Temporal Analysis of Influence of Resource Failures on Cyber-Physical Systems Based on Discrete Timed Petri Nets

Fu-Shiung Hsieh

Department of Computer Science and Information Engineering, Chaoyang University of Technology, Taichung 41349, Taiwan; fshiueh@cyut.edu.tw; Tel.: +886-2332-3000 (ext. 4759)

Abstract: Advancement of IoT and ICT provide infrastructure to manage, monitor and control Cyber-Physical Systems (CPS) through timely provision of real-time information from the shop floor. Although real-time information in CPS such as resource failures can be detected based on IoT and ICT, improper response to resource failures may cripple CPS and degrade performance. Effective operations of CPS relies on an effective scheme to evaluate the impact of resource failures, support decision making needed and take proper actions to respond to resource failures. This motivates us to develop a methodology to assess the impact of resource failures on operations of CPS and provide the decision support as needed. The goal of this study is to propose solution algorithms to analyze robustness of CPS with respect to resource failures in terms of the impact on temporal properties. Given CPS modeled by a class of discrete timed Petri nets (DTPNs), we develop theory to analyze robustness of CPS by transforming the models to residual spatial-temporal network (RSTN) models in which capacity loss due to resources is reflected. We formulate an optimization problem to determine the influence of resource failures on CPS based on RSTNs and analyze the feasibility to meet the order deadline. To study the feasibility to solve a real problem, we analyze the computational complexity of the proposed algorithms. We illustrate the proposed method by application scenarios. We conduct experiments to study efficiency and verify computational feasibility of the proposed method to solve a real problem.

Keywords: cyber-physical systems; IoT; optimization; resource failure; discrete timed; petri net

1. Introduction

The advancement of information and communications technology (ICT) and Internet of Things (IoT) [1] provides the infrastructure for Cyber-Physical Systems (CPS) [2,3] to manage, monitor and control manufacturing systems. ICT refers to technologies involving any kind of computing/communication devices, networking components and information systems that enable efficient interaction in the digital world, while IoT can be defined as the network of physical objects embedded with sensors, software and relevant ICT technologies for interacting with other objects/systems over the Internet. Although CPS provides the potential visibility and a paradigm for enterprises to respond to dynamic changing environments based on ICT and IoT, these advantages also pose challenges in modeling, design and implementation of CPS [4]. In particular, robust design of CPS has become an important issue to achieve stable and secure CPS and is attracting a lot of attention in the CPS research community. A comprehensive survey of the concept and strategies for robust design of CPS can be found in [5]. In CPS for manufacturing systems, failures of machines/resources are unavoidable. Managers often
wonder and worry about whether the original due dates of orders can still be met when failures of machines/resources occur, i.e., whether the manufacturing system is robust with respect to the failures. Providing a decision support tool for managers to assess the impact of failures of machines/resources on the orders is an important issue.

In terms of the robustness design requirements of CPS, the real-time information and communication infrastructure provided by IoT and ICT alone cannot ensure effective operations and robustness of CPS. Robustness of CPS can be achieved only if the real-time information provided by IoT and ICT are timely and properly used to support decision making and respond to the changing environment. For example, real-time information from the shop floor—such as resource failures [6]—can be detected in CPS. However, real-time resource failures information from IoT and ICT is useful only if an effective scheme to evaluate the impact of, and deal with, resource failures is available. In the literature, design issues of CPS include robustness, security and cost effectiveness [7]. Study on the impact of resource failures on CPS is less explored. This motivates us to develop a methodology to assess the impact of resource failures on operations and temporal properties of CPS. The goal of this study is to propose a method for modeling and analysis of CPS, in order to pave the way for developing an effective strategy to handle resource failures in CPS.

To evaluate the impact of resource failures on CPS, we will study the robustness of CPS in the context of manufacturing systems in this paper. The robustness property is concerned with the influence of resource failures on the operations and performance of CPS. In make-to-order manufacturing, meeting the deadline of orders is an important issue. Resource failures often cripple the operations of manufacturing systems and may lead to delays in fulfillment of orders. Therefore, the robustness property of CPS with respect to resource failures to be studied in this paper is concerned with how the orders will be influenced due to resource failures. In classical control theory, robustness addresses the issue about whether the desired system property can be maintained in the presence of uncertainty. The problem to be studied in this paper is to investigate the effects of resource failures on operations of CPS and the fulfillment of an order with multiple product demands and deadlines. The problem setting of this paper is different from that of [8], in that multiple types of product demands are considered in this study whereas the work of [8] considers a single type of product demands. The objective of this study is to propose a solution algorithm to analyze robustness of CPS with respect to resource failures, based on a class of discrete timed Petri net (DTPN) model that extends the one in [8].

