New Methodology for Backward Analysis of Reconfigurable Event Control Systems using R-TNCESs

Yousra Hafidi[1,2,3,4] [a], Laid Kahloul[3] and Mohamed Khalgui[1,2] [b]

[1] *School of Electrical and Information Engineering, Jinan University, China*

[2] *LISI Laboratory, National Institute of Applied Sciences and Technology, University of Carthage, Tunis 1080, Tunisia*

[3] *LINFI Laboratory, Computer Science Department, Biskra University, Algeria*

[4] *University of Tunis El Manar, Tunis, Tunisia*

Keywords: Reconfigurable Systems, Modeling and Verification, Petri Net, Backward Reachability, Model-base Diagnosis.

Abstract: This paper deals with reconfigurable discrete event control systems (RDECSs). We model RDECSs using reconfigurable timed net condition/event systems (R-TNCESs) formalism which is an extension from Petri nets to deal with reconfiguration properties. Model-based diagnosis algorithms are widely used in academia and industry to detect faulty components and ensure systems safety. The application of these methods on reconfigurable systems is impossible due to their special behavior. In this paper, we propose accomplishing techniques of backward reachability to make reconfigurable systems model-based diagnosis possible using R-TNCESs. The flexibility among reconfigurable systems like RDECSs allows them to challenge recent requirements of markets. However, such properties and complicated behavior make their verification task being complex and sometimes impossible. We deal with the previous problem by proposing a new methodology based on backward reachability of RDECSs using (R-TNCESs) formalism including improvement methods. The proposed methodology serves to reduce as much as possible redundant computations and gives a package to be used in model-based diagnosis algorithms. The paper's contribution is applied to a benchmark modular production system. Finally, a performance evaluation is achieved for different sizes of the problem to study benefits and limits of the proposed methodology among large-scale systems.

1 INTRODUCTION

Nowadays flexibility in manufacturing systems is challenging markets. For example, a system with fault tolerance should be dynamic and respond without any malfunction while hardware failures occur. Reconfigurable systems (Aissa et al., 2019; Hafidi. et al., 2019; Ramdani. et al., 2019; Aissa et al., 2018) have flexible configurations that allow them to switch from a configuration to another in order to respond for user requirements or to prevent from system malfunction (Lakhdhar et al., 2018; Ramdani et al., 2018). However, their special behavior and properties of reconfiguration make of them complex discrete event systems. In fact, any failure or dysfunction of a critical system can result serious consequences. Reconfigurable systems like reconfigurable discrete event control systems (RDECS) (Khalgui et al., 2004; Khalgui and Mosbahi, 2010; Khalgui et al., 2007) are often subjected to malfunctions that are due to hardware components breakdowns or software dismisses. A safe system should never reach an undesirable state during its working process (Dubinin et al., 2015).

Many research works ensure safety of systems using methods such as Model-based Diagnosis (Cong et al., 2017; Bennoui et al., 2009; Liu et al., 2016). Model-based diagnosis (MBD) is a verification method that explains an observed system's malfunction (De Kleer and Kurien, 2003). When an abnormal system's behavior is observed, MBD method backtracks system execution in the model, and combines with predefined data to detect faulty components that cause this behavior. Backward reachability is frequently used to construct the backward state space that serves with model checking responding to system diagnosis problems. Model-checking (Baier et al., 2008) is a verification technique that explores possible system states in order to check if a system meets its specifications. If a required property

[a] https://orcid.org/0000-0002-3543-6731
[b] https://orcid.org/0000-0001-6311-3588

Hafidi, Y., Kahloul, L. and Khalgui, M.
New Methodology for Backward Analysis of Reconfigurable Event Control Systems using R-TNCESs.
DOI: 10.5220/0007979901290140
In *Proceedings of the 14th International Conference on Software Technologies (ICSOFT 2019)*, pages 129-140
ISBN: 978-989-758-379-7

is proved false, model checking provides the counterexample that falsified it. One of the main problems is how to check the largest possible state spaces and treat them as quick as possible with current means of processors and memories. Existing research works (Baier et al., 2008) have proven results for larger systems state spaces by including some clever algorithms. Consequently, more problems are covered.

Despite the advantage of system diagnosis method, there still a lack of research works on diagnosis of reconfigurable systems. Their special behavior as well as their reconfiguration properties (Wang et al., 2011; Gharsellaoui et al., 2012; Ghribi et al., 2018) should be taken into consideration. In addition, the diagnosis of these complex systems like RDECSs needs optimization methods that improve the process and prevent unnecessary redundant computations.

In order to deal with previous problems, we propose in this paper the following contributions:

1. A backward reachability method for R-TNCESs formalism to facilitate reconfigurable systems diagnosis: backward reachability is used rather than forward reachability (for ordinary Petri nets, colored Petri nets ...etc. (Pózna et al., 2016; Bhandari et al., 2018)) to solve systems diagnosis. R-TNCESs reverse rules and accomplishing techniques are proposed to run backward reachability of reconfigurable systems. Our motivation about using R-TNCESs formalism resides in the way that unlike most other formalisms, R-TNCESs are modular and support modeling of system reconfigurations. In addition, the composition of interconnected modules communicating with signals, deals with interactions that actually happen between sensors and actuators in reconfigurable discrete event control systems (Zhang et al., 2013; Hafidi et al., 2018; Naidji et al., 2018), i.e., sensors send signals to activate actuators. By setting this method, the application of classical algorithms of model-based diagnosis on reconfigurable systems becomes possible using R-TNCESs.
2. A new methodology to cover a wider state space and resolve more problems. Diagnosis is a time and space consuming problem, and the proposed methodology in this paper includes improvement methods that serve to prevent redundancies during backward reachability analysis.

The purpose of this paper is to propose accomplishing methods to allow the application of classical model based diagnosis algorithms on reconfigurable systems using R-TNCES formalism. Note that the problem of applying the classical algorithms of model-based diagnosis is left outside the scope of this paper.

To the best of our knowledge, (1) no existing previous works have proposed methods for reconfigurable systems backward reachability, (2) no existing rules showing how to reverse a system modeled by R-TNCES formalism, and (3) no research works deal with optimization of R-TNCESs backward state space to improve model-based diagnosis abilities.

