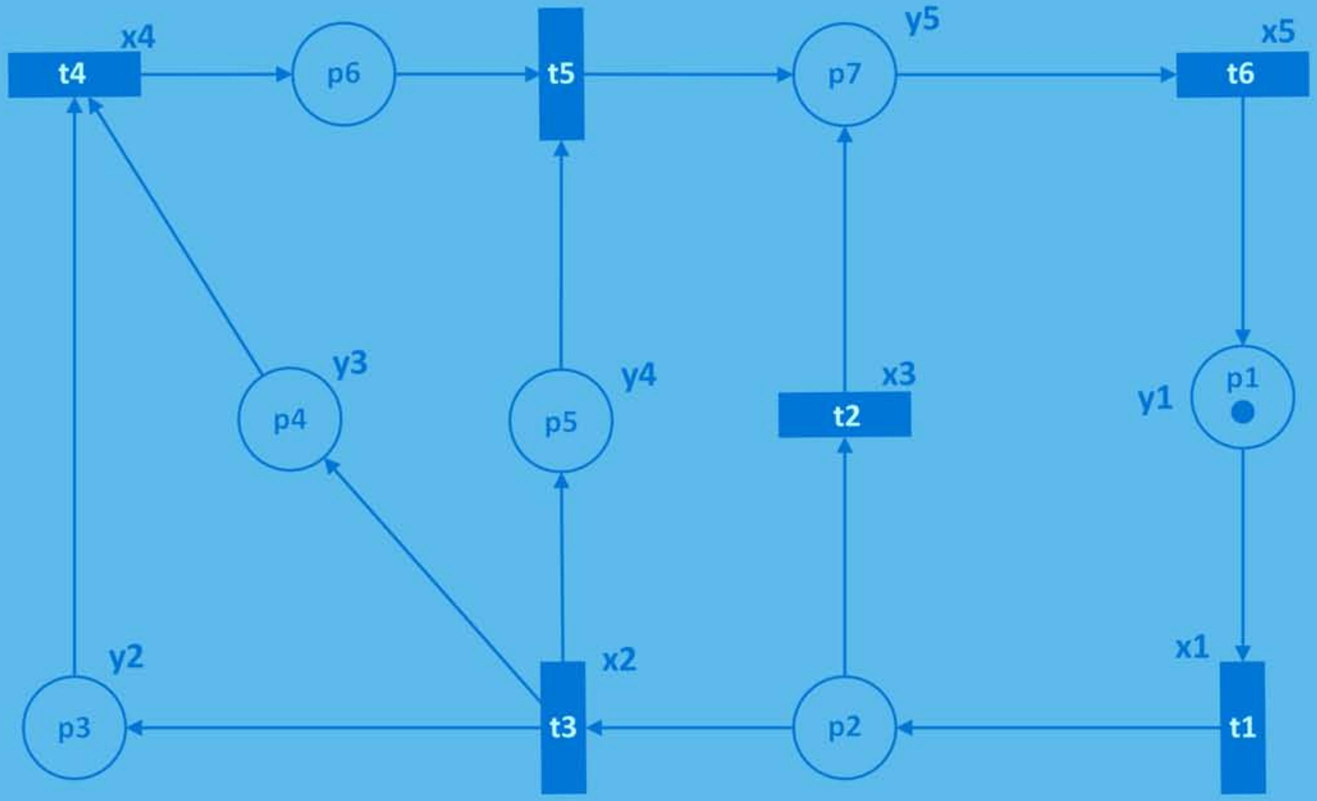
# ANALYSIS OF BOUNDEDNESS AND SAFENESS IN A PETRI NET-BASED SPECIFICATION OF CONCURRENT CONTROL SYSTEMS



Marcin Wojnakowski

Bentham Books

# Analysis of Boundedness and Safeness in a Petri Net-Based Specification of Concurrent Control Systems

Authored by

**Marcin Wojnakowski**

*Institute of Control & Computation Engineering*
*University of Zielona Góra,*
*ul. Szafrana 2, 65-516 Zielona Góra, Poland*

# Analysis of Boundedness and Safeness in a Petri Net-Based Specification of Concurrent Control Systems

Author: Marcin Wojnakowski

ISBN (Online): 978-981-5322-70-5

ISBN (Print): 978-981-5322-71-2

ISBN (Paperback): 978-981-5322-72-9

First published in 2025.

## CONTENTS

# FOREWORD

I am very pleased to introduce here a comprehensive and well-balanced book on the analysis of boundedness and safeness in Petri net-based specification of discreteevent concurrent control systems.

In the dynamic landscape of control systems, Petri nets emerge as a powerful modeling approach. Their simplicity, combined with their ability to represent complex systems, has made them essential in various domains. Petri nets provide a robust framework for specification and analysis of systems in very diverse fields, such as software engineering, robotics and automation, manufacturing and industrial engineering, telecommunications, transportation and logistics, and healthcare and medical systems, just to mention a few.

I particularly like the structure of the book. The first chapter covers introductory topics on control systems and Petri nets, followed by three chapters addressing boundedness and safeness in Petri nets models, where new methods are presented.

The last two chapters cover two case studies, complemented by an appendix containing extensive results of the application of the proposed methods for verification of properties of boundedness and safeness.

The journey with this book begins with a comprehensive overview of the tools and algorithms necessary for analyzing the boundedness and safeness of Petri nets. The novel algorithms proposed in this work are not just theoretical constructs; they are rigorously validated through extensive experimentation on a varied set of 243 Petri nets. This empirical approach provides valuable insights into the practical applicability and limitations of these algorithms. To illustrate the real-world impact, this publication presents a manufacturing control system modeled using Petri nets.

Through this example, the author serves a concrete demonstration of how theoretical principles can be effectively translated into practical solutions, offering a guide for both researchers and industry practitioners.

I am confident that “Analysis of Boundedness and Safeness in a Petri Net Based Specification of Concurrent Control Systems” will become an important resource for researchers, educators, and professionals involved in the design, analysis, and implementation of concurrent control systems based on Petri nets. Its synthesis of theoretical rigor and practical relevance makes it a standout contribution to the field of computer science and engineering.

Summing up my overall impression about this book, I strongly recommend it to anyone who likes to use Petri net models in systems design, emphasizing boundedness and safeness properties verification. The book offers insightful contributions, and may be used as a text for advanced courses on the subject.

