Abstract. This paper presents a new method for computing reduced representation of vector state spaces consisting of infinitely many states. Petri nets are used as a model for generating vector state spaces, and the state space is represented in the form of semilinear subsets of vectors. By combining the partial order methods with the proposed algorithm, we can compute reduced state spaces which preserve some important properties, such as liveness of each transition and the existence of deadlocks. The state space of a finite capacity system can be viewed as that of an infinite capacity system projected to the states satisfying the capacity condition. We also show that the proposed algorithm is applicable to vector state spaces with finite capacities.

1 Introduction

Largeness of state space is a serious problem in analyzing systems that allow concurrent occurring of events. Even if the size of each subsystem is small, the state space generated by their composition becomes very large since independent events can be interleaved in many possible orders. This problem is called state space explosion.

To avoid this problem, several methods have been proposed, such as utilization of binary decision diagrams (BDDs) [11], symbolic model checking [3, 13], on-the-fly model checking [15], and the partial order methods [4, 5, 16]. Combination of these methods is also studied.

In the partial order methods, the partial order defined on the set of actions is used to compute reduced state spaces which preserves some important properties, such as liveness of each transition and the existence of deadlocks. In this paper, we aim to extend the partial order methods in the following two points.


We will consider the situation that new processes are dynamically generated during the execution. This means that the state space may contain infinitely many states. Such an infinite state space will be described in the form of semilinear subsets of vectors, in which every two states are equivalent to
each other. The equivalence is defined to at least satisfy that all states in each equivalence class have the same set of enabled transitions.

2. Treating the state space of a finite capacity system as an projection of the state space of an infinite capacity system.

Since practical systems do not allow infinitely many resources, we usually give a capacity to each component of the system. The state space of a finite capacity system can be viewed as that of an infinite capacity system projected to the states satisfying the capacity condition. We will apply the state space generation algorithm for infinite capacity systems to finite capacity systems.

We will use Petri nets as a model for describing systems. The state space of a Petri net is represented by a labeled transition system in which every state is a \( k \)-dimensional nonnegative integer vector. We will refer such transition systems as vector transition systems. The vector transition systems generated by Petri nets have a characteristic property, which will be called order-persistence. The proposed algorithm uses this property to obtain a reduced representation of the state space. In addition, the stubborn set method\,[14], which is a kind of the partial order methods, will be combined with the proposed algorithm.

In Section 2, we first show the outline of the proposed algorithm. After that, we will show basic definitions and notations in Section 3. In Petri nets, the coverability tree was proposed to represent the infinite state space by a finite search tree. In Section 4, we will describe the proposed algorithm, comparing with that of the coverability tree. In Section 5, the algorithm will be modified to apply to the finite capacity systems. Discussions on the implementation and experimental results will be shown in Section 6. Some remarks and future work will be described in Section 7.

2 Outline of the Algorithm

Fig. 1 shows an unbounded Petri net, i.e., a Petri net that generates a state space containing infinitely many states. While processes are not explicitly described in Petri nets, process algebras are a model based on the behavior of processes. A CSP program\,[8] equivalent to this net is shown as follows:

\[
\begin{align*}
P &= a \to (b \to P \mid c \to STOP), \\
Q &= (a \to Q_1 \mid d \to Q_1), \\
Q_1 &= (b \to Q_2 \mid e \to Q_2), \\
Q_2 &= f \to (Q||Q), \\
R &= P||Q.
\end{align*}
\]

Process \( Q \) is duplicated after the execution of action \( f \). The state space of this Petri net contains infinitely many states. The coverability tree was proposed for representing such an infinite state space in the form of a finite search tree\,[9, 12]. In order to treating the infinity, the special symbol \( \omega \), representing “infinity”, is introduced. By the depth-first search, the following sequence of states are generated in the coverability tree.