The approach to robustness analysis of the DTPN models in this study is different from the existing reachability analysis methods that rely on abstraction of state space of timed Petri nets [9,10]. Our approach is based on the concept of model transformation in Model Driven approach (MDA) [11], which transforms DTPNs into spatial-temporal networks (STNs) to enable the compact representation and efficient search for solutions based on mathematical programming/optimization tools. As this study focuses on assessing the impact of resource failures on temporary properties based on model transformation, this study is different from [12,13], which focus on robust control and scheduling based on timed, extended reachability graphs. This paper is different from [8,14] as it focuses on the robustness property of CPS. In addition, the problem studied in this paper considers order requirements with multiple types of product demands instead of a single type of product demands. This paper is different from [15] as it includes important properties, algorithms, rigorous proof of properties of CPS, complexity analysis and verification of complexity analysis results by experiments. In summary, the contributions of this paper include: (1) proposing a formal timed model to capture resource failures in CPS, (2) formulating a feasibility problem for nominal CPS and a robustness analysis problem to analyze the impact of resource failures on temporal properties, (3) developing algorithms for the problems based on model transformation and (4) analyzing complex-
ity of the proposed algorithms and verifying the computational feasibility of the proposed approach.

The structure of this paper is as follows. A literature review and the robustness analysis problems in CPS will be introduced in Sections 2 and 3, respectively. In Section 4, construction of STNs and analysis of the nominal CPS based on STNs will be presented. In Section 5, the impact of resource failures on CPS will be analyzed. In Section 6, we will illustrate the proposed method by examples and present the results. We will discuss the results in Section 7 and conclude this paper in Section 8.

2. Literature Review

CPS consist of two parts: the cyber space and the physical space. Control of entities in physical space of CPS is based on the cyber space model. To analyze CPS, proper modeling tools to construct the cyber space model of the CPS are relied upon. A modeling tool must be able to capture concurrent, synchronous and asynchronous operations/events in CPS and must support construction, edition and representation of the model in industrial standard format. A variety of modeling approaches/tools can be used to capture the operations of CPS by constructing the cyber space model of CPS, e.g., Discrete Event System [16], Extended Hybrid Automata [17] and Petri nets (PN) [18]. Among these tools, Petri nets [19] provide an easy-to-use graphical user interface that can capture concurrent, synchronous and asynchronous operations/events in CPS. Petri nets can be represented by the Petri Net Markup Language (PNML) [20], an XML-based interchange format widely supported by many software tools [21]. Therefore, we adopt Petri nets in this paper to model CPS. Despite the variety of Petri net tools such as the ones in [22,23] available for modeling CPS, these tools do not provide a method for analyzing the influence of resource failures on performance.

As the objective of this paper is to analyze the influence of resource failures on temporal property of CPS, Petri nets supporting modeling of time are adopted. In the literature, some Petri net models assign time delay to transitions while others assign time delay to places in the nets [24]. A timed Petri net is called deterministic timed Petri net if the time delay is deterministic [25–29]. A timed Petri net is called stochastic timed Petri net if the time delay is probabilistically specified [30–33]. As this paper focuses on influence of resource failures on temporal property of CPS, a class of deterministic discrete timed Petri nets are adopted. Different models have been proposed to capture resource failures. Some of the works (e.g., [34,35]) model resource failures as loss of tokens in untimed Petri nets models and develop methods to assess the impact of resource failures on the system after the occurrence of failures. The works [36,37] classify the resources into reliable resources and unreliable resources based on a class of untimed Petri nets (S4PR) and develop methods to allocate resources. The works [12,13] consider uncertainties due to the interruptions of operations and unreliable resources and take performance and risk into account in the objective function to deal with uncertainties in manufacturing systems based on timed Petri nets. As the goal of this paper is to develop a method to assess the impact of resource failures on the system after the occurrence of failures, we model resource failures as loss of tokens in discrete timed Petri nets. Although the proposed discrete timed Petri nets can capture the characteristics of CPS, there still lacks a theory to analyze the robustness property. For this reason, this paper is devoted to development of theory to facilitate the analysis of robustness property of CPS based on the proposed discrete timed Petri nets.

To study robustness property of CPS, a nominal solution must be generated first. Robustness analysis for CPS is done based on the nominal solution. To generate a nominal solution, models that can capture the spatial and temporal properties of workflows are constructed. In this paper, we consider order requirements with multiple product demands and a deadline. A spatial-temporal network (STN) model is constructed for each type of production process required by the orders. The STN models are constructed based on transformation of the DTPN models of CPS. The STNs constructed for CPS are
capable of capturing the dynamics of workflows in space and time. The STNs can only capture the nominal operations of CPS. To study the influence of resource failures on CPS, we construct residual spatial-temporal network (RSTN) models in which capacity loss due to resources is reflected and modeled. We analyze the impact of resource failures on CPS based on RSTNs. To optimize the performance after resource failures, we formulate an optimization problem to determine the influence of resource failures on CPS and analyze the feasibility to meet the deadline. The optimization problem can be solved by applying any integer programming solver such as the CPLEX Optimizer [38]. We assess practicality of the proposed method by applying it to application scenarios in which resource failures occur in executing production schedules. The influence of resource failures on the schedules in each scenario can be visually and clearly represented in the residual spatial-temporal networks. To study the feasibility to solve a real problem, we analyze the computational complexity of the proposed algorithm. Our analysis indicates that the lower bound on the complexity of the proposed algorithm is polynomial with respect to the tasks (product demands) and scale of the production processes. We verify the computational feasibility of the proposed method by conducting experiments. The results indicate that the computational time grows polynomially with the quantity of the products influenced by resources failures and the number of transitions in the task subnet.