**Luís Gomes**
Department of Electrical and Computer Engineering,
NOVA University Lisbon,
Lisbon, Portugal.

# PREFACE

Control systems play a crucial role in various aspects of everyday life, including banking, medical care, manufacturing, transportation, and entertainment. Their continual development necessitates designers to employ advanced and multi-functional tools to facilitate the design process. Petri nets stand out as an effective modeling approach in this context. They have gained popularity for their ease of analysis and graphical representation. Consequently, the analysis of boundedness and safeness of systems specified by Petri nets poses a significant challenge. This book offers a comprehensive overview of applications and algorithms for analyzing the boundedness and safeness of Petri nets.

Boundedness indicates a finite number of reachable states in a control system, while safeness characterizes a binary behavior essential for Petri nets used in configurable FPGAs. Given the exponential computational complexity of Petri net properties, the book proposes novel algorithms to address this analytical gap. These new solutions are elaborately described and supported by experimental results from a set of 243 Petri nets, along with a discussion of their limitations. Furthermore, a real-life manufacturing control system modeled with a Petri net is presented to highlight the benefits of designing concurrent systems using bounded and safe nets.

**Marcin Wojnakowski**
Institute of Control & Computation Engineering,
University of Zielona Góra,
ul. Szafrana 2, 65-516 Zielona Góra, Poland.

# DEDICATION

I dedicate the book to my beloved wife and to bl. Carlo Acutis, who is the patron of this work.

# ACKNOWLEDGEMENT

I would like to express my deep gratitude to Professor Remigiusz Wiśniewski for involving me in fascinating research procedures and publication of many scientific papers and this book.

I would like to extend my thanks to my colleagues Dr. Grzegorz Bazydło and Mateusz Popławski, who provided valuable advice regarding my research. I also want to thank Dariusz Nowosad from the University Language Center of University of Zielona Góra for his professional language proofreading.

Finally, I wish to thank my wife and parents for their support.

CHAPTER 1

# Control Systems

**Abstract:** Control systems surround people everywhere. Recent years have witnessed wide development of such systems. This is due to the growing demand for systems that autonomously control our environment. For example, there are lamps that automatically turn on at dusk, heating systems that turn offwhen we go to work, and similar ideas that make our lives easier, cheaper, and more ecological. These so-called smart solutions can be found in various aspects of human life, such as banking, manufacturing, automotive, transportation, medical care, process mining, and others.

**Keywords:** Analysis, Boundedness, Concurrent control system, Cyber-physical system, Embedded system, Petri net, Safeness.

## 1.1 INTRODUCTION

*Control systems* can be seen as systems that manage other systems, known as operational systems [1]. Such systems send appropriate signals to operational systems based on input values and logic conditions [2]. Several years ago, there were simple control systems with few input and output signals. In recent years, designers have elaborated complex, concurrent, and advanced systems, commonly known as embedded systems, Internet of Things (IoT), flexible manufacturing systems (FMS), and cyber-physical systems (CPS), which is the most popular nowadays [3-34]. Since it is impossible to design such sophisticated systems without the application of computer-aided design (CAD) tools, CAD software is used, for *e.g.*, to perform modeling based on finite state machines, UML diagrams, Petri nets, interpreted nets, *etc.* [11].

*Example 1.1 (Control System)* : Let us imagine a real-life situation and assume that there is a moving car near a playground where children are playing. One child suddenly crosses the road in front of the car. Now consider a modern car with advanced control systems on board. In such a case, the car’s front sensors simultaneously detect an object moving into the road. A dedicated control system simultaneously analyzes the input data from the front sensors and decides to send an emergency signal to the operational system. As a result, the car is automatically halted. Such solutions are offered by Bosch Mobility Solutions[1] (Fig. **1.1**) or Continental-Automotive[2] (Fig. **1.2**). In this way, various control systems make our lives easier, cheaper, and also safer.

**Fig. (1.1).** Automatic emergency braking.

1 More information available online: https://www.bosch-mobility-solutions.com/en/solutions/assistance-systems/automatic-emergency-braking (accessed on Novem- ber 30, 2024)

2 More information available online: https://www.continental-automotive.com/en-gl/Passenger- Cars/Autonomous-Mobility/Functions/Cruising-Driving/Emergency-Brake-Assist/Pedestrian (accessed on November 30, 2024)

**Fig. (1.2).** Emergency brake assist – pedestrian.

## 1.2 CHALLENGES

The real-life automotive example attests to dynamic development and increasing levels of complexity in modern concurrent control systems. It encourages the use of advanced modeling (specifying) approaches, one of which is Petri nets, which, in addition to graphic (graph-based) representation, allow for formal description (specification) [1, 35]. Moreover, Petri nets are also supported by formal mechanisms that make it possible to analyze models. Such analyses permit the examination of systems' reliability and robustness at the specification stage, which may impact the time and costs of a designed concurrent control system [9].

In their very recent research, various scientists have proved the profitability and usefulness of the idea of Petri nets, for *e.g.*, Gomes *et al.* [36], Grobelna *et al.* [9], Lee and Seshia [35], Nuño-Sánchez *et al.* [37], Ramírez-Trevino *et al.* [38], and Wiśniewski *et al.* [1]. Nevertheless, their research also gives

**CHAPTER 2**

# Theoretical Aspects of Petri Nets

**Abstract:** A Petri net is one of the forms of graphical specification and representation of various concurrent control systems. Wide applications of Petri nets can be found in the field of distributed systems, embedded systems, edge computing, manufacturing systems, and cyber-physical systems. They are a useful modeling approach because they are supported by verification, validation, and analytical methods. This way, system designers are able to verify the robustness and reliability of their projected systems.

**Keywords:** Algorithm, Boundedness, Computational complexity, Concurrent control system, Directed graph, Petri net-based specification, Safeness, Verification.

## 2.1 INTRODUCTION

This chapter opens with a discussion on the computational complexity of algorithms, as algorithms can be seen as procedures that solve computational problems. Subsequently, there are examples of graphs as an introduction to Petri nets. Finally, underlying definitions and notation related to Petri nets and their fundamental properties are presented. Then, theories and definitions are discussed that illustrate in detail the currently available algorithms (supported by simple examples). Chapter 4 addresses novel methods to analyze the properties of boundedness and safeness.

