Science of Computer Programming 157 (2018) 41–55

Contents lists available at ScienceDirect

Science of Computer Programming

www.elsevier.com/locate/scico

Pre-synthesis of Petri nets based on prime cycles and distance paths

Eike Best[a,*,1], Raymond Devillers[b]

[a] Department of Computing Science, Carl von Ossietzky Universität Oldenburg, D-26111 Oldenburg, Germany
[b] Département d'Informatique, Université Libre de Bruxelles, Boulevard du Triomphe – C.P. 212, B-1050 Bruxelles, Belgium

ARTICLE INFO

Article history:
Received 22 December 2016
Received in revised form 4 July 2017
Accepted 31 July 2017

Keywords:
Choice-freeness
Labelled transition systems
Petri nets
Synthesis

ABSTRACT

This paper proposes a fail-fast pre-synthesis method supporting the synthesis of unlabelled Petri nets from labelled transition systems, while focusing on the class of choice-free systems. Such systems have applications, amongst others, in hardware design and in manufacturing. Necessary conditions which must be satisfied by any choice-freely Petri net synthesisable transition system will be identified. They include the prime cycle property and the distance path property, as well as various forms of determinism. Checking such properties before synthesis allows the early detection of non-synthesisable transition systems and the production of meaningful messages about the reasons of synthesis failure. Various interdependencies between these properties will be revealed. The prime cycle property and the distance path property will be shown to imply other ones. This allows pre-synthesis to be organised in an efficient way, because implied properties do not need to be checked if the properties they are implied by are already checked.

© 2017 Elsevier B.V. All rights reserved.

1. Introduction

Synthesis of Petri nets [1,20] has applications in hardware design [16,26] as well as in manufacturing [23,31]. The idea is to view a labelled transition system as a (usually sequential) specification and to derive for it a (usually much smaller, concurrent, and correct-by-design) Petri net implementation with the same behaviour. Synthesis algorithms exist, provided the input is finite [3,13,16],[2] but their performance is heavily dependent on the input size.[3] It is therefore important to investigate potential for making them more efficient.

This paper presents a fail-fast pre-synthesis approach serving to detect inputs which are unsuitable for synthesis. Pre-synthesis offers the following advantages:

- For many ill-designed transition systems, synthesis failures can be predicted at an early stage, avoiding unnecessary, and usually costly, computations.

---

[*] Corresponding author.
E-mail addresses: eike.best@informatik.uni-oldenburg.de (E. Best), rdevil@ulb.ac.be (R. Devillers).

[1] Supported by DFG (German Research Foundation) through grants Be 1267/15-1 `ARS` (Algorithms for Reengineering and Synthesis) and Be 1267/16-1 `ASYST` (Algorithms for Synthesis and Pre-Synthesis Based on Petri Net Structure Theory).

[2] And even for some infinite inputs specified in finitary ways [18].

[3] To date, in fact, a few million states will almost certainly beat all of the existing tools, including [11–13,15,28].

http://dx.doi.org/10.1016/j.scico.2017.07.005
0167-6423/© 2017 Elsevier B.V. All rights reserved.

- Finding the causes of a synthesis failure is supported by issuing meaningful error messages, as opposed to starting synthesis and waiting for possibly more cryptic messages (or even a mere time-overflow failure).

The price to pay is that pre-synthesis necessarily produces some computational overhead for non-failing inputs. This can be mitigated by two measures: firstly, by realising pre-synthesis in such a way that it anticipates computations, and creates data structures, which are useful during synthesis; and secondly, by organising its various checks for input failures as reasonably and efficiently as possible. It is the last aspect which this paper is most concerned about.

Let us call a Petri net N a *solution* of an edge-labelled transition system TS if the reachability graph of N is isomorphic to TS. The basic synthesis problem treated extensively in [1] can be stated more precisely as follows[4]:

$$\begin{array}{ll} \textit{Given}: & \text{A finite edge-labelled transition system } TS. \\ \textit{Decide}: & \text{Is there an unlabelled Petri net } N \text{ solving } TS? \end{array} \tag{1}$$

where "unlabelled" means that no transition inscriptions are allowed.[5] The algorithm described in [1,3] involves deriving many systems of linear-algebraic inequalities from a given TS and returning a Petri net solution of TS if all of them are solvable. It is polynomial, likely with an exponent around 6 in the number of states.[6]

With a view to applications, it may be desirable to modify (1) by targeting various classes of Petri nets. For instance, in a hardware design context [16], it is interesting to replace "Petri net" by "elementary net", and it is possible to obtain a new algorithm for (1) by adding inequalities [29]. Theoretically, however, the problem then becomes NP-complete [2].