3. Robustness Analysis Problem Formulation in CPS

The problem to be studied in this paper is to analyze the impact of resource failures on CPS. To formulate the problem to be addressed in this paper, we summarize a list of notations/symbols that will be used in this paper in Table 1. The cyber world model of a CPS is represented by $G$. It is used to capture the dynamics of CPS. The nominal cyber world model is an abstraction of the physical world of CPS. The structure of the nominal cyber world model considered in this paper is similar to the S3PR model and the S4PR model in the literature [39,40] that can capture sequential processes with shared resources, is extended with time factor and a proper uncertainty model to capture failures of resources and the time intervals during which failures occur. Uncertainties such as machine failures may occur in the physical world of CPS and influence the operations of CPS. In the remainder of this paper, we will use $\Delta$ to denote the uncertainty model. Given the nominal cyber world model $G$ and the uncertainty model $\Delta$, the problem is to study the influence of $\Delta$ on the operations of CPS. To analyze the influence of $\Delta$ on the operations of CPS, the nominal cyber world model $G$ and the uncertainty model $\Delta$ must be represented properly. In this paper, the nominal cyber world model $G$ will be presented by a DTPN model. The uncertainty model $\Delta$ will be characterized based on the failed resources as well as the time intervals during which unexpected events occur. In this section, we first introduce construction of the nominal model $G$ for CPS and then present the uncertainty model to formulate the problem.
Table 1. Notations.

| Symbol/Variable | Meaning |
|-----------------|---------|
| \( J \)         | The number of different types of tasks |
| \( J \)         | The set of indices of all different types of tasks, \( J = \{1,2,3,...,J\} \) |
| \( j \)         | A type of task \( j \in J \) |
| \( GJ_j \)      | A task subnet, \( GJ_j = (P_j,T_j,F_j,m_{j0},\mu_j) \) |
| \( N_j \)       | The number of operations in \( GJ_j \) |
| \( R \)         | The number of different types of resources |
| \( R = \{1,2,3,...,R\} \) | The set of different types of resources in the system |
| \( k \)         | The \( k \)-th operation performed by a type \( r \) resource, where \( r \in R \) |
| \( G_r^j \)     | A circuit (a DTPN model) that represents the activity of the \( k \)-th operation in type \( j \) task to be performed by a type \( r \) resource |
| \( \Omega'_j \)  | The set of indices of circuits of type \( r \) resources involved in type \( j \) task, \( \Omega'_j = \{1,2,3,...,K'_j\} \), where \( K'_j \) is the number of circuits involved. |
| \( || \)         | An operator to merge Petri nets |
| \( GR_r \)      | A type \( r \) resource subnet, \( GR_r = (P_r,T_r,F_r,m_{r0},\mu_r) = \bigcup_{j \in J} G_r^j \) |
| \( C_{rt} \)    | The initial capacity (the number) of type \( r \) resources in period \( t \); initially, \( C_{rt} \) is set to \( m_{r0}(r) \) for all \( t \) |
| \( \overline{C}_{rt} \) | The residual capacity of \( C_{rt} \) of type \( r \) resources in period \( t \), initial value of \( \overline{C}_{rt} \) is set to \( C_{rt} \). |
| \( G \)         | A nominal CPS model, \( G = (P,T,F,m_0,\mu) = GJ \bigcup_{r \in R} GR_r \) |
| \( m \)         | A marking of \( G \) |
| \( \delta \)    | A \( |P| \)-dimensional perturbation vector |
| \( \delta_i \)  | The \( i \)-th element of the perturbation vector \( \delta \) |
| \( \omega_{il} \) | A discrete time failure interval, \( \omega_{il} = [\alpha_{il},\beta_{il}] \), where \( i \in \{1,2,..,|P|\} \) and \( l \in \{1,2,..,\delta_i\} \) |
| \( \omega \)     | All failure intervals \( \omega_{il} \) associated with \( \delta \), \( i \in \{1,2,..,|P|\} \) and \( l \in \{1,2,..,\delta_i\} \), |
| \( \Delta(m) \)  | The uncertainty model that captures resource failures for a reachable marking \( m \), \( \Delta(m) = (m,\delta,\omega) \) |
| \( Q_j \)       | The product demands, \( Q_j \), where \( j \in J = \{1,2,3,...,J\} \) |
| \( Q^j_{f} \)   | The quantity of products influenced by resource failures \( \Delta(m) \) |
| \( \theta \)    | The deadline |
| \( Pf \)        | The set of all final state places in \( P \) |
| \( \Pi \)       | The number of periods in the time horizon |
| \( STN_j(V_j,A_j) \) | A spatial-temporal network constructed for type \( j \) task, \( j \in J = \{1,2,3,...,J\} \) |
| \( RSTN_j(V_j,A_j) \) | A residual spatial-temporal network constructed based on \( STN_j(V_j,A_j) \) for \( j \in J \) |
We consider a make-to-order manufacturing system in which an order is specified by the quantity of different types of product demands, $Q_j$, where $j \in J = \{1,2,3,\ldots,J\}$ and the deadline $\theta$. The production process of each type of product is associated with a type of task. Therefore, the number of different types of tasks is $J$. The set of indices of all different types of tasks is denoted by $J = \{1,2,3,\ldots,J\}$ and we refer to a type of task by $j \in J = \{1,2,3,\ldots,J\}$. We define a task subnet to describe the production process of a type of products. A task subnet can be described by a discrete timed Petri net (DTPN). A DTPN is a five-tuple $D = (P,T,F,m_0,\mu)$ in which $P$ is a set of places, $T$ is a set of transitions, $F \subseteq (P \times T) \cup (T \times P)$ is a set of flow relation, $m_0 : P \rightarrow \mathbb{Z}$ is an initial marking, and $\mu : T \rightarrow \mathbb{Z}$ is a function that specifies either a lower bound of the firing time of each transition or the firing time of each transition, where $\mathbb{Z}$ is the set of nonnegative integers.