## 2.2 ALGORITHMS

First of all, the analysis of the properties of Petri nets, such as boundedness and safeness, can be treated as a computational problem to be solved. *An algorithm* can be viewed as a method for solving a well-specified computational problem [1]. Formally, an algorithm is any well-defined computational procedure that takes a set of values as *input* and returns a set of values as *output* [1]. Therefore, it is a sequence of steps that convert input into output. In our considerations, Petri nets constitute the input to algorithms and the output will result from the analysis of a relevant net property.

**Marcin Wojnakowski**

*Example 2.1 (Algorithm)* : Consider a simple computational problem of computation of absolute values as an example. Algorithm 2.1 proposes a method to solve the problem. We have the following set of inputs of real numbers: $S_{in} = \{x_1, x_2, x_3, \cdots, x_n\}$ and expect the following set of outputs: $S_{out} = \{|x_1|, |x_2|, |x_3|, \cdots, |x_n|\}$. Firstly, an empty set is assigned to $S_{out}$. Then, for each number $x$ of the input set, the value is calculated: $|x|$ and added to the output set $S_{out}$. Finally, the output set contains absolute values of the numbers from the input set.

```
   Data: set $S_{in} = \{x_1, x_2, x_3, \cdots, x_n\}$
   Result: set $S_{out} = \{|x_1|, |x_2|, |x_3|, \cdots, |x_n|\}$
1  $S_{out} \leftarrow \emptyset$ ;
2  foreach variable $x$ of set $S_{in}$ do
3      add $|x|$ to $S_{out}$ ;
4  end
```

**Algorithm 2.1:** An algorithm example — calculation of the absolute value.

## 2.3 COMPUTATIONAL COMPLEXITY

Each computer algorithm can be characterized by its computational complexity. Computational complexity is necessary to estimate the efficiency of a method. Below are some preliminaries related to the computational complexity of algorithms [2], which we will describe in the subsequent chapters of this book.

**Definition 2.1** *Time complexity* of an algorithm is the function $f : \mathbb{N} \to \mathbb{N}$ such that $f(n)$ is the maximum number of iterations that the algorithm uses on any input of length $n$.

**Definition 2.2** *An upper bound* for $f(n)$ is $g(n)$ that $f(n) = O(g(n))$ .

**Definition 2.3** An algorithm is bounded by *a polynomial* in the size of inputs $n$ if the number of its iterations is estimated as

$$f(n) = O(n^c) \,, \tag{2.1}$$

where $c > 0$ .

**Definition 2.4** An algorithm is bounded by *an exponential* in the size of inputs $n$ if the number of its iterations is estimated as

$$f(n) = O(c^n) \,, \tag{2.2}$$

where $c > 1$ .

**Definition 2.5** *A polynomial complexity (polynomial time)* of an algorithm indicates that the total run time of the algorithm to generate all outputs is bounded by a polynomial in the size of the input.

**Definition 2.6** *An exponential complexity (exponential time)* of an algorithm means that the total run time of the algorithm to generate all outputs is bounded by an exponential in the size of inputs.

*Example 2.2 (Computational Complexity)* : The computational complexity of Algorithm 2.1 can be estimated as follows. The *foreach* loop on lines 2–3 is performed $n = |S_{in}|$ times. Hence, the algorithm is bounded by the function $O(n)$, where $n$ is the number of elements in the input set $S_{in}$. It is said to be characterized by polynomial (first degree) computational complexity.

**> Efficiency and Effectiveness**

Computer science algorithms are usually measured in two directions: execution time and memory usage by the program. The *efficiency* term in this book refers to the optimization (minimization) of the execution of the run time with respect to the response in the assumed time. Thus, it does not include other aspects. The focus on improvements is only to reduce running time while maintaining correct results (*effectiveness*).

## 2.4 GRAPHS

Before we define Petri nets, we shall introduce some basic concepts and notation related to graph theory [3-6].

**Definition 2.7** *A graph* is defined by a pair

$$G = (V, \mathbb{E}) , \tag{2.3}$$

where

- $V = \{v_1, v_2, \cdots, v_n\}$ is a finite, nonempty set of vertices,
- $\mathbb{E} = \{E_1, E_2, \cdots, E_m\}$ is a finite set of *unordered* pair of vertices, called edges.

**Definition 2.8** *A digraph (directed graph)* is a pair

$$D = (V, \mathbb{E}) , \tag{2.4}$$

CHAPTER 3

# Boundedness and Safeness

**Abstract:** This chapter provides an in-depth overview of related work. Boundedness and safeness are at the core of various methodologies using the theory of Petri nets. The problems of analyzing the properties of Petri nets, such as boundedness and safeness, were undertaken more than 30 years ago and are still the subject of research. Boundedness is an important feature of various systems, such as manufacturing systems. A bounded Petri net determines a finite number of reachable states.

**Keywords:** Boundedness, Coverability graphs, Petri net invariants, Reachability graphs, Safeness.

## 3.1 INTRODUCTION

Let us imagine a control system specified by a Petri net that is in charge of an elevator. If the Petri net were unbounded, the elevator could behave in an indefinite way and enter states not foreseen by the designer. For example, the elevator could stop between floors or go higher than is allowed and thus cause mechanical failure. Moreover, a safe Petri net may be used to model systems aimed at hardware implementation in microcontrollers or FPGAs — logical control (token/ no token) [1, 2]. In the elevator example, such logical behavior could make the elevator go up or could disable going up.

The most popular analysis techniques [3-6] that allow one to check the boundedness and safeness of a Petri net can be generally divided into two main groups [7]: based on reachability graphs and structural analyzes using linear algebra methodology. The first approach is based on the exploration of reachability graphs.

**Marcin Wojnakowski**

Upon constructing a state space, each subsequent possible reachable state is placed on the graph. In general, the number of reachable states of a system can grow exponentially (a state explosion problem). It translates into exponential run-time of algorithms analyzing properties on reachability graphs. Therefore, for some Petri nets consisting of up to several dozen places or transitions, the methods may not produce a result in the assumed time, *i.e.*, they are not efficient in such cases. The second approach is based on the linear algebra technique [2, 8-12]. In this approach, all minimal invariants of a Petri net are computed (minimal means that they are not contained in other invariants). Place invariants are used by linear algebraic procedures to check the boundedness and safeness. However, the number of all minimal invariants can also grow exponentially. Therefore, both verification paths have serious limitations because the number of reachable states and place invariants in the graph can be exponential [1, 13, 14]. This makes the computational complexity of algorithms investigating these properties exponential, which means that a solution may not be found within the assumed reasonable time.