The paper's contribution is applied to a real case study: FESTO modular production system (FESTO MPS) (Koszewnik et al., 2016; Khalgui et al., 2011), which is an industrial reconfigurable benchmark platform. Obtained results show that after applying proposed methods, classical algorithms of model based diagnosis becomes possible on R-TNCESs. In addition, the covered state space using new methodology is improved. A performance evaluation is achieved for different sizes of problems.

The remainder of the paper is organized as follows. Section 2 recalls the most recent basic elements of R-TNCESs formalism, introduces backward reachability method concepts, presents the proposed R-TNCES reverse method that will be used as a basic element in backward reachability of R-TNCESs, and finally reminds *Mu* method that will be used to improve computations. Section 3 explains the main motivations of this paper, proposes the new methodology of backward reachability including *Mu* improvement method, presents the algorithm and computes its complexity. Section 4 is the experimentation part which contains some applications and results. Finally, Section 5 concludes this paper and describes the future work.

2 THEORETICAL FOUNDATIONS

In this section we first introduce an extension from Petri nets formalism (Qin et al., 2012) called reconfigurable discrete event/condition systems (R-TNCESs) (Zhang et al., 2013; Zhang et al., 2017a). R-TNCESs are used for formal modeling and verification of reconfigurable discrete event control systems. However, their verification is often expensive and needs some improvement methods. In this section, we present backward reachability analysis for R-TNCES and some basic elements proposed to improve the verification task.

2.1 Reconfigurable Timed Net Condition/Event Systems

According to the definition reported in (Zhang et al., 2013; Hafidi et al., 2018), reconfigurable timed net condition/event systems (R-TNCESs) are formally defined by a couple $RTN = (B, R)$ where B (respectively, R) is the behavior (respectively, the control) module of a reconfigurable discrete event control system (RDECS). B is a union of multi-TNCESs represented by

$$B = (P, T, F, W, CN, EN, DC, V, Z_0)$$

where,

- P (respectively, T) is a superset of places (respectively, transitions),
- $F \subseteq (P \times T) \cup (T \times P)$ [1] is a superset of flow arcs,
- $W : (P \times T) \cup (T \times P) \longrightarrow \{0, 1\}$ maps a weight to a flow arc, $W(x, y) > 0$ if $(x, y) \in F$, and $W(x, y) = 0$ otherwise, where $x, y \in P \cup T$,
- $CN \subseteq (P \times T)$ (respectively, $EN \subseteq (T \times T)$) is a superset of condition signals (respectively, event signals), (v) $DC : F \cap (P \times T) \rightarrow \{[l_1, h_1], \ldots, [l_{|F \cap (P \times T)|}, h_{|F \cap (P \times T)|}]\}$ is a superset of time constraints on input arcs of transitions, where $\forall i \in [1, \mid F \cap (P \times T) \mid]$, $l_i, h_i \in \mathbb{N}$ and $l_i < h_i$,
- $V : T \longrightarrow \{\vee, \wedge\}$ maps an event-processing mode (AND or OR) for every transition,
- $Z_0 = (M_0, D_0)$, where $M_0 : P \longrightarrow \{0, 1\}$ is the initial marking, and $D_0 : P \longrightarrow \{0\}$ is the initial clock position.

The graphical model of a TNCES is depicted in Fig. 1.

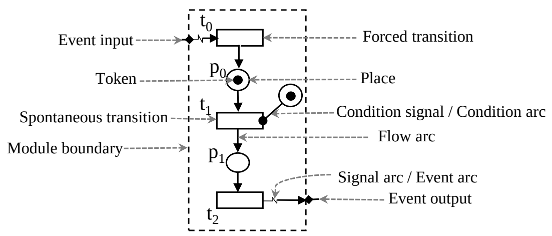


Figure 1: Modules graphical model.

R is a set of reconfiguration rules such that rule r is a structure represented by

$$r = (Cond, s, x)$$

where,

- $Cond \rightarrow \{True, False\}$ is the pre-condition of r, i.e., r is executable only if $Cond = True$,
- $s : TN(^\bullet r) \rightarrow TN(r^\bullet)$ is the structure-modification instruction such that $TN(^\bullet r)$ (respectively, $TN(r^\bullet)$) represents the structure before (respectively, after) applying the reconfiguration r,
- $x : last_{state}(^\bullet r) \rightarrow initial_{state}(r^\bullet)$ is the state processing function. In this paper, we denote by r_{ij} the reconfiguration rule that transforms $TNCES_i$ to $TNCES_j$.

As reported in (Zhang et al., 2013; Hafidi et al., 2018), structure-modification instructions are presented in Table 1. A place is denoted by x, a transition by y, a control component module by $\mathbb{CC}$, and the AND instruction to represent complex modification instructions is presented by ",".

Table 1: Structure-modification instructions of R-TNCESs.

N°	Instruction	Symbol
1	Add condition signals	$Cr(cn(x, y))$
2	Add event signals	$Cr(ev(y, y))$
3	Add control component	$Cr(\mathbb{CC})$
4	Delete condition signals	$De(cn(x, y))$
5	Delete event signals	$De(ev(y, y))$
6	Delete control component	$De(\mathbb{CC})$
7	Add place x with its marking $m(x)$	$Cr(x, m(x))$
8	Add transition y	$Cr(y)$
9	Add flow arc $fa(x, y)$ or flow arc $fa(y, x)$	$Cr(fa(x, y))$ or $Cr(fa(y, x))$
10	Delete place x	$De(x)$
11	Delete transition y	$De(y)$
12	Delete flow arc $fa(x, y)$ or flow arc $fa(y, x)$	$De(fa(x, y))$ or $De(fa(y, x))$
13	Modify transition's y event-processing mode to "AND"	$Mo(AND(y))$
14	Modify transition's y event-processing mode to "OR"	$Mo(OR(y))$

R-TNCESs semantic is defined by both the reconfiguration between TNCESs in behavior module B, and the firing of transitions in each TNCES. The former has the priority to be applied first when its preconditions are fulfilled. The latter depends on the rules of firing transitions in TNCESs and the chosen firing mode. Two kinds of transitions are distinguished, i.e., spontaneous and forced transitions. A transition t is called spontaneous if it is not forced by any other transition (i.e., there are no event signals incoming to t), otherwise it is called forced transition. Each type of transitions has its firing rules. The firing rules are described in detail in (Zhang et al., 2013). However for the firing mode, we adopt the mode in

[1] Cartesian product of two sets: $P \times T = \{(p, t) | \quad p \in P, t \in T\}$.

which only "*one spontaneous transition is fired by step*".