**Definition 1**: A task subnet is a discrete timed Petri net, $GJ_j = (P_j,T_j,F_j,m_{j0},\mu_j)$, with sequential structure, where $j \in J$ and the function $\mu_j : T_j \rightarrow \mathbb{Z}$ is used to specify the lower bound of the firing time of each transition in $T_j$. For a task subnet $GJ_j = (P_j,T_j,F_j,m_{j0},\mu_j)$ that describes $N_j$ operations in $GJ_j$, there are $2N_j$ transitions in $GJ_j$ as each operation is described by a start transition and an end transition. Therefore, there are $2N_j + 1$ places in $P_j$. We assume that the places in $P_j$ are numbered from upstream to downstream. That is, $P_j = \{p^j_1, p^j_2, p^j_3, \ldots, p^j_{2N_j+1}\}$.

Note that for transition $t \in T_j$, $\mu_j(t)$ only specifies a lower bound of the firing time of transition $t$ by using any resource. Firing a transition relies on the use of a resource. The firing time of transition $t \in T_j$ depends on the type of resource used to fire $t$. The exact firing time of $t$ is determined by the firing time of the same transition specified in the corresponding resource subnet of the resource involved. Therefore, $\mu_j(t)$ can be set to zero for each transition $t \in T_j$ because is always a lower bound of transition $t$ by using any resource. Figure 1 shows two examples of task subnets, $GJ_1$ and $GJ_2$. The firing time of transitions in $GJ_1$ can be set as $\mu_1(t_1) = \mu_1(t_2) = \mu_1(t_3) = \mu_1(t_4) = 0$ if the lower bound of the firing time of each transition in $T_1$ is unknown. Similarly, if the lower bound of the firing time of each transition in $T_2$ is unknown, the firing time of transitions in $GJ_2$ can be set as $\mu_2(t_5) = \mu_2(t_6) = \mu_2(t_7) = \mu_2(t_8) = \mu_2(t_9) = \mu_2(t_{10}) = 0$. 
To facilitate modeling, the operator “$\parallel$” is defined as follows to merge two DTPNs through common transitions, places and arcs.

**Definition 2:** Given two discrete timed PNs, $G_1 = (P_1, T_1, F_1, m_{10}, \mu_1)$ and $G_2 = (P_2, T_2, F_2, m_{20}, \mu_2)$, the operator “$\parallel$” to combine $G_1$ and $G_2$ is defined as follows:

$$G_1 \parallel G_2 = (P, T, F, W, m_0, \mu), \text{ where } P = P_1 \cup P_2, \quad T = T_1 \cup T_2,$$

$$F(p, t) = \begin{cases} F_1(p, t) \text{ if } p \in P_1 \text{ and } t \in T_1, \\ F_2(p, t) \text{ if } p \in P_2 \text{ and } t \in T_2 \end{cases}, \quad F(t, p) = \begin{cases} F_1(t, p) \text{ if } p \in P_1 \text{ and } t \in T_1, \\ F_2(t, p) \text{ if } p \in P_2 \text{ and } t \in T_2 \end{cases},$$

$$m_0(p) = \begin{cases} m_{10}(p) \text{ if } p \in P_1, \\ m_{20}(p) \text{ if } p \in P_2 \end{cases} \quad \text{and} \quad \mu(t) = \begin{cases} \max(\mu_1(t), \mu_2(t)) \text{ if } t \in T_1 \cap T_2, \\ \mu_1(t), \text{ if } t \in T_1 \setminus T_2, \\ \mu_2(t), \text{ if } t \in T_2 \setminus T_1 \end{cases}.$$ 