Below, problems of reachability graph building, computing invariants, and analyses, which attempt to handle these problems by means of popular methods of boundedness and safeness, are presented. In addition, computer-aided tools used in concurrent control specification based on Petri nets are mentioned.

## 3.2 COVERABILITY GRAPHS

Finite graphs can be prepared in which every reachable marking is explicitly represented by a node [15]. Such graphs are known as coverability graphs [13]. Petri net properties can be directly obtained from the exploration of such graphs. Such analyses can guarantee that a Petri net is bounded or unbounded, safe or unsafe, or whether it contains deadlocks. However, this approach can be inefficient as the number of reachable states can be exponential. This means that the operation of the algorithm may not be completed in the assumed time. Net reduction, *i.e.*, simplification of a Petri net with preservation of its boundedness, safeness, and other behavior, can be one of the practical solutions to this problem [1]. However, such reduced graphs can still be too large to be computed [1, 16]. The size of the reductions depends on the characteristics of a Petri net. Some Petri nets can be significantly reduced to make it possible to analyze them efficiently. However, other Petri nets may reduce marginally or not at all, which makes it impossible to verify them.

### 3.2.1 Overview

As Lipton proved in a study [17], the building of a reachability graph requires an exponential space and a decidable marking of its reachability problem [18]. Although space hardness has been shown for Petri nets in general [17], for some classes of Petri nets, *e.g.*, acyclic nets, the state reachability problem can be resolved by means of an integer linear programming problem. Moreover, it can be solved in polynomial time in state machines and marked graphs. The reachability problem is part of the Petri net theory. Many properties, including boundedness and safeness, can be resolved by a system reachability analysis.

In the following sections, the most interesting properties will be described to proceed with the construction techniques of the reachability graphs. Binary decision diagrams (BDD) can be used for some properties using the symbolic traverse technique. However, since the BDD data structure is based on place invariants, reconstructing the behavior of a Petri net is not straightforward [19].

Valmari's set of methods [20] is perfectly applicable to Petri nets with identical concurrent structures. For instance, various identical workflows are triggered simultaneously, and they run in parallel. However, the requirement that all workflows have an identical structure, which is a very restrictive presumption, limits the applicability of this approach. A subnet that reduces into a single transition with preservation properties such as boundedness is presented in some articles [21, 22]. In a study [23], the author creates a dependency graph for removing some substructures without changing the property of reachability. However, the limitation of these approaches is that they are highly based on some particular substructures of Petri nets.

Recently, a state compression approach [24, 25] has been proposed by Cabaisnoet *et al*. The benefit of this methodology is that only a part of the reachability space (called basis markings) is enumerated. This technique can be effectively used to solve controllability problems [26] or opacity problems [27, 28] in Petri nets. In addition, numerous different methods with new classes of Petri nets are developed to handle the marking reachability problem. For example, siphon analysis is applied to determine if deadlock markings exist in $S^3PR$ nets [29-31].

In a paper [19], a practically efficient algorithm based on a reachability graph solves the marking reachability problem in Petri nets. This method has a wide relevance, since it does not depend on particular substructures of the Petri nets. Reachability sets can be precisely characterized by this technique.

**CHAPTER 4**

# Methods of Analyzing Boundedness and Safeness

**Abstract:** In this chapter, novel ideas for possible improvement of Petri nets analysis methods are presented. The proposed algorithms are not only dedicated to larger Petri nets, where, as expounded in detail in the previous chapters, exact methods cannot compute in the assumed time. For larger Petri nets, that is, nets with several and more places, the described algorithms do not provide a perfect solution for all cases because, as discussed earlier, the problems are of exponential character. However, the recent ideas presented in this chapter will certainly support concurrent control designers in the field of modeling and analyzing Petri nets.

**Keywords:** Analysis, Boundedness, Coverability graph, Invariants, Safeness.

## 4.1 BOUNDEDNESS

In the proposed methods for boundedness, we focus on analyses based on reachability graphs as well as on a linear algebra approach. Each of these two approaches finds its application, as both have their own benefits and limitations. As discussed in the previous chapter, these problems possess exponential computational complexity in the general class of Petri nets [1-3]. Therefore, the proposed algorithms are geared toward efficiency and effectiveness. Efficiency is understood as obtaining a response to a given method within the assumed time. And the effectiveness is reflected in the accuracy of the result achieved. Detailed research of the proposed methods in terms of their efficiency and effectiveness is presented in Chapter 5.

### 4.1.1 A Method Based on Reachability Graphs

As explained in the previous chapter, boundedness analyses can be conducted by reviewing reachability graphs. This solution has an obvious disadvantage, as it requires examining all the Petri net markings, and this results in exponential computational complexity. Even for relatively small Petri nets, the current tools for testing boundedness based on state-space analysis may not yield results within the assumed time, that is, they are not efficient. Some popular computer-aided tools, together with their limitations, were discussed in Chapter 3. Upon constructing a finite graph of the reachable states of a given Petri net according to the reference algorithm shown in Algorithm 3.1, we can see that if the conditions of Equation 4.1 are met, a symbol $\omega$ is inserted in a place that will accumulate tokens indefinitely.

$$\begin{aligned} &\exists \mathbf{M_r} \text{ on the path from } \mathbf{M_0} \text{ that} \\ &\qquad \forall p \in P, M_s(p) \geq M_r(p)\,, \\ &\Longrightarrow \forall p \in P, M_s(p) > M_r(p) \colon M_s(p) \leftarrow \omega \end{aligned} \tag{4.1}$$

where $\mathbf{M_r} \neq \mathbf{M_s}$ and $\mathbf{M_r}$ are any reachable markings on the graph from the initial state to the current analyzed marking $\mathbf{M_s}$. The authors used this technique to keep the graph finite even for unbounded Petri nets. We shall apply this to analyze the property of boundedness, and during the construction of the graph, we shall interrupt its operation when such a place is found. It can be noticed that the proposed algorithm finds its special application for Petri nets that are unbounded, because it is not required to explore all markings of a Petri net to prove its unboundedness. The novel algorithm can work best in cases where a Petri net is suspected of being unbounded. For example, the designer has prepared a specification of a system based on a Petri net and wishes to check that he did not make any mistakes by creating an unbounded net, which would lead to an indefinite number of system states and its unpredictable behavior. The novel solution presented here will also coexist with other methods proposed further, *e.g.*, a method on a reduced row matrix, which is dedicated to rapid initial classification of boundedness.