We use the concept of control components (CCs) which was firstly introduced in (Khalgui et al., 2011) in order to model RDECSs. This means that each configuration is a set of CCs interconnected with each other to compose a TNCES. The concept of CCs serves the modularity which enabels the readability and the re-usability of models.

Note that in this paper, we use non marked TNCESs which are TNCESs structures with no given initial marking and non marked R-TNCESs which are R-TNCESs with configurations represented by non marked TNCESs. We use non marked R-TNCESs to describe many possible systems in one model, i.e., each R-TNCES with a possible initial marking represents a system. In addition, by non marked R-TNCESs we are able to describe systems with missed information on their behavior.

2.2 Backward Reachability Analysis

Backward reachability analysis (BRA) theory has been already used for ordinary Petri nets (Anglano and Portinale, 1994) and colored Petri nets (Bhandari et al., 2018). BRA on ordinary Petri nets uses methods such as the reverse of the net, where arcs directions are just reversed (i.e., source becomes target and target becomes source). However, this method is disadvantageous for other high level Petri nets like R-TNCESs. We propose a method that helps to apply BRA on R-TNCESs and study its benefits comparing with other existed theories.

Backward reachability analysis (BRA) can be started from an undesirable state which leads the system to a critical behavior, and it highlights all possible scenarios that cause it. Backward reachability analysis are widely used in model-based diagnosis problems. Let (1) S be a system that works incorrectly, (2) M_S be an abstracted model of S, and (3) $OBS = \{o_1, o_2, ..., o_n\}$ be a set of states specifying the observed misbehavior. The model-based diagnosis method backtracks the system states according to its behavior extracted from M_s, and gives sequences of initial states that are supposed to be reasons for this unpredictable misbehavior starting from *OBS* (Fig.2).

This reasoning is beneficial when we have a non completed model of system S, i.e., sometimes system's behavior cannot be completely modeled 100%, thus, some parts are missed such as the initial state from which a system starts its process. In this case, model M_S is built from hardware components data and their interactions. Using Petri nets formalism, the missed behavior can be presented as lack of information about initial marking (i.e., initial state) in the model. Therefore, M_S is given as a Petri net model without initial marking (i.e., non marked Petri net). Suppose that we aim to explain a misbehavior of such system using the forward method, then all sequences with each possible combination of initial marking is generated. The problem is that in some cases, this reasoning costs a lot of extra time due to a huge number of initial marking possibilities that can even be infinite and not beneficial for the diagnosis process. Some diagnosis works take as an input a system that is modeled using Petri nets like M_s. Then, backward reachability analysis (BRA) is adopted to generate the system's state space starting from the undesirable state in *OBS*. The obtained state space serves to understand possible causes of resulted observations. The main strength point of this method is that it is able to have a model M_S that represents all possible systems with all combinations of inputs and parameters. Therefore, each real system of these possible ones in M_S is supposed to be diagnosed at the end of the process. One of BRA advantages is that it focuses on critical scenarios rather than all possible ones. Unfortunately, it is possible that the graph resulted from BRA be larger or infinite comparing with the original one obtained using the forward reachability analysis (FRA) (Leveson and Stolzy, 1985) for a marked input system. For this case, BRA approach is practical only if the subsequent graph is smaller than the original one obtained by FRA approach. Therefore, generating backward reachability graphs is infeasible in some cases like the above one. In the next subsection, we define what is R-TNCES reverse that will be used to generate R-TNCES backward reachability graphs.

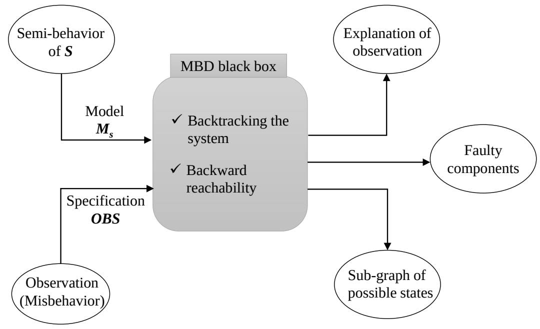


Figure 2: Model based diagnosis and backward reachability.

2.3 Contribution: R-TNCES Reverse

Ordinary Petri nets reversion method can be generalized to R-TNCESs by (1) inverting arcs directions in the nets, and (2) adapting R-TNCESs semantics. The result is a reversed R-TNCES which is possible to be backward analyzed. Adapting R-TNCESs allows to add necessary procedures related to R-TNCESs semantic in order to complete the reversion and to facilitate the analysis among resulted structures. The reversion applied in ordinary Petri nets does not require adaptations, i.e., a simple reversion of arcs directions is sufficient to perform backward reachability. However in R-TNCESs, where the dynamic of this high level Petri net is different and contains more constraints, the inversion of arcs directions is not sufficient. We propose some complementary methods to R-TNCESs reversion method to consider the adaptation of token's evolution in this special Petri net, e.g., cases of, condition/event arcs, reconfigurations,.. etc.

We consider that the reverse of a non marked R-TNCES $RTN(B_{RTN}, R_{RTN})$ is an imaginary non marked R-TNCES given by

$$RTN^{-1}(B_{RTN}^{-1}, R_{RTN}^{-1})$$

where,

- B_{RTN}^{-1} is a set of reversed non marked TNCESs generated from original non marked TNCESs in B_{RTN} by using arcs inversion generic algorithm and reversed firing rules as in Table 2,
- R_{RTN}^{-1} is a set of reversed reconfiguration rules that are generated from original ones in R_{RTN} using Tables 3 and 4.

Table 2: R-TNCESs reversed firing rules.

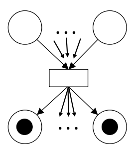
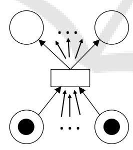
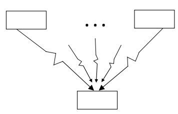
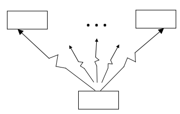
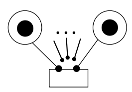
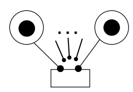
Arcs	RTN	RTN^{-1}
Flow		
Event		
Condition		

Table 3: Reconfiguration rules inversion.

r	RTN	RTN^{-1}
$cond$	c	c^{-1}
S	S	S^{-1}
X	$TN(^{\bullet}r) \rightarrow TN(r^{\bullet})$	$TN(r^{\bullet}) \rightarrow TN(^{\bullet}r)$

Table 4: S^{-1} : Reversed structure modification instructions.

