Models from Scenarios

Robert Lorenz\textsuperscript{1,*}, Jörg Desel\textsuperscript{2}, and Gabriel Juhás\textsuperscript{3}

\textsuperscript{1} Department of Computer Science  
University of Augsburg, Germany  
robert.lorenz@informatik.uni-augsburg.de  
\textsuperscript{2} Department of Software Engineering  
Distance University of Hagen, Germany  
joerg.desel@fernuni-hagen.de  
\textsuperscript{3} Faculty of Electrical Engineering and Information Technology  
Slovak University of Technology, Bratislava, Slovakia  
gabriel.juhas@stuba.sk

Abstract. Synthesis of Petri nets from behavioral descriptions has important applications in the design of systems in different application areas. In this paper we present a survey on the technique of region based synthesis of Petri nets from languages. Each word in a given language specifies one run of the searched Petri net, i.e. represents one observable scenario of the system.

We concentrate on recent developments for languages of different kinds of causal structures (such as partial orders and stratified order structures). Causal structures represent causal relationships between events of one run. Expressible causal relationships are for example direct and indirect causal dependency, concurrency and synchronicity of events.

Concerning infinite languages, several possibilities of a finite representation are discussed. As the goal of synthesis, place/transition nets and inhibitor nets as well as several restrictions of these net classes are used. The presented framework integrates all classical results on sequential languages.

Keywords: Synthesis, Region Theory, Petri Net, Causal Semantics, Partial Language, Partial Order, Stratified Order Structure.

1 Introduction

Synthesis of Petri nets from behavioral descriptions has been a successful line of research since the 1990s. There is a rich body of nontrivial theoretical results and there are important applications in industry, in particular in hardware design \cite{9,19}, in control of manufacturing systems \cite{33} and recently also in process mining \cite{32,31,4,17} and workflow design \cite{12,6}.

The synthesis problem is the problem to construct, for a given behavioral specification, a Petri net such that the behavior of this net coincides with the specified behavior (if such a net exists). There are many different methods which are presented in literature to solve this problem. They differ mainly in the Petri net class and the model for...
the behavioral specification considered. All these methods are based on one common theoretical concept, the notion of a region of the given behavioral specification.

In this paper, we present an overview of region-based synthesis methods, which regard languages as behavioral specifications, where each word in a given language specifies one run of the searched Petri net. Classical results consider sequential languages representing sequential runs of Petri nets. Recent developments examine languages of different kinds of causal structures (such as partial orders and stratified order structures) representing non-sequential runs. Such causal structures are able to represent different causal relationships between events of one run, such as for example direct and indirect causal dependency, concurrency and synchronicity.

In the following we describe the general approach of region based synthesis from languages. Denote the set of runs of a Petri net $N$ by $L(N)$. It depends on the Petri net class and the considered net semantics, which kind of runs are considered in $L(N)$. Formally the synthesis problem w.r.t. different Petri net classes and different language types is:

**Given:** A prefix-closed language $L$ over a finite alphabet of transition names $T$.

**Searched:** A Petri net $N$ with set of transitions $T$ and $L(N) = L$.

This means, we search for an exact solution of the problem. Such an exact a solution may not exist, i.e. not each language $L$ is a net language.

The classical idea of region-based synthesis is as follows: First consider the net $N$ having an empty set of places but all transitions occurring as labels in $L$. This net generates each execution in $L$ (i.e. $L \subseteq L(N)$), because there are no places restricting transition occurrences. But it generates much more executions. Since we are interested in an exact solution, we restrict $L(N)$ by adding places.

There are places $p$, which restrict the set of executions too much in the sense that $L \setminus L(N) \neq \emptyset$, if $p$ together with adjacent weighted arcs is added to $N$. Such places are called non-feasible (w.r.t. $L$). We only add so called feasible places $p$ satisfying $L \subseteq L(N)$, if $p$ is added to $N$ (Figure 1). The idea of region-based synthesis is to add all feasible places to $N$. The resulting net $N_{sat}$ is called the saturated feasible net. $N_{sat}$ has by construction the following very nice property:

**($\min$)** $L(N_{sat})$ is the smallest net language satisfying $L \subseteq L(N_{sat})$.

---

![Fig. 1. The place $p_1$ is feasible, the place $p_2$ is not feasible w.r.t. the language $L = \{a, b, ab, ba, abb, bab\}$ (b is no execution of the net shown in the middle). The place $p_3$ is feasible w.r.t. $L$ for each integer $n \in \mathbb{N}$](image-url)