A method for determining boundedness based on the introduced idea is presented in Algorithm 4.1. The novel solution aims mainly at prompt denial of the fact that a Petri net is bounded. The word *prompt* suggests that there is no need to construct a complete reachability graph. The option of analyzing boundedness already at the level of building state space and not after its completion, as is usually the case with other tools, is an indisputable advantage of this proposal. Common-use tools are mainly dedicated to general analysis. Thus, for example, first the state space ofa Petri net is constructed and then a certain property of the net is examined, such as boundedness, safeness or other. On the other hand, the introduced solution is restricted directly to studying the property of boundedness. The construction of the reachability graph is done on lines 2–18 by means of Algorithm 3.1. If a place is found that will keep tokens infinitely (lines 6–8), it is a place where the number of tokens in the current building state $\mathbf{M_s}$ is greater than the number of tokens for the same place in marking $\mathbf{M_r}$. $\mathbf{M_r}$ is any marking from the initial state $\mathbf{M_0}$ to $\mathbf{M_s}$, excluding that for all places; $M_s(p) \geq M_r(p)$ is satisfied. Then, such a place is marked by $\omega$ in the marking $\mathbf{M_s}$ on the graph. Next, the algorithm immediately terminates the graph construction (line 10) with a result of unboundedness (line 22). Otherwise, all states are acquired, and the net is bounded. Therefore, the proposed method is aimed at potentially unbounded Petri nets and perfectly combines with other algorithms proposed in this chapter. Details of effectiveness and efficiency and their applications will be more broadly discussed in Chapter 5.

*Example 4.1 (Boundedness Analysis on a Coverability Graph)* : Below, Algorithm 4.1 is presented step by step on the example of the Petri net shown in Fig. (**3.1**). The input to the algorithm is the Petri net $N_3 = (P_3, T_3, F_3, M_0)$ and the expected result is a text output: net $N_3$ is unbounded. At the beginning, let **M** be equal to the initial state $\mathbf{M_0}$ = [1 0 0 0 0] (line 1). Next, there is a possibility of obtaining a new state (line 2), thus we fire all the enabled transitions in the current state one by one (line 3). At this stage, it is one enabled transition $t_1$, after that, we obtain a new state $\mathbf{M_s} = \mathbf{M_1}$ = [0 1 0 0 0] (line 4). Then, the method checks whether the condition in Equation 4.1 is true (line 6), *i.e.*, there is such a state $\mathbf{M_r}$ reachable from the initial marking that $M_s(p) \geq M_r(p)$ for all $p$ places. No such state is found at present. We add the current state to the collection of markings (line 13), and the fired transition to the set of fired transitions (line 14). Further, the algorithm searches for another state, *i.e.*, the next iteration (line 2). The new state can be obtained by firing the transition $t_2$ (line 3), so that $\mathbf{M_s} = \mathbf{M_2}$ = [1 0 1 0 0]. Again, the checker (line 6) inspects a state that satisfies the condition in Equation 4.1. Such a state $\mathbf{M_r}$ exists on the path from $\mathbf{M_0}$ to $\mathbf{M_{s=2}}$. Therefore, for any place such that $M_{s=2}(p) > M_{r=0}(p)$, an $\omega$ can be assigned (lines 7–8). The place $p_3$ is a place where the tokens will accumulate indefinitely since $M_2(3) > M_0(3)$, *i.e.*, the place is unbounded. It can be noticed that loop of firings $\{t_1, t_2\}$ holds tokens in the place $p_3$. Finally, the algorithm finishes searching for new states (line 10) and informs that net $N_3$ is not bounded (line 22) as expected.

Note that the algorithm for the Petri net $N_3$ in Fig. (**3.1**) terminates with the correct result only after two states are obtained from the initial marking. In the case of the reference method, it is necessary to build an entire reachability graph; in this example, it consists of eight distinct markings. In the worst case, such as a bounded Petri net, the proposed algorithm requires the preparation of a full graph. However, as will be shown by the experimental results presented in Chapter 5, the operating time has been optimized in practical application. The initial suspicion of unboundedness of a Petri net is a principal element, for which a novel solution described in Section 4.1.3 has been prepared.

**CHAPTER 5**

# Experimental Verification of the Proposed Methods

**Abstract:** Experimental research is an essential element in developing novel algorithms and testing already available solutions for their optimal application. Each proposed algorithm should be evaluated using reference methods in terms of its accuracy and operating time. The algorithms introduced in Chapter 4 were tested experimentally. The conducted research was meant to compare the presented methods with reference algorithms. Here, we also discuss the best application of the selected algorithms in order to verify boundedness and safeness. The benchmark library used in the experiments, containing 234 Petri nets, is part of a Hippo project developed at the University of Zielona Góra. The set of test modules consists of various classes as well as features various levels of complexity of the Petri nets used for the specification of hypothetical and real-life examples of control systems.

**Keywords:** Analysis, Boundedness, Benchmarks, Experiments, Safeness, Verification.

## 5.1 INTRODUCTION

Our research focuses on inspecting the effectiveness and efficiency of the selected algorithms. As already stated in previous chapters, effectiveness is understood here as consistency of results and efficiency as time needed to achieve a result. In the case of methods based on the calculation of invariants, the amount of system memory usage is not an obstacle at this stage. Exponentiality with respect to memory is an important optimization problem within state space construction. However, since we must first deal with time complexity, memory performance tests are not part of this work. A comparison of algorithms allowed us to indicate the advantages and disadvantages of the presented methods and to indicate the best cases of their use.

In the following, the Hippo system that was used in the experimental research is introduced. The introduction is followed by a comparison of the proposed analyses

**Marcin Wojnakowski**

of boundedness algorithms to reference methods. Later, safeness methods are scrutinized, and finally, the summary of methods that are optimal to apply in particular cases is presented.