An operation is performed by a resource. In this study, the number of different types of resources is denoted by $R$. We use $R = \{1, 2, 3, \ldots, R\}$ to denote the set of different types of resources in the system. We use $r$ to refer to a type of resources, where $r \in R$. The $k$-th operation performed by type $r$ resource is represented by a circuit, $G_r^k$, which is a discrete timed PN. Each circuit $G_r^k$ has an idle place $r$, a busy place, one start transition and one end transition. Figure 2a–e shows examples of resource circuits. Let $\Omega_r^k = \{1, 2, \ldots, K_r^k\}$ denote the set of indices of circuits of type $r$ resources involved in type $j$ task. The capabilities of a resource type $r$ is described by a resource subnet defined below.
Examples of a GR cause time (combine a task subnet and the resource subnet. Therefore, after applying the merging operator, the firing time of the transition in the task subnet is just a task subnet and the associated resource subnet. Note that as the firing time for a transition in the resource subnet is set as \( \mu_r(t_3) = 2, \mu_r(t_4) = 0 \). The firing time of transitions in \( GR_n \) is set as \( \mu_n(t_7) = 1, \mu_n(t_8) = 0 \). The firing time of transitions in \( GR_n \) is set as \( \mu_n(t_9) = 1, \mu_n(t_{10}) = 0 \).

Operations in manufacturing systems are performed by resources to process parts. The proposed model captures synchronization between parts and resources by applying the merging operator “\( \mid \)” to a common transition in a task subnet and a resource subnet. The operator “\( \mid \) ” also merges relevant common places and arcs involved between a task subnet and the associated resource subnet. Note that as the firing time for a transition in the task subnet is just a lower bound, which is always smaller than the firing time specified in the resource subnet, the exact time for firing a transition depends on the resource involved. Therefore, after applying the merging operator, the firing time of the merged transition is defined as the maximum of the transition firing time specified in the task subnet and the resource subnet.

Consider a task subnet \( GJ_j = (P_j, T_j, F_j, m_{j0}, \mu_j) \) and a resource subnet \( GR_r = (P_r, T_r, F_r, m_{r0}, \mu_r) \). For the model \( GJ_j \| GR_r \) obtained by applying the “\( \| \) ” operator to combine a task subnet \( GJ_j = (P_j, T_j, F_j, m_{j0}, \mu_j) \) with a resource subnet \( GR_r = (P_r, T_r, F_r, m_{r0}, \mu_r) \), the firing time of transition \( t \in T_j \cap T_r \) in \( GJ_j \| GR_r \) is \( \mu(t) = \max(\mu_j(t), \mu_r(t)) = \mu_r(t) \) as \( \mu_j(t) \) is a lower bound of \( \mu_r(t) \). That is, the firing time \( \mu(t) \) of transition \( t \in T_j \cap T_r \) in \( GJ_j \| GR_r \) is determined by the firing time \( \mu_r(t) \) specified in \( GR_r \). Therefore, \( \mu_j(t) \) can be set to zero for each transition \( t \in T_j \) because \( \mu_j(t) \) is a lower bound of \( \mu_r(t) \).
To capture cooperation and interaction between different types of tasks and resources in the CPS, we merge the task subnets with the resource subnets to construct the discrete timed Petri net model $G$ for the nominal CPS defined $G$ as follows.

**Definition 4:** A nominal model is described by a discrete timed Petri net $G = (P, T, F, m_0, \mu) = GJ \parallel GR$, where $GJ = \bigcup_{j \in J} GJ_j$ and $GR = \prod_{r \in R} GR_r = (P_R, T_R, F_R, m_{R0}, \mu_R)$, where $m_0 : P \rightarrow Z^{|P|}$ and $\mu : T \rightarrow Z^+ \cup \{0\}$. The state of $G$ is called a marking represented by a vector $m \in Z^{|P|}$.

In this paper, it is assumed that the timed Petri net models work under infinite server policy. As we focus on the deterministic and aim to determine the feasible firing sequences to meet the deadline, the firing of transitions is determined by the algorithm. Note that operations of the DTPN models are based on the firing sequences found by the proposed algorithm. Therefore, firing policy and memory policy are not required in the proposed method.

Figure 2 shows an example of $G = (P, T, F, m_0, \mu) = GJ \parallel GR$, where $GJ = GJ_1 \parallel GJ_2$, $GJ_1$ and $GJ_2$ are shown in Figure 1 and $GR$ is obtained by merging all the circuits in Figure 2. The firing time of each transition of $G$ in Figure 3 is as follows:

- $\mu(t_1) = \max(\mu_1(t_1), \mu_2(t_1)) = 1$, $\mu(t_2) = \max(\mu_1(t_2), \mu_2(t_2)) = 0$,
- $\mu(t_3) = \max(\mu_1(t_3), \mu_2(t_3)) = 2$, $\mu(t_4) = \max(\mu_1(t_4), \mu_2(t_4)) = 0$,
- $\mu(t_5) = \max(\mu_2(t_5), \mu_3(t_5)) = 2$, $\mu(t_6) = \max(\mu_2(t_6), \mu_3(t_6)) = 0$,
- $\mu(t_7) = \max(\mu_3(t_7), \mu_4(t_7)) = 1$, $\mu(t_8) = \max(\mu_3(t_8), \mu_4(t_8)) = 0$,
- $\mu(t_9) = \max(\mu_2(t_9), \mu_3(t_9)) = 1$, $\mu(t_{10}) = \max(\mu_2(t_{10}), \mu_3(t_{10})) = 0$.