In this paper, we shall target a specific class of Petri nets called *choice-free nets*. Thus, we consider the following problem:

$$\begin{array}{ll} \textit{Given}: & \text{A finite edge-labelled transition system } TS. \\ \textit{Decide}: & \text{Is there a choice-free Petri net } N \text{ solving } TS? \end{array} \tag{2}$$

Choice-free nets are defined by the structural requirement that every place has at most one outgoing transition [8,17,31]. They are distinct from (in fact, incomparable with) free-choice Petri nets [19] and their generalisations, equal-conflict nets [30]. Because of the presence of arbitrary arc weights and unrestricted place inputs, choice-free nets are also much more powerful than marked graphs [14]. In asynchronous hardware design, choice-free solutions are welcome because they are race-free [26], and in general, they always permit a physically distributed implementation [7].

The present paper investigates how a *pre-synthesis* algorithm for (2) can be organised in efficient ways. In a previous paper [8], it was shown how the *synthesis* algorithm for (2) can be refined by judiciously removing (rather than adding) inequalities. Based on this mathematical groundwork, [10] shows that the combination of pre-synthesis and synthesis yields a powerful, efficient, and successfully benchmarked, full synthesis algorithm for (2).

The envisaged pre-synthesis phase works by first consulting Petri net structure theory for a comprehensive list of necessary properties a transition system must enjoy if it is isomorphic to the reachability graph of a choice-free Petri net. As many as possible of those properties should be checked during pre-synthesis on any arbitrary transition system, knowing that synthesisable ones satisfy them by default. Not all of them may be easy to check, however, but we shall show that they are interdependent and that we can exploit such interdependencies to avoid lengthy explicit (but redundant) tests of properties which have already been tested implicitly.

The main purpose of the present paper is thus to define and to investigate the mathematical interdependencies of structural properties checkable during choice-free Petri net pre-synthesis. The aim in each case is to bring conditions which are easy to test in front of an implication sign, letting other ones be implied, if possible. The overall aim is thus to assist finding a suitable, optimised subset of necessary structural properties to be checked on a transition system during pre-synthesis.

Section 2 recalls – very briefly – the definitions of labelled transition systems and Petri nets. Several properties of transition systems (that are satisfied by choice-free Petri net reachability graphs) are then described. In Section 3, we prove that one of them, the prime cycle property, implies several others. Section 4 presents a similar result with respect to a property called the distance path property. A summary of the paper can be found in Section 5, together with some observations about the difficulty to check the needed properties, and with some concluding remarks.

2. Labelled transition systems, Petri nets, and necessary properties

In this section, the reader will find basic definitions pertaining to labelled transition systems (Definition 1) and to Petri nets (Definition 5). In the central part of the section (consisting of Definitions 2–4 and Fig. 1), various properties applicable to transition systems in general will be specified and illustrated. At the end of the section, it is then shown that all of them

---

[4] Additionally, of course, a solution should be produced if one exists.

[5] This requirement is a basic assumption in [1] and in this paper. It allows the precise analysis of physical distributability [4,7,21,22] but could be relaxed in different contexts [16].

[6] Indeed, if TS has n states, $O(n^2)$ transitions, and m labels, then $(n \cdot (n+1)/2) + O(n \cdot m)$ systems of inequalities, each having $O(n)$ linear inequalities and $O(m)$ variables, have to be solved. Solving a system of k inequalities (with few unknowns) by Khachiyan's algorithm [25], we may expect a runtime of $O(k^3)$, but exact analyses are difficult to get hold of (Evgeny Erofeev: private communication).

Download English Version:

https://daneshyari.com/en/article/6875246

Download Persian Version:

https://daneshyari.com/article/6875246

Daneshyari.com