N°	RTN: S	RTN^{-1}: S^{-1}
1	$Cr(cn(x, y))$	$De(cn(x, y))$
2	$Cr(ev(y, y))$	$De(ev(y, y))$
3	$Cr(\mathbb{CC})$	$De(\mathbb{CC})$
4	$De(cn(x, y))$	$Cr(cn(x, y))$
5	$De(ev(y, y))$	$Cr(ev(y, y))$
6	$De(\mathbb{CC})$	$Cr(\mathbb{CC})$
7	$Cr(x, m(x))$	$De(x)$
8	$Cr(y)$	$De(y)$
9	$Cr(fa(x, y))$ / $Cr(fa(y, x))$	$De(fa(x, y))$ / $De(fa(y, x))$
10	$De(x)$	$Cr(x, 1)$ or $Cr(x, 0)$
11	$De(y)$	$Cr(y)$
12	$De(fa(x, y))$ / $De(fa(y, x))$	$Cr(fa(x, y))$ / $Cr(fa(y, x))$
13	$Mo(AND(y))$	$Mo(Or(y))$
14	$Mo(OR(y))$	$Mo(And(y))$

2.4 *Mu* Method

As reported in (Hafidi et al., 2018), Mu function improves the generation of accessibility graphs by reducing redundancies and unnecessary computations. Let $RS(B_{RS}, R_{RS})$ be an R-TNCES, where (1) $B_{RS} = \{C_1, ..., C_n\}$ is the behavior module containing $n > 1$ configurations C_1, .., C_n, and (2) R_{RTN} is the control module containing $k > 1$ reconfiguration rules: r_{ij}, $1 \leq i, j \leq n$ that transforms the system from configuration C_i to configuration C_j. $\mu(AG(C_i), r_{ij})$ is the function that takes the accessibility graph of a configuration $AG(C_i)$ and transforms it into another accessibility graph of another configuration $AG(C_j)$ according to the structure-modifications in the applied reconfiguration rule r_{ij}, i.e., $r_{ij}.s$ is a list containing one or more structure-modification instructions defined in Table 1. Function Mu, generates new accessibility graphs of new configurations from already generated ones. Rather than computing each graph from zero, Mu helps to avoid repetitive computation and keep similar already computed parts of the state space. Mu function uses a set of rewriting rules on an already computed graph to transform it to a new graph. Table 5 presents some rewriting rules of Mu function. Other rewriting rules of all possible reconfiguration scenarios are presented and explained in (Hafidi et al., 2018). A set of rewriting rules for each possible structure-modification instruction $SMI_m \in r_{ij}.s$, i.e., SMI_m denotes the structure-modification instruction symbol number m. We denote by (1) a and a': accessibility graph edges, (2) y, y_1, and y_2: R-TNCESs transitions, (3) $y_1 \backsim y_2$ an event signal from y_1 to y_2, (4) $enb(s, y)$ a boolean function that returns 1 if the transition t is enabled in the state s or 0 otherwise, (5) src: $A \rightarrow S$ the function that returns the state representing the source node of the edge e and tg: $A \rightarrow S$ the function that returns the state representing the target node of the edge e, and (6) $SimulateFrom(s)$ the function that continues the simulation from a non-complete graph (i.e., a set of states and a set of edges), eventual enabled transitions are fired to compute the additional reachable states on the new structure, starting from the state s.

3 METHODOLOGY

This section presents: our motivation in this paper, new proposed backward reachability methodology, algorithm and complexity.

3.1 Motivation

Model-based diagnosis (MBD) of systems (Hamscher et al., 1992) has attracted many interest since it ensures systems safety (Berghout and Bennoui, 2019; Bennoui et al., 2009; Liu et al., 2016; Bhandari et al., 2018). Some of diagnosis abilities is explaining the appearance of an observed system's misbehavior, determining the faulty components of the system, and defining what additional information need to be gathered to identify faulty components (De Kleer and Kurien, 2003). Backward reachability analysis is very important in model based diagnosis, i.e., it represents the principal function that backtracks the system process. Unfortunately, BRA is a complex function that is expensive in terms of computing time and memory. One of BRA high complexity reasons is that it generates branches of all possible systems. BRA function is important in complex systems diagnosis and it deserves to be improved.

Despite its long success in systems diagnosis, BRA has a number of problems in use such as

1. Consideration of reconfigurable systems: the proposed algorithms in literature lacks from the consideration of some complex systems like reconfigurable ones. Contrarily to non-reconfigurable systems, reconfigurable ones have their own special dynamic behavior that needs to be particularly considered when they are backtracked.
2. Improvement of required time/memory: less research interests focus on optimizing the backward reachability function. Such an expensive function needs to include some optimization technique to improve required time and memory. This is beneficial because it makes backward reachability analysis easy and possible for complex systems such as reconfigurable ones.

Petri nets (Murata, 1989) and their extensions are ones of the most widely used formalisms (Khawla and Molnár, 2018) that have been extensively exploited for modeling and analyzing concurrent, parallel and dynamic system. In this paper, we address the problem of reconfigurable systems backward reachability using Petri nets extension called R-TNCESs formalism (Zhang et al., 2017b; Zhang et al., 2013; Zhang et al., 2015; Hafidi et al., 2018).

3.2 Backward Reachability with *Mu* Method

In this subsection, we propose a new methodology for an efficient verification of reconfigurable systems. Foremost, we use a non marked R-TNCESs formalism for modeling reconfigurable systems. Then, specify as R-TNCESs states the set of system's situation(s) to be checked. Systems situations may represent undesirable states such as failures, or desirable ones such as required results. Therefore, situations are defined according to the problem and the type of the studied system (i.e., a detailed example that explains that on Subsection 4.1). The suggested method in this paper uses the proposed backward reachability analysis method to generate the backward accessibility graph of the initial configuration. Then, it uses *Mu* method (Hafidi et al., 2018) to improve the computation of other backward accessibility graphs.