![Figure 3. The model of two processes $G$.](image-url)

Note that the set $P_R$ of all the places in $GR$ can be divided into two subsets: the subset of idle places $\{s_r | s \in R\}$ of resources and the subset of non-idle places of resources $P_R \backslash \{s_r | s \in R\}$. That is, $P_R = (P_R \backslash \{s_r | s \in R\}) \cup \{s_r | s \in R\}$. As each non-idle places subset in $P_R \backslash \{s_r | s \in R\}$ belongs to $P = \bigcup_{j \in J} P_j \cup \{s_r | s \in R\}$, the set of places in $G$ is
Thus, the dimension of the set of places \( P \) is \( |P| = \sum_{j=1}^{J} (2|N_j| + 1) + R \). Therefore, the dimension of \( P \) is \( |P| = \sum_{j=1}^{J} (2|N_j| + 1) + R \). A marking \( m \) is defined by

\[
m = \{m(r_1), m(p_2), m(p_3), \ldots, m(p_{|P_{2|N_j|1}}), m(r_2), m(r_3), \ldots, m(r_{|R|})\}.
\]

In this paper, the final state place in \( P_j \) is \( p_{|P_{2|N_j|1}}^j \). The set of all final state places in \( P \) is denoted by \( P^F = \{p_{|P_{2|N_j|1}}^j, j \in J = \{1,2,\ldots,J\}\} \).

In a make-to-order manufacturing system, an order is specified by the quantity of different types of product demands, \( Q_j \), where \( j \in J = \{1,2,\ldots,J\} \), and the deadline \( \theta \). The requirements to meet an order with product demands, \( Q_j \), where \( j \in J = \{1,2,\ldots,J\} \) can be described by a marking \( m_d \) in the discrete timed Petri net \( G = (P,T,F,m_0,\mu) \), where \( m_d(p) = \begin{cases} 0 & p \in P \setminus (P^F \cup \{r_j \mid s \in R\}) \\ Q_j & p = p_{|P_{2|N_j|1}}^j \in P^F, j \in \{1,2,\ldots,J\} \end{cases} \).

Therefore, the problem to determine whether the order can be fulfilled by the deadline \( \theta \) can be stated as the following feasibility problem:

**Feasibility Problem for Nominal CPS (FPN):**

Given a DTTPN model \( G \) under a reachable marking \( m \), a target marking \( m_d \) defined by \( m_d(p) = \begin{cases} 0 & p \in P \setminus (P^F \cup \{r_j \mid s \in R\}) \\ Q_j & p = p_{|P_{2|N_j|1}}^j \in P^F, j \in \{1,2,\ldots,J\} \end{cases} \) and a deadline \( \theta \), determine whether there exists a firing sequence that brings \( G \) from \( m \) to \( m_d \) by the deadline \( \theta \).

Operations of manufacturing systems are usually influenced by unexpected events such as resource failures. The influence of resource failures may have impact on the operations and orders in the system. To deal with resource failures properly, the impact due to resource failures must be evaluated.

To study the influence of resource failures on CPS, we must represent resource failures properly. We introduce the uncertainty model \( \Delta(m) \) based on the concept of perturbation vectors and failure time intervals as follows.

**Definition 5:** A \( |P| \) dimensional perturbation vector \( \delta \) is an integer vector to specify the number of tokens lost for the set of places in \( P \) due to resource failures. The \( i \)-th element \( \delta_i \) of the perturbation vector \( \delta \) is a non-negative integer that represents the number of tokens lost in the place \( p_i \), where \( i \in \{1,2,\ldots,|P|\} \).

In this study, a discrete time horizon of \( \Pi \) periods is considered. To describe the time intervals of resource failures, we define a discrete time failure interval. Note that if \( \delta_i > 1 \), it means that there are \( \delta_i \) failures of multiple resources in place \( p_i \). As the time interval of each resource failure may not be the same, we define a failure interval as follows.

**Definition 6:** For each resource failure \( l \in \{1,2,\ldots,\delta_i\} \), the corresponding discrete time failure interval that describes the starting time and the end time of the resource failure is denoted by \( \omega_l = (\alpha_l, \beta_l) \), where \( i \in \{1,2,\ldots,|P|\} \). Note that if \( \delta_i = 0 \), \( \omega_l = [L, \infty) \), where \( L \) is a number larger than \( \Pi \).

According to Definition 6, a failure taking place outside the time horizon \( \Pi \) can be represented by \( \delta_i = 0 \) with \( \omega_l = [L, \infty) \), where \( L \) is a number larger than \( \Pi \). We use \( \omega_l \) to refer to all failure intervals \( \omega_l, i \in \{1,2,\ldots,|P|\}, l \in \{1,2,\ldots,\delta_i\} \), associated with \( \delta \).
Given a nominal cyber-physical system model \( G = (P, T, F, m_0, \mu) \), an uncertainty model is defined based on a marking \( m \) reachable from \( m_0 \). As the dimension of a marking is \( |P| \), the uncertainty model is defined as follows:

**Definition 7:** A \(|P|\) dimensional uncertainty model \( \Delta(m) \) for a reachable marking \( m \) is described by a three-tuple \( \Delta(m) = (m, \delta, \omega) \), where the \(|P|\) dimensional perturbation vector \( \delta \) represents perturbation of nominal marking and the \(|P|\) dimensional vector \( \omega \) describes all failure intervals associated with the perturbation \( \delta \).