## 5.2 THE HIPPO SYSTEM

There are various tools available on the market to work with Petri nets. Some of these software are probably known only to their authors, and the others are problematic even in their installation or launch. Only some are suitable for daily use by end users, that is, designers of concurrent control systems based on Petri nets. In response to this demand, Hippo [1] [1-3] was launched in 2005, designed as a set of standalone tools for broad Petri net analysis. Initially, the components were oriented to systems based on hypergraph theory. The computer-aided design system has been significantly developed over the past several years. Now, it focuses on control systems specified by a Petri net. The most recent version of Hippo consists of a set of programs that support control system designers, including prototyping, verification, analysis, decomposition, modeling, and automatic code generation (implementation). Furthermore, since they provide a balance between optimal results and computational time, the modules allow solving the same problems using various methods.

Hippo tools include three completely different approaches to the analysis (or decomposition) of control systems. They can be used depending on one's needs. In the previous chapters, we stated that there was no perfect analytical method in the general case of Petri nets. Therefore, the following techniques are available:

- linear algebra approach,
- algorithms based on graph theory,
- algorithms based on hypergraph theory.

In the following, the methods included in Hippo are presented. Moreover, currently, the Hippo project consists of a library of 264 various Petri nets. The software

[1] The detailed description of the project and access to Petri net benchmarks can be found online: https://hippo.uz.zgora.pl (accessed on November 30, 2024)

accepts input as a PNH [2] file. The format includes an incidence matrix of a Petri net and an initial marking vector. In addition, other options can be specified. However, they are optional for analytical proposes. Hippo permits verification by classification of Petri nets, analysis of concurrency, and sequentiality relations. Furthermore, it allows the computation of place invariants and the checks of the safeness and boundedness of the Petri nets. Finally, the Hippo system can return the model in one of the following formats: Verilog (for FPGA implementation), PNG (graphical representation of the model), and XML (for PIPE).

## 5.3 BOUNDEDNESS

The novel algorithms for boundedness analysis have been introduced in Section 4.1. Three experiments are performed. The first compares methods based on reachability graphs. The second report exposes algorithms based on linear algebra, *i.e.*, computation of invariants of places. The sample results of the conducted research are shown in Table **5.1** (reachability graph-based algorithms) and Table **5.2** (invariant coverage-based algorithms). Furthermore, the third experiment, *i.e.*, a method to estimate which of the proposed algorithms should be used in the particular case, is shown in Table **5.3.** The full results of all experiments are attached in Appendix A. The tests are prepared using a dedicated computational server equipped with an Intel(R) Xeon(R) Gold 5220 @2.2 GHz processor and 128 GB of RAM.

First of all, a fundamental difference between methods based on browsing state space and computation of place invariants can be identified. An overview of all possible states gives the answer to whether a Petri net is bounded or unbounded. On the other hand, algorithms based on computing invariants are usually faster (with shorter run-time). In the case of analysis of invariants, we talk about their coverage of a Petri net. The coverage guarantees the boundedness of a Petri net; however, the lack of coverage does not indicate that the net cannot be bounded. Generally, such cases are rare. In our test library, there are 14 such Petri nets out of 243, which is less than 6%. Of the selected Petri nets shown in the tables, *balduzzi1* is an example of an uncovered and bounded Petri net.

Table **5.1** compiles reachability graph-based algorithms for boundedness verification. The reference method for full-graph construction presented in Algorithm 3.1 and its implementation for the purpose of experimental research is the first reference

[2] Petri Net Hippo format

CHAPTER 6

# Effective and Efficient Analysis of Boundedness and Safeness - Case Study

**Abstract:** This chapter deals with a case study of a Petri net-based specification of a manufacturing system. The purpose of this chapter is to introduce one of the manufacturing systems and to illustrate the proposed analytical methods on a reallife example. This chapter also provides a synthetic summary of the research with the presentation of key innovative elements of the author's over seven-year work. The chapter comments on the proposition as well as on the contribution of this book. Moreover, it outlines possible directions for further endeavors.

**Keywords:** Boundedness, Case study, Effectiveness, Efficiency, Experimental verification, Manufacturing system, Petri net-based specification, Safeness.

## 6.1 INTRODUCTION

Petri nets have gained popularity among researchers and engineers due to the combination of their graphic form of presentation with a mathematical description of operation and the possibility of modeling concurrent control systems. Based on mathematical formalism, it is possible to clearly determine the correctness and robustness of the modeled system, while graphical projection of algorithms controlling control systems promotes the ease of the design process. In addition, it is possible to automatically generate codes ready for implementation in microprocessor or FPGA devices.

The gap in effective and efficient methods of analyzing boundedness and safeness, *i.e.*, two key properties in modeling systems specified by Petri nets, constituted a chief motivation to undertake the research described in this book. The analysis of the properties is not a trivial issue as it involves exponential computational complexity in the general case. Therefore, the aim of this work was to propose novel methods of verification, dedicated separately to the efficiency or effectiveness criterion. The introduced algorithms were implemented and compared to known reference methods by several comprehensive tests. A library of 243 Petri nets of various levels of complexity from the Hippo project constituted the basis for the test modules applied. The full results of the experiments performed can be found in Appendix A. The novel algorithms implemented have become part of the Hippo project developed at the University of Zielona Góra.

## 6.2 CASE STUDY EXAMPLE

The flat bar manufacturing process consists of nine points as described in a study [1].

1. Initially, a customer reports a demand for a product with specified parameters.
2. The documentation provided by the customer is studied. Designers assess the feasibility of the order. At this stage, the documentation is accepted or changes are made after consulting with the customer.
3. The approved documentation is delivered to the implementation team and material is ordered in order to manufacture a flat bar of a length specified by the customer within the range of 3 to 6 meters.
4. The material received from the warehouse that does not meet the requirements is adapted to the specifications. The processed material is transferred to a band saw station.
5. The operator cuts the material accordingly. Production waste is stored in postproduction waste storage.
6. The semifinished product is transported for milling. The scrap material formed during the grinding process is stored in a special place.
7. The quality of the resulting product is controlled. The produced product that does not meet the requirements is disposed of as waste. The product that requires corrections is subject to additional processing. The produced product compliant with the quality parameters is transported to the pallet packing station.
8. The products are packed on pallets and other production activities are conducted.
9. Finally, the ready pallets are handed over to the customer.