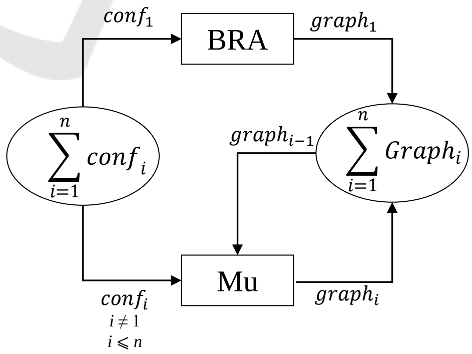


Figure 3: BRA with Mu (the proposed methodology).

The proposed methodology represents a combination between *Mu* method and the suggested backward reachability analysis of R-TNCESs to generate backward reachability graphs. Let us have a reconfigurable system with n configurations such that $n \in \mathbb{N}$ and $n > 1$. The proposed method, as depicted in Fig. 3, uses the proposed BRA for R-TNCESs to compute

Table 5: Mu Rules.

m	SMI_m	Rewriting rules on accessibility graphs	Comments
(1)	$Cr(cn(x,y))$	a) $\forall a \in A$, $Label(a) = y \wedge \neg enb(src(a), y) ::=$ $A \leftarrow A \setminus \{a\}$.	a) For each edge labeled by y: if y is not enabled, then delete it.
(2)	$Cr(ev(y_1,y_2))$	a) $\forall a \in A$, $Label(a) = y_2 ::= A \leftarrow A \setminus \{a\}$. b) $\forall a \in A$, $Label(a) = y_1 \wedge$ $enb(src(a), y_1 \backsim y_2)::=$ $A \leftarrow A \setminus \{a\} \cup \{a'\} \wedge Label(a') = y_1 \backsim y_2 \wedge$ $src(a') = src(a) \wedge tg(a') = src(a) \xrightarrow{y_1 \backsim y_2}$.	a) Delete all edges labeled by y_2. b) For each edge labeled by y_1, check from its source state if $y_1 \backsim y_2$ is enabled, then delete the edge labeled by $y_1$and add a new edge labeled by $y_1 \backsim y_2$.
(3)	$De(cn(x,y))$	a) $\forall s \in S$, $enb(s, y) ::= SimulateFrom(s)$.	a) In each state: if y is enabled, then continue simulation from this state.
(4)	$De(ev(y_1,y_2))$	a) $\forall a \in A$, $Label(a) = (y_1 \backsim y_2) ::= A \leftarrow A \setminus \{a\}$ b) $\forall s \in S$, $enb(s, y_1) ::= SimulateFrom(s)$. c) $\forall s \in S$, $enb(s, y_2) ::= SimulateFrom(s)$.	a) Delete all edges labeled by $y_1 \backsim y_2$. b) In each state if y_1 is enabled, then continue the simulation from this state. c) In each state if y_2 is enabled, then continue the simulation from this state.

backward accessibility graph $graph_1$ of initial configuration $conf_1$. After that, it employs *Mu* method to generate other graphs of the other configurations. Old methods as explained in Fig. 4 should generate all graphs using BRA algorithm. Therefore, the difference between the proposed and the old methods is that the suggested one generates only one graph. Other graphs are generated from the initial one, and then, graph from another until the end of all system's configurations. However, in old methods, each graph is generated independently from others. In addition, *Mu* method is used previously in (Hafidi et al., 2018) with forward reachability analysis methods to generate forward reachability graphs. However, in this paper, Mu method is used with the proposed backward reachability analysis method to generate backward reachability graphs. This combination between both methods allows in one hand to backward analyze systems under reconfigurability constraints, and in another hand, to improve time and memory while generating all the graphs of such complex systems.

Figure 4: BRA without Mu (old methods).

3.3 Algorithm and Complexity

Algorithm 1 describes the proposed method of R-TNCES backward reachability analysis. The algorithm takes as inputs (1) *RT* a non marked R-TNCES structure, (2) *Configurations* a set of TNCESs structures describing system's configurations, (3) *Reconfigurations* a set of *Rules* describing system's transformations, (4) $Conf_0$ a non marked TNCES structure describing the initial configuration of the system, and gives as output *Graphs* the set of accessibility graphs of all the system.

Algorithm 1 uses some additional functions (1) *BRA* function that takes the initial configuration as input and returns its graph using the backward reachability analysis method described in Subsection.2.2. (2) *AdaptingModel* function that adapts *RT* so that *Mu* function, which was proposed for forward analysis, can be applied within the current backward analysis. (3) *GetGraphsWithMu* function that computes other graphs using *Mu*. *GetGraphsWithMu* function as described in Algorithm 2, takes the same inputs as in Algorithm 1, besides the initial accessibility graph $Graph_0$ that was already computed using *BRA* method. The algorithm uses *connections* function to get the set of next reachable configurations from the graph of the current one. After that it recursively computes each new graph from the previous one and stops when (1) no next configurations are reachable, i.e., $SetC = Nil$, or (2) the graph is already computed, i.e, $Graph_i \in Graphs$.

Algorithm 1: GenerateGraphs.

Input: $RT(Configurations$: Set of $TNCESs$; $Reconfigurations$: Set of $Rules) : R-TNCES$; $Conf_0$: $TNCES$;
Output: $Graphs$: Set of $Accessibility$ $Graphs$;

$Graph_0 = BRA(Conf_0)$;
$AdaptingModel(RT, Conf_0, Graph_0)$;
$Graphs = GetGraphsWithMu(RT, Conf_0, Graph_0)$;
$Graphs \leftarrow Graph_0 \cup Graphs$;

Algorithm 2: GetGraphsWithMu.

Input: $RT(Configurations$: Set of $TNCESs$; $reconfigurations$: Set of $Rules) : R-TNCES$; $Conf_0$: $TNCES$; $Graph_0$: Accessibility Graph;
Output: $Graphs$: Set of $Accessibility$ $Graphs$;
Variables: $SetC$: Set of $TNCESs$;

$SetC \leftarrow connections(Graph_0)$;
if $SetC \neq Nil$ **then**
    foreach $Conf_i \in SetC$ **do**
        $Graph_i = Mu(conf_i , conf_0, Graph_0)$;
        if $graph_i \notin Graphs$ **then**
            $Graphs \leftarrow graph_i \cup Graphs$;
            $GetGraphsWithMu(RT, Conf_i, Graph_i)$;
        end
    end
end

The time complexity of the entire algorithm: Algorithm 1 in systems with at least 2 configurations is computed as follows

$$O(1 * e^m + (\mid Configurations \mid -1) * n)$$

where, (1) $O(e^m)$ is complexity of the *BRA* function used only once for computing the graph of the initial configuration, and (2) $O(n)$ is complexity of *Mu* function (Hafidi et al., 2018) used to compute other accessibility graphs, i.e., $(\mid Configurations \mid -1)$ times in the worst case when all configurations are reachable.

