Chapter 6

MEMORY-SAVING ANALYSIS OF PETRI NETS

Andrei Karatkevich

University of Zielona Góra, Institute of Computer Engineering and Electronics, ul. Podgórska 50, 65-246 Zielona Góra, Poland; e-mail: A.Karatkevich@iie.uz.zgora.pl

Abstract: An approach to Petri net analysis by state space construction is presented in the paper, allowing reducing the necessary memory amount by means of removing from memory the information on some of intermediate states. Applicability of the approach to deadlock detection and some other analysis tasks is studied. Besides this, a method of breaking cycles in oriented graphs is described.

Key words: Petri nets; simulation; analysis; formal models.

1. INTRODUCTION

Petri nets\(^1\) are a popular formal model of a concurrent discrete system, widely applied for specifying and verifying control systems, communication protocols, digital devices, and so on. Analysis of such nets is a time- and memory-consuming task, because even a simple net may have a huge number of reachable states caused by its concurrent nature (the so-called state explosion problem\(^2,3\)).

However, state space search remains one of the main approaches to Petri net analysis (deadlock detection, for example). But there are various methods handling state explosion problem, such as lazy state space constructions, building reduced state spaces instead of the complete ones\(^2\). Among such methods, Valmari’s stubborn set method\(^4,5\) is best known.

Theoretically there is no necessity of huge memory amount to solve Petri net analysis problems; there is a polynomial-space algorithm of deadlock detection in a safe Petri net, but it is practically absolutely inapplicable because its time consumption is woeful\(^3\). Generally, the known algorithms solving verification tasks in a relatively small memory are extremely slow\(^3\).
In this paper we concentrate on a less radical approach, reducing memory amount and keeping it, however, exponential in worst case. The approach is based on removing from memory some of the intermediate states. Some results are recalled from Refs. 6 and 7; also, new results are presented.

2. PRELIMINARIES

A Petri net\(^1\) is a triple \(\Sigma = (P, T, F)\), where \(P\) is a set of places, \(T\) is a set of transitions, \(P \cap T = \emptyset\) and \(F \subseteq (P \times T) \cup (T \times P)\). For \(t \in T\), \(\star t\) denotes \(\{p \in P | (p, t) \in F\}\), \(t^*\) denotes \(\{p \in P | (t, p) \in F\}\), and \(\star t\) and \(t^*\) are the sets of input and output places, respectively. \(\forall t \in T : \star t \neq \emptyset, t^* \neq \emptyset\). A similar notation is used for places \((\star p, p^*)\). A Petri net can also be considered as an oriented bipartite graph. A state (marking) of a net is defined as a function \(M : P \rightarrow \{0, 1, 2, \ldots\}\). It can be considered as a number of tokens situated in the net places. \(M(p)\) denotes the number of tokens in place \(p\) at \(M\). \(M' > M\) denotes that \(\forall p \in P : M'(p) \geq M(p)\) and \(\exists p \in P : M'(p) > M(p)\). Initial state \(M_0\) is usually specified.

A transition \(t\) is enabled and can fire if all its input places contain tokens. Transition firing removes one token from each input place and adds one token to each output place, thus changing the current state. If \(t\) is enabled in \(M\) and its firing transforms \(M\) into \(M'\), then that is denoted as \(MtM'\). This denotation and the notion of transition firing can be generalized for firing sequences (sequential firing of the transitions, such that each transition is enabled in the state created by firing of the previous transition). If a firing sequence \(\sigma\) leads from state \(M\) to \(M'\), it is denoted as \(M \sigma M'\). A state that can be reached from \(M\) by a firing sequence is called reachable from \(M\); the set of reachable states is denoted as \([M]\). A transition is live if there is a reachable marking in which it is enabled; otherwise it is dead. A state in which no transitions are enabled is called a deadlock. A net is live if in all the reachable markings, all the transitions of the net are live. A net is safe if in any reachable marking no place contains more than one token. A net is bounded if \(\exists n : \forall p \in P \forall M \in [M_0] \exists M(p) \leq n\) (there is an upper bound of number of tokens for all the net places in all reachable markings).

A reachability graph is a graph \(G = (V, E)\) representing state space of a net. \(V = [M_0]; e = (M, M') \in E \Leftrightarrow MtM'\) (then \(t\) marks \(e\)). The reachability graph is finite if and only if the net is bounded. A strongly connected component (SCC) of a reachability graph is a maximal strongly connected subgraph. A terminal component of a graph \(G\) is its SCC such that each edge which starts in the component also ends in it\(^\text{3,5}\).

A set \(T_S\) of the transitions of a Petri net at state \(M\) is a stubborn set if (1) every disabled transition in \(T_S\) has an empty input place \(p\) such that all