With reference to the informal specification, a model of the described manufacturing system was prepared on a Petri net. A Petri net used for the specification is required to be safe thus also bounded. In addition, the net features input and output ports to communicate with the environment. Strictly speaking, it is an interpreted net. The analyzed system may be found in a production company that belongs to a group of small and medium-sized enterprises of western Poland. They specialize in manufacturing and providing services to other industrial companies. The business operates in the field of processing large-format materials by means of laser and plotter technology. The Petri net-based specification of this system is shown in Fig. (**6.1**). This Petri net consists of 24 places and 18 transitions along with four input signals and twenty output signals. Active input signals are assigned to the transitions under some conditions and are described in Table **6.2**. On the other hand, the output signals that activate certain actions are related to places and are presented in Table **6.1**. A transition is enabled as defined by Def. 2.20, including the fulfillment of the logical input conditions assigned to it. The output signals connected to a given place are activated when the token is in that place. For this reason, the net must be safe.

## 6.3 BOUNDEDNESS AND SAFENESS ANALYSIS

Below, a Petri net specifying the introduced manufacturing system by means of proposed algorithms is analyzed. It should be noted at the beginning that the analysis of boundedness and safeness by means of popular CAD tools, such as PIPE and IOPT-Tools, resulted in failure due to the problems described in this book. Let us begin with the analysis of boundedness, which will be followed by the analysis of safeness.

At the beginning, we used the 3rd proposed method (q.v. Algorithm 4.3), which revealed which algorithm to choose for optimal verification. This algorithm informs in polynomial time that a Petri net is uncovered by place invariants, so there are probably unbounded places in it. Hence, the 1st proposed method (q.v. Algorithm 4.1) on the reachability graph is selected. This algorithm interrupts the construction of the graph at the first unbounded place found, *i.e.*, $p_{12}$. This way, we know that the Petri net is unbounded, and therefore also unsafe. Places $p_{12}$ and $p_{16}$ (marked in red in Fig. (**6.1**) do not have output transitions. This is one of the frequent mistakes made by designers. The model must therefore be improved. We add output transition $t_{19}$ of $p_{12}$ and $p_{16}$ places, along with the transition output place $p_{25}$ (marked in green), to the corrected specification, as shown in Fig. (**6.2**). They are responsible for waste storage and are synchronized with the main production process in transition $t_{13}$.

# APPENDIX A

# Detailed Experimental Results

This appendix includes full experimental verification of properties of boundedness and safeness.

## A.1 BOUNDEDNESS

**Table A.1. Full results of experiments, state space-based algorithms.**

| Name | $\|P\|$ | $\|T\|$ | Reference Method Bounded Runtime [ms] | | 1st Proposed Method Bounded Runtime [ms] | |
|---|---|---|---|---|---|---|
| prio ex | 2 | 3 | true | 56.8116 | true | 18.2667 |
| semāphore | 3 | 4 | true | 22.6309 | true | 19.2944 |
| coloured | 4 | 3 | true | 25.5637 | true | 17.2615 |
| traffic light v2 | 4 | 3 | true | 18.1265 | true | 15.3904 |
| pn silva 05e | 4 | 4 | false | 23.7932 | false | 16.9574 |
| tank heating | 4 | 4 | true | 19.8070 | true | 18.3392 |
| Entrance1 | 4 | 6 | true | 21.8608 | true | 15.8056 |
| kovalyov92 | 4 | 6 | true | 20.1515 | true | 14.5507 |
| return_books | 4 | 6 | true | 22.5549 | true | 16.4531 |
| pn_silva_05c | 5 | 3 | false | 23.8781 | false | 16.9954 |
| PUMA_unloading | 5 | 3 | true | 57.5249 | true | 18.7439 |
| consumerReachability | 5 | 4 | false | 22.2833 | false | 21.4140 |
| pn_silva_05b | 5 | 4 | false | 33.7118 | false | 21.9474 |

***(Table A.1) cont.....***

| | | | | | | |
|---|---|---|---|---|---|---|
| pn_silva_05f | 5 | 4 | true | 44462.0000 | true | 19.1288 |
| pn_silva_04 | 5 | 6 | false | 84.3863 | false | 21.4031 |
| invariants_exponent_3_2 | 6 | 2 | true | 20.4170 | true | 14.3074 |
| np3 | 6 | 3 | true | 61.0780 | true | 18.5387 |
| gals-example | 6 | 4 | true | 58.9145 | true | 17.6626 |
| lnet_p1n1 | 6 | 4 | true | 21.2580 | true | 15.8184 |
| pcncp | 6 | 4 | true | 23.5670 | true | 18.2273 |
| pn_silva_02 | 6 | 4 | true | 55.8726 | true | 19.4230 |
| PUMA_loading | 6 | 4 | true | 75.6695 | true | 24.4826 |
| RHINO_loading | 6 | 4 | true | 59.6393 | true | 16.7076 |
| RHINO_unloading | 6 | 4 | true | 21.8541 | true | 18.7518 |
| lnet_p1n2 | 6 | 5 | true | 23.0505 | true | 13.9244 |
| lnet_p6n3 | 6 | 5 | true | 50.4137 | true | 15.4346 |
| net1 | 6 | 5 | true | 62.4939 | true | 17.8283 |
| traffic_lights | 6 | 5 | true | 19.8527 | true | 14.4626 |
| fig311_01 | 6 | 6 | true | 29.7733 | true | 19.0608 |
| PNwD | 6 | 6 | true | 76.1078 | true | 33.1066 |
| pn_desel_03 | 6 | 7 | true | 32.4344 | true | 17.1911 |
| communications_protocol | 6 | 8 | true | 25.1123 | true | 16.3243 |
| health_care_process | 6 | 8 | true | 22.3302 | true | 14.3417 |
| cncrr001 | 7 | 4 | true | 19.9029 | true | 16.0355 |
| FMS_main_SIPN | 7 | 4 | true | 23.7193 | true | 15.9449 |
| agostini1 | 7 | 5 | true | 18.4699 | true | 17.2495 |
| CNC_machine | 7 | 5 | true | 21.2913 | true | 17.5161 |
| transfer | 7 | 5 | true | 24.6107 | true | 19.3973 |
| voorhoeve3 | 7 | 5 | true | 18.0765 | true | 14.1707 |
| net2 | 7 | 6 | false | 73.3893 | false | 26.4170 |
| pnbr_expl_05 | 7 | 6 | true | 23.4051 | true | 20.0628 |
| two_traffic_lights | 7 | 6 | true | 18.2165 | true | 16.2687 |
| pn_desel_01 | 7 | 7 | true | 63.6640 | true | 22.6454 |
| silva1 | 7 | 7 | true | 49.9186 | true | 19.6985 |
| traffic_light_v1 | 8 | 3 | true | 16.7578 | true | 16.0724 |
| agerwala1 | 8 | 4 | true | 20.2320 | true | 16.3630 |
| gaubert2 | 8 | 4 | true | 22.4147 | true | 15.9989 |
| mixer_mod2 | 8 | 5 | true | 24.3639 | true | 21.2758 |
| pn_desel_02 | 8 | 5 | true | 66.3045 | true | 22.9693 |
| prod_cons | 8 | 5 | true | 87.3341 | true | 36.9736 |