4 EXPERIMENTATION

This section is composed of two subsections (1) a case study where paper's contributions are applied, and (2) performance evaluation where proposed and related methodologies are compared using different factors.

4.1 Case Study: FESTO MPS

FESTO MPS is a benchmark production system used as real case study to apply paper's contribution. In next subsections, we first describe FESTO MPS working process, then, formally model the system using a non marked R-TNCESs, and finally, verify it using the proposed method. The same case study was studied in (Hafidi et al., 2018; Zhang et al., 2013), however in this paper, for model-based diagnosis purposes, we use the suggested backward reachability method to generate accessibility graphs rather than the forward one that was used the previous paper. In the end of this section, both case studies will be compared to conclude benefits and limits of each method.

4.1.1 Working Process

FESTO MPS is composed of three main functions: distributing, testing, and processing peace-works. In this case study, the system contains two drilling machines $Driller_1$ and $Driller_2$, and works in three possible production modes: $High$, $Light_1$, and $Light_2$. First, the system is in $High$ production mode, where both drilling machines work simultaneously. In some cases, FESTO MPS changes its behavior and switches from a mode to another, i.e., when $Driller_1$ (respectively, $Driller_2$) breaks down, the system switches to $Light_2$ (respectively, $Light_1$) production mode where only $Driller_2$ (respectively, $Driller_1$) works. This happens in order to prevent system from malfunctions when partial hardware failures occur (i.e., a component breaks down) or to respond to external instructions (i.e., user requirements). FESTO MPS main working process is explained in Fig. 5

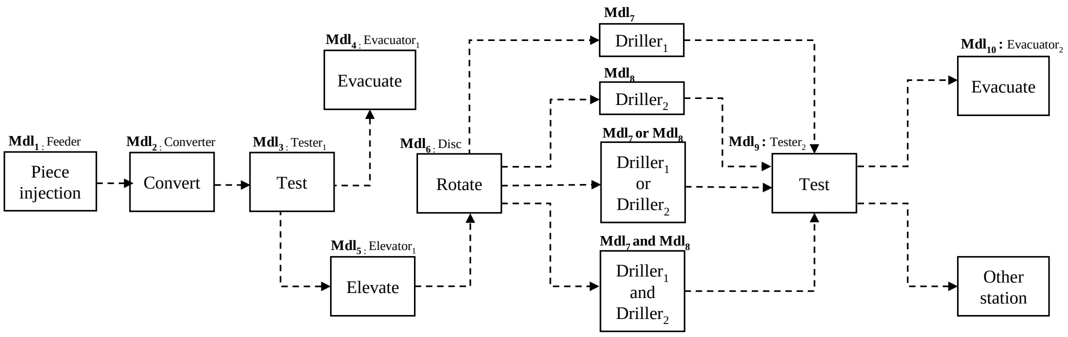


Figure 5: FESTO MPS main process. (Hafidi et al., 2018)

4.1.2 System Encoding

In order to apply formal analysis techniques, it is necessary to mathematically model the studied system FESTO MPS. We model FESTO MPS using R-TNCES formalism already presented in Subsection 2.1. FESTO MPS is an R-TNCES

$$FT(B_{FT}, R_{FT})$$

where,

- $B_{FT} = \{c_1, c_2, c_3\}$: is the behavior module that contains all possible configurations, i.e., FESTO MPS production modes are represented by R-TNCESs configurations, where C_1, C_2, and C_3 configurations respectively represent *High*, $Light_1$, and $Light_2$ production modes. Each configuration is presented by a set of interconnected modules (Mdl_i) which are control components communicating with signals.
- $R_{FT} = \{r_{c_1-c_2}, r_{c_1-c_3}, r_{c_2-c_1}, r_{c_3-c_1}\}$ is the control module that involves all reconfiguration rules that transforms the system from a configuration to another.

The initial configuration c_1 of the studied system is represented by the TNCES that is graphically shown in Fig. 6. Other configurations c_2 and c_3 can be obtained by applying possible reconfiguration rules from R_{FT}.

Considered reconfiguration rules are described as follows,

- $r_{c_1-c_2} = (Driller_2$ breaks down; $De(Mdl_8)$, $De(t_{20})$, $De(p_{11})$; $(p_1, C_1) \rightarrow (p_1, C_2))$;
- $r_{c_1-c_3} = (Driller_1$ breaks down; $De(Mdl_7)$, $De(t_{18})$, $De(p_{10})$; $(p_1, C_1) \rightarrow (p_1, C_3))$.

4.1.3 System Verification

We define a set of goal states from which we start backward reachability: $\{goal_1 = (S_4, C_1), goal_2 = (S_4, C_2)\}$. $goal_1$ and $goal_2$ are undesirable states that represents tests failure in configurations C_1 and C_2, respectively.

We backtrack the system using the proposed R-TNCESs reverse method, and obtain

$$FT^{-1}(B_{FT}^{-1}, R_{FT}^{-1})$$

where,

- $B_{FT}^{-1} = \{C_1^{-1}, C_2^{-1}, C_3^{-1}\}$, i.e., obtained using R-TNCESs reversed firing rules 2 in each configuration,
- $R_{FT}^{-1} = \{r_{c_1-c_2}^{-1}, r_{c_1-c_3}^{-1}, r_{c_2-c_1}^{-1}, r_{c_3-c_1}^{-1}\}$, i.e., obtained using Tables 3 and 3.

The set of considered FT^{-1} reconfiguration rules are described as follows,

- $r_{c_1-c_2}^{-1} = (Driller_2$ works; $Cr(Mdl_8)$, $Cr(t_{20})$, $Cr(p_{11})$; $(p_1, C_2) \rightarrow (p_1, C_1))$;
- $r_{c_1-c_3}^{-1} = (Driller_1$ works; $Cr(Mdl_7)$, $Cr(t_{18})$, $Cr(p_{10})$; $(p_1, C_1) \rightarrow (p_1, C_3))$.