In this paper, we will study whether deadline \( \theta \) can be met in the presence of uncertainty \( \Delta(m) = (m, \delta, \omega) \) based on the following problem formulation.

**Robustness Analysis Problem (RAP)**

Given a DTPN model \( G = (P, T, F, m_0, \mu) \) under a reachable marking \( m \), a target marking \( m_d \) defined by \( m_d(p) =\begin{cases} 0, & p \in P \setminus (P^r \cup \{r_s | s \in R\}) \\ Q_j, & p = p^t_{j,1} \in P^r, j \in \{1,2,\ldots,J\} \end{cases} \) and \( \theta \), suppose there exists a firing sequence (a nominal solution) that brings \( G \) from \( m \) to \( m_d \) by the deadline \( \theta \). Determine whether there exists a firing sequence that brings \( G \) from \( m \) to \( m_d \) by the deadline \( \theta \) under the uncertainty \( \Delta(m) = (m, \delta, \omega) \).

**4. Generation of Solutions for Nominal CPS**

The two problems presented in the previous section, FPN and RAP, aim to analyze the robustness of a solution of nominal CPS. In this section, we first present our approach to generate a solution to FPN. We will study RAP and analyze the robustness of the solution in the next section.

As the classical reachability analysis methods suffer from state explosion problem, we will propose a different approach to representing and finding solutions based on spatial-temporal network (STN) models, which describe the movement of flow tasks in space and time. The steps to construct STN will be presented later and are intuitively done by capturing the flows of tasks from upstream to downstream spatially and temporally based on DTPN.

The feasibility problem of nominal CPS aims to determine the feasibility to meet an order with \( J \) types of product demands by a deadline \( \theta \), which extends the problem for a single type of product demand addressed in [8]. The novelty of the proposed approach is to analyze the DTPN based on transformation of the model into spatial-temporal network (STN) models without relying on classical reachability analysis method for timed Petri nets. Such transformation enables compact representation of solutions and efficient search for solutions based on mathematical programming/optimization tools. The proposed method is obviously different from the ones that rely on abstraction of state space of timed Petri nets.

To deal with an order with multiple types of product demands by a deadline, a spatial-temporal network \( STN_j(V_j, A_j) \) is constructed for each \( GJ_j \), where \( j \in J = \{1,2,\ldots,J\} \), based on the demands of type \( j \) tasks, \( Q_j \), order deadline, \( \theta \), and the residual capacity. For each type of resource, the residual capacity in a period is defined as the number of unallocated resources. The main algorithm (Algorithm 2) iteratively invokes the algorithm to construct the STN, \( STN_j(V_j, A_j) \), for each type of task according to the residual capacity, allocates the tasks by solving the Feasibility Problem of Spatial-Temporal Network (FPSTN) (to be introduced later) and updates the residual capacity. The capacity on arcs of STN is bounded by the residual capacity of resources involved. For each STN, tasks can only be allocated according to the residual capacity of resources involved. The residual capacity for each period is calculated by deducting the
number of allocated resources from the original capacity. The residual capacity is updated after allocating tasks to STN. The cost on arcs is specified according to the deadline. To optimize the schedule to meet the deadline, we set the cost for arcs connecting to the end node according to the periods. The cost for an arc corresponding to period \( t \) after the deadline \( \theta \) is simply set to \( t \). The cost for all the other arcs is set to zero.

A spatial-temporal network \( STN_j(V_j, A_j) \) is constructed based on the task subnet \( GJ_j = (P_j, T_j, F_j, m_{j0}, \mu_j) \) for each \( j \in J \). Note that the set of places in \( P_j \) can be decomposed into two disjoint subsets, \( P_{j1} \) and \( P_{j2} \), where \( P_{j1} \) represents the set of idle state places for parts in type \( j \) tasks and \( P_{j2} \) represents the set of busy state places of parts in type \( j \) tasks. That is, \( P_j = P_{j1} \cup P_{j2} \). Without loss of generality, it is assumed that the places in \( P_{j1} \) correspond to states in the production process from upstream to downstream. Suppose there are \( N_j \) operations in \( GJ_j \) and each operation is represented by one start transition and one end transition. Then there are \( 2N_j \) transitions in \( P_j \). In this case, the places in \( P_j \) are numbered from upstream to downstream and are denoted as \( p_1^{\ell}, p_2^{\ell}, p_3^{\ell}, \ldots, p_{2N_j+1}^{\ell} \). \( P_j = \{ p_1^{\ell}, p_2^{\ell}, p_3^{\ell}, \ldots, p_{2N_j+1}^{\ell} \} \) and \( P_{j1} = \{ p_1^{\ell}, p_3^{\ell}, p_5^{\ell}, \ldots, p_{2N_j+1}^{\ell} \} \). To represent timing in a spatial-temporal network \( STN_j(V_j, A_j) \), a time horizon of \( \Omega \) periods is considered. A node in \( STN_j(V_j, A_j) \) represents a specific point in time and space. We create \( \Pi \) nodes for each place in \( P_{j1} \), one start node and one end node in \( STN_j(V_j, A_j) \) to represent spatial-temporal information.

