of annotations to the net can be used to formulate progress measures).

Our progress measure is not necessarily monotonous. We derive the measure from an analysis of the system’s transition vectors, and their linear dependencies. We arrive at an incremental progress measure. That is, we can assign to each transition a fixed value such that the progress value of a successor state differs from the original state exactly by the value assigned to the fired transition. One advantage of this kind of progress measure is that, on occurrence of a transition, the progress value of the successor state can be computed by addition of an offset to the progress value of the predecessor, that is, in constant time. Moreover, so-called regress transitions—transitions that decrease the progress value—can be identified statically.

We start with a brief description of the sweep-line method, and continue with some basics about the linear algebra of Petri nets. Then, we present our proposal to the definition of a progress measure. Finally, we discuss the combination of the sweep-line method with well-known state space reduction techniques such as partial order reduction or symmetry reduction.

2 The sweep-line method

First, we sketch the basic sweep-line method [2]. At any snapshot during explicit state space exploration, we can distinguish three kinds of states. We have states already seen, and states not yet seen. Among the states already seen there are those where all enabled transitions have already been explored, and those where some successors have not yet been explored. The last kind of states is called the front.

K. Schmidt (✉)
Institut für Informatik, Humboldt-Universität zu Berlin, Unter den Linden 6, 10099 Berlin, Germany
E-mail: kschmidt@informatik.hu-berlin.de
Assume we assigned a progress value\(^1\) \(p(s)\) to each state \(s\) such that for all transitions \(t, s \xrightarrow{t} ts'\) implies \(p(s) \leq p(s')\). Obviously, all states still to be tested for presence in the state space are (transitive) successors of states in the front and have therefore a progress value greater or equal to the minimum progress value among the front states. Consequently, states with smaller progress values than the minimum progress value appearing in the front can be safely removed from the state space. This is exactly the reduction the sweep-line method aims at. As the front evolves forward, more and more states can be removed, leading to the intuition of a sweep line following the front of state space exploration and removing every state behind it (cf. Fig. 1). For being able to remove as many states as possible, and as early as possible, a search strategy is recommendable where front states are explored in ascending order of their progress values. In contrast, depth-first search is not recommendable as the initial state is the last one to leave the front thus making it impossible for the sweep line to proceed forward.

For the generalized sweep-line method [3], the monotony condition for the progress measure is dropped. Thus, the method works, at least in principle, with any assignment \(p\) of progress values to states. Now, there can be situations where a transition leads to a state with smaller progress value. Such a pair of states is called a regress edge in the state space.

The generalized sweep-line method complements the basic method with the following twist. Whenever a regress edge occurs during a run (as in the basic method), the target state of that edge (the state with smaller progress value) is stored and marked persistent. That is, it will never be removed subsequently. It is, however, not explored immediately. Due to the removal of states behind the sweep line, we cannot be sure whether or not we have already seen that state. Thus, after having finished one state space exploration, we start another state space exploration with all states recently marked persistent as initial front. This exploration possibly re-explores parts of the state space, and can lead to further persistent states that need to be explored subse-

\(^1\) In general, progress values can be members of any partially ordered set. For this paper, however, it is sufficient to view progress values as integer numbers. \(p\) is not necessarily injective.