Now, we compute backward reachability graphs starting from undesirable states $goal_1$ and $goal_2$. Obtained state space is a set of sub-graphs $\{subG(C_1), subG(C_2)\}$ from whole system accessibility graphs. sub-graph $subG(C_1)$ (respectively, $subG(C_2)$) contains branches leading to the undesirable state $goal_1$ (respectively, $goal_2$) in C_1 (respectively, C_2). In real, obtained branches correspond to the critical executions that the system may pass by during its working process. The sub-graph $subG(C_2)$ is depicted in Fig. 7. The advantage of using backward reachability, is that if focuses on explaining the appearance of an undesirable behavior $goal_1$ and $goal_2$, i.e., other behavior of system is not included in the verification. By using the proposed methodology, we were able to successfully apply backward reachability analysis for the studied reconfigurable system using R-TNCESs formalism.

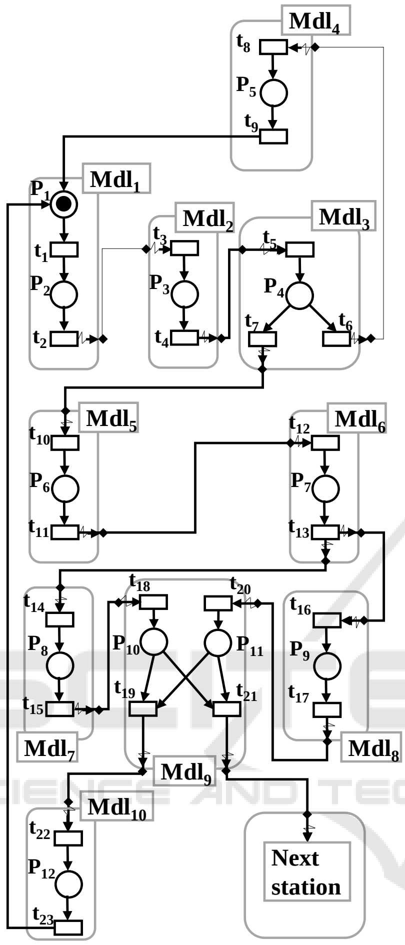


Figure 6: Initial configuration of FESTO MPS C_1. (Hafidi et al., 2018).

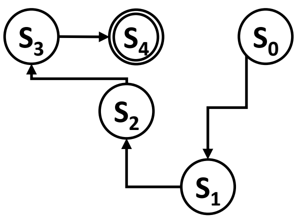


Figure 7: Backward reachability graph of $goal_2$: $subG(C_2)$.

4.2 Performance Evaluation

In this section, we first study results obtained for the same case study using different methodologies. Then, we study the evaluation in large scale systems using different factors. Finally, we summarize in a comparison table limits and benefits of the proposed method and previous related ones.

4.2.1 Case Study

In this subsection, we compare obtained results by the paper's case study with those of the previous work in (Hafidi et al., 2018).

We notice that the total number of computed states is almost the half in current methodology compared to previous ones. Backward reachability helped to identify only critical scenarios and their related states rather than all possible system's behavior. This can serve the verification of systems with complex behavior using less time and memory.

Table 6: Number of states current case study VS previous case study.

Configuration	Number of states	
	previous works (Hafidi et al., 2018)	current work
C_1	10	5
C_2	10	5
C_3	10	4
Total	30	14

4.2.2 Number of Computed States vs. Number of Undesirable States

In this subsection, we apply proposed and related methodologies in a large scale system using different number of undesirable states. The curve depicted in Fig. 8 shows that the number of computed states using the proposed methodology is less than the number of states generated using related methodology. In the best cases, backward reachability generates less states starting from the undesirable states to the source (possible initial marking), however, forward methods generate all possible branches with all possible initial markings.

5 CONCLUSION

This research work deals with the backward reachability of reconfigurable systems such as reconfigurable discrete event control systems RDECSs. The proposed method allows the applicability of backward reachability methods on reconfigurable systems modeled by R-TNCESs. The suggested methodology allows to compute backward reachability graphs using improvement methods that reduce repetitive computations.

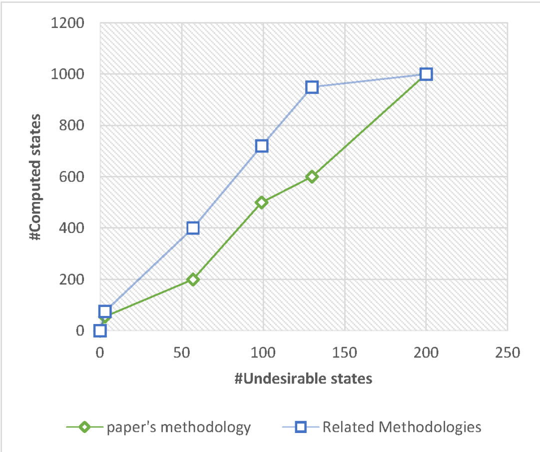


Figure 8: Computed states VS undesirable states.

According to the case study, and performance evaluation, it is shown that backward reachability becomes possible. In addition, the proposed methodology for RDECSs improved verification.

Future works will: (1) involve comparison with tools and methods that use different models, (2) consider probabilistic constraints in computing branches, and (3) include the proposed improvement method in a tool in order to automatize it and profit from its gain.

REFERENCES

Aissa, Y. B., Bachir, A., Khalgui, M., Koubaa, A., Li, Z., and Qu, T. (2019). On feasibility of multichannel reconfigurable wireless sensor networks under real-time and energy constraints. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

Aissa, Y. B., Mosbahi, O., Khalgui, M., and Bachir, A. (2018). New scheduling mechanism in multi-channel reconfigurable wsn under qos and energy constraints. In *32nd annual European Simulation and Modelling Conference 2018*, pages 187–191.

Anglano, C. and Portinale, L. (1994). Bw analysis: a backward reachability analysis for diagnostic problem solving suitable to parallel implementation. In *International Conference on Application and Theory of Petri Nets*, pages 39–58. Springer.

Baier, C., Katoen, J., and Larsen, K. (2008). *Principles of Model Checking*. MIT Press.

Bennoui, H., Chaoui, A., and Barkaoui, K. (2009). Distributed causal model-based diagnosis based on interacting behavioral petri nets. pages 99–106.

Berghout, Y. M. and Bennoui, H. (2019). Distributed diagnosis based on distributed probability propagation nets. *International Journal of Computational Science and Engineering*, 18(1):72–79.

Bhandari, G. P., Gupta, R., and Upadhyay, S. K. (2018). Colored petri nets based fault diagnosis in service oriented architecture. *International Journal of Web Services Research (IJWSR)*, 15(4):1–28.

Cong, X., Fanti, M. P., Mangini, A. M., and Li, Z. (2017). Decentralized diagnosis by Petri nets and integer linear programming. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

De Kleer, J. and Kurien, J. (2003). Fundamentals of model-based diagnosis. *IFAC Proceedings Volumes*, 36(5):25–36.

Dubinin, V., Vyatkin, V., and Hanisch, H.-M. (2015). Synthesis of safety controllers for distributed automation systems on the basis of reverse safe net condition/event systems. In *Proc. Trustcom/BigDataSE/ISPA*, volume 3, pages 287–292. IEEE.

Gharsellaoui, H., Gharbi, A., Khalgui, M., and Ahmed, S. B. (2012). Feasible automatic reconfigurations of real-time os tasks. In *Handbook of Research on Industrial Informatics and Manufacturing Intelligence: Innovations and Solutions*, pages 390–414. IGI Global.

Ghribi, I., Abdallah, R. B., Khalgui, M., Li, Z., Alnowibet, K., and Platzner, M. (2018). R-codesign: Codesign methodology for real-time reconfigurable embedded systems under energy constraints. *IEEE Access*, 6:14078–14092.

Hafidi, Y., Kahloul, L., Khalgui, M., Li, Z., Alnowibet, K., and Qu, T. (2018). On methodology for the verification of reconfigurable timed net condition/event systems. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, (99):1–15.

Hafidi., Y., Kahloul., L., Khalgui., M., and Ramdani., M. (2019). On improved verification of reconfigurable real-time systems. In *Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering - Volume 1: ENASE,*, pages 394–401. INSTICC, SciTePress.

Hamscher, W., Console, L., and de Kleer, J., editors (1992). *Readings in Model-based Diagnosis*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Khalgui, M. and Mosbahi, O. (2010). Intelligent distributed control systems. *Information and Software Technology*, 52(12):1259–1271.

Khalgui, M., Mosbahi, O., Li, Z., and Hanisch, H. M. (2011). Reconfigurable multiagent embedded control systems: From modeling to implementation. *IEEE Transactions on Computers*, 60(4):538–551.

Khalgui, M., Rebeuf, X., and Simonot-Lion, F. (2004). A behavior model for iec 61499 function blocks. In *Proceedings of the 3rd Workshop on Modelling of Objects, Components, and Agents*, pages 71–88.

Khalgui, M., Rebeuf, X., and Simonot-Lion, F. (2007). A deployment method of component based applications on distributed industrial control systems. *European Jonal of Automated Systems*, 41(6):707–732.

Khawla, B. and Molnár, B. (2018). An fsm approach for hypergraph extraction based on business process modeling. In *International Conference on Computer Science and its Applications*, pages 158–168. Springer.

Koszewnik, A., Nartowicz, T., and Pawluszewicz, E. (2016). Fractional order controller to control pump in FESTO MPS PA compact workstation. In *Proc. 17th International Carpathian Control Conference (ICCC)*, pages 364–367.

Lakhdhar, W., Mzid, R., Khalgui, M., Li, Z., Frey, G., and Al-Ahmari, A. (2018). Multiobjective optimization approach for a portable development of reconfigurable real-time systems: From specification to implementation. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*.

Leveson, N. G. and Stolzy, J. L. (1985). Analyzing safety and fault tolerance using time petri nets. In *International Joint Conference on Theory and Practice of Software Development*, pages 339–355. Springer.

Liu, B., Ghazel, M., and Toguyéni, A. (2016). Model-based diagnosis of multi-track level crossing plants. *IEEE Transactions on Intelligent Transportation Systems*, 17(2):546–556.

Murata, T. (1989). Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580.

Naidji, I., Smida, M. B., Khalgui, M., and Bachir, A. (2018). Non cooperative game theoretic approach for residential energy management in smart grid. In *The 32nd Annual European Simulation and Modelling Conference*, pages 164–170, Ghent, Belgium.

Pózna, A. I., Gerzson, M., Leitold, A., and Hangos, K. (2016). Colored petri net based diagnosis of process systems.

Qin, M., Li, Z., Zhou, M., Khalgui, M., and Mosbahi, O. (2012). Deadlock prevention for a class of petri nets with uncontrollable and unobservable transitions. *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, 42(3):727–738.

Ramdani, M., Kahloul, L., and Khalgui, M. (2018). Automatic properties classification approach for guiding the verification of complex reconfigurable systems. In *Proc. Proceedings of the 13th International Conference on Software Technologies - Volume 1: ICSOFT,*, pages 591–598. INSTICC, SciTePress.

Ramdani., M., Kahloul., L., Khalgui., M., and Hafidi., Y. (2019). R-tnces rebuilding: A new method of ctl model update for reconfigurable systems. In *Proceedings of the 14th International Conference on Evaluation of Novel Approaches to Software Engineering - Volume 1: ENASE,*, pages 159–168. INSTICC, SciTePress.

Wang, X., Khalgui, M., and Li, Z. (2011). Dynamic low power reconfigurations of real-time embedded systems. In *PECCS*, pages 415–420.

Zhang, J., Frey, G., Al-Ahmari, A., Qu, T., Wu, N., and Li, Z. (2017a). Analysis and control of dynamic reconfiguration processes of manufacturing systems. *IEEE Access*.

Zhang, J., Khalgui, M., Li, Z., Frey, G., Mosbahi, O., and Salah, H. B. (2015). Reconfigurable coordination of distributed discrete event control systems. *IEEE Trans. Contr. Sys. Techn.*, 23(1):323–330.

Zhang, J., Khalgui, M., Li, Z., Mosbahi, O., and Al-Ahmari, A. (2013). R-TNCES: A novel formalism for reconfigurable discrete event control systems. *IEEE Trans. Systems, Man, and Cybernetics: Systems*, 43(4):757–772.

Zhang, S., Wu, N., Li, Z., Qu, T., and Li, C. (2017b). Petri net-based approach to short-term scheduling of crude oil operations with less tank requirement. *Information Sciences*, 417:247–261.