In constructing \( STN_j(V_j, A_j) \), the capacity of an arc involving the use of type \( r \) resources is set to the residual capacity \( \overline{C}_{rt} \) of type \( r \) resources. Let \( C_r \) denote the initial capacity (the number) of type \( r \) resources in period \( t \), where \( C_r \) is set to \( m_0(r) \) for all \( t \). Initially, the residual capacity \( \overline{C}_{rt} \) of type \( r \) resources in period \( t \) is set to \( C_r \). The residual capacity \( \overline{C}_{rt} \) will be updated iteratively in the Algorithm 2 (to be introduced later) to check feasibility of nominal CPS to reflect the assignment of tasks to resources. The algorithm (Algorithm 1) to construct \( STN_j(V_j, A_j) \) for type \( j \) tasks, where \( j \in J \), is as follows.

**Algorithm 1: Construct Spatial-Temporal Network \( STN_j(V_j, A_j) \) for Type \( j \) Tasks**

Input: \( GJ_j, G_r^k, k \in \Omega_j^l = \{1, 2, \ldots, K_j^l\} \), \( \Omega \), \( Q_j \), \( \theta \)

Output: \( STN_j(V_j, A_j) \), where \( V_j \) is the set of nodes and \( A_j \) is the set of arcs

Step 0: Create a start node \( s_j \) and an end node \( e_j \)

\[ V_j = \{ s_j, e_j \} \]

Step 1: For \( n = 1 \) to \( N_j \)

For each \( t = 1 \) to \( \Pi \)

Create a node numbered \( (n-1)\Pi + t \)

Add the node to \( V_j = V_j \cup \{(n-1)\Pi + t\} \)

End For

End For

Step 2: Add an arc \( a = (s_j, 1) \) from \( s_j \) to the node 1

\[ A_j \leftarrow A_j \cup \{ a \} \]
Set arc capacity $c_a$ to $Q_j$ and arc cost $w_a$ to 0.

Step 3: For $n = 1$ to $N_j$
  For each $t = 1$ to $\Pi - 1$
    Add an arc $a$ from node $(n-1)\Pi + t$ to node $(n-1)\Pi + t + 1$
    $A_j \leftarrow A_j \cup \{a\}$
    Set arc capacity $c_a$ to according to the residual capacity $C_{rt}$, where $r$ is the resource type used by operation $n$
    Set arc cost $w_a$ to 0
  End For
End For

Step 4: For $n = 1$ to $N_j$
  For each $t = 1$ to $\Pi - 1$
    If $t + \mu(t_{2n}) + \mu(t_{2n}) \leq \Pi$
      Add an arc $a$ from node $(n-1)\Pi + t$ to node $n\Pi + \mu(t_{2n})$
      $A_j \leftarrow A_j \cup \{a\}$
      Find the resource type $r$ for processing operation $n$
      Update $A_{jr} \leftarrow A_{jr} \cup \{a\}$, the set of arcs in the spatial-temporal network of type $j$ task involved in the use of type-$r$ resource in period $t$
      Set arc cost $w_a$ to 0
    End If
  End For
End For

Step 5: For each $t = 1$ to $\Pi$
  Add an arc $a$ from $N_j\Pi + t$ to node $e_j$
  $A_j \leftarrow A_j \cup \{a\}$
  If $t > \theta$
    Set arc cost $w_a$ to $t$
  Else
    Set arc cost $w_a$ to 0
  End If
End For

$STN_j(V_j, A_j)$ captures the flows of parts temporally and spatially. To optimize the schedule to meet the deadline, the cost for an arc corresponding to period $t$ after the deadline $\theta$ is simply set to $t$ and the cost for all the other arcs is set to zero. We use $f_{j}$ to denote the flows of parts in $STN_j(V_j, A_j)$. We use $f_{j(u,v)}$ to represent the flows of parts from node $u$ to node $v$ in $STN_j(V_j, A_j)$ across time and space. To concisely represent the flows in $STN_j(V_j, A_j)$, we will use $f_{u,v}$ instead of $f_{j(u,v)}$ whenever it is clear from the context. In $STN_j(V_j, A_j)$, the flows of parts in horizontal arcs denote parts in waiting state and flows of parts in arcs connecting to the end node represent the finished parts. Flows of parts in all the other arcs denote the parts in busy state (being processed by machines/resources). Figure 4 shows the structure of $STN_j(V_j, A_j)$ for type 1 tasks ($j = 1$) obtained by applying the above algorithm to $GJ