***(Table A.1) cont.....***

| | | | | | | |
|---|---|---|---|---|---|---|
| sub-task_of_PLT_and_PMN˜ | 8 | 5 | true | 18.9115 | true | 15.4892 |
| bridge | 8 | 6 | true | 34.7380 | true | 21.3181 |
| cp2 | 8 | 6 | true | 33.6237 | true | 18.8141 |
| Exe5_split | 8 | 6 | true | 61.8587 | true | 17.6638 |
| img_280 | 8 | 6 | true | 31.9077 | true | 20.0653 |
| lab5 | 8 | 6 | true | 26.3049 | true | 18.8704 |
| TP5_I | 8 | 6 | true | 19.2282 | true | 17.7237 |
| bit_protocol | 8 | 7 | false | 53.0126 | false | 27.6159 |
| net4 | 8 | 7 | true | 25.4435 | true | 21.7310 |
| silva14 | 8 | 7 | true | 61.6688 | true | 21.8435 |
| elevator_2 | 8 | 8 | true | 26.6660 | true | 16.9500 |
| girault8 | 8 | 8 | true | 91.0943 | true | 28.9878 |
| net3 | 8 | 8 | true | 65.6040 | true | 26.5484 |
| hard_case | 9 | 3 | true | 20.7165 | true | 13.7562 |
| invariants_exponent_3_3 | 9 | 3 | true | 21.6780 | true | 13.9898 |
| bridge_semaphore | 9 | 6 | true | 27.2764 | true | 22.8517 |
| cortadella1 | 9 | 6 | true | 29.8670 | true | 17.4769 |
| lnet_p1n4 | 9 | 6 | true | 31.9204 | true | 17.6487 |
| multi_robot | 9 | 6 | true | 36.1574 | true | 24.6878 |
| oneWayTransmissionSystem | 9 | 6 | true | 81.1547 | true | 25.1693 |
| semaphore2 | 9 | 6 | true | 63.7779 | true | 18.6128 |
| simple_production_system | 9 | 6 | true | 73.2633 | true | 24.9712 |
| lnet_p2n3 | 9 | 7 | true | 22.2145 | true | 16.9937 |
| mutual exclusion | 9 | 7 | true | 69.1031 | true | 21.5975 |
| miczulski1 | 9 | 8 | true | 67.0112 | true | 20.4400 |
| pnbrexpl 03 | 9 | 8 | true | 28.3472 | true | 20.9835 |
| reactor small | 9 | 8 | true | 63.2469 | true | 20.8127 |
| pn silva 01 | 9 | 9 | true | 23.4675 | true | 20.5181 |
| medeiros1 | 9 | 13 | true | 70.0372 | true | 18.9670 |
| vanDerAalst6 | 9 | 13 | true | 23.0026 | true | 17.6228 |
| np5 | 10 | 5 | true | 25.4073 | true | 19.3974 |
| ConsistentExampleMessageView | 10 | 6 | true | 65.9098 | true | 31.7462 |
| pn silva 05d | 10 | 6 | true | 62.7914 | true | 19.0724 |
| exOR | 10 | 7 | true | 21.0055 | true | 14.3336 |
| girault4 | 10 | 7 | true | 41.7014 | true | 28.7987 |
| speedway | 10 | 7 | true | 25.8559 | true | 18.4950 |
| barkaoui2 | 10 | 8 | true | 21.4339 | true | 16.9570 |

# ACRONYMS

| | | |
|---|---|---|
| **ACN** | = | Asymmetric Choice Net |
| **BDD** | = | Binary Decision Diagram |
| **BRG** | = | Basis Reachability Graph |
| **CAD** | = | Computer-Aided Design |
| **CPS** | = | Cyber-Physical System |
| **EFC** | = | Extended Free-Choice net |
| **FCN** | = | Free-Choice Net |
| **FMS** | = | Flexible Manufacturing System |
| **FPGA** | = | Field-Programmable Gate Array |
| **FSM** | = | Finite State Machine |
| **GPU** | = | Graphics Processing Unit |
| **ILPP** | = | Integer Linear Programming Problem |
| **IOPT** | = | Input-Output Place-Transition net |
| **IoT** | = | Internet of Things |
| **LTL** | = | Linear-time Temporal Logic |
| **MG** | = | Marked Graph |
| **MIP** | = | Mixed Integer Programming |

| | | |
|---|---|---|
| **NP-hard** | = | Non-deterministic Polynomial-time hardness |
| **PLC** | = | Programmable Logic Controller |
| **PN** | = | Petri Net |
| **PNH** | = | Petri Net Hippo format |
| **PNML** | = | Petri Net Markup Language |
| **RAM** | = | Random Access Memory |
| **SM** | = | State Machine |
| **SMC** | = | State Machine Component |
| **UCF** | = | User Constraint File |
| **UML** | = | Unified Modeling Language |
| **VHDL** | = | Very high-speed integrated circuit Hardware Description Language |
| **XML** | = | Extensible Markup Language |

# SUBJECT INDEX

## Marcin Wojnakowski

Dr. Marcin Wojnakowski received a Ph.D. in computer engineering from the University of Zielona Góra in 2023. Currently, he is an assistant professor at the Institute of Control and Computation Engineering at the University of Zielona Góra (Poland). His research interests include Petri nets, formal modeling, analysis, and decomposition of concurrent control systems described by Petri nets. Since 2017, he has been a member of the research project Hippo www.hippo.uz.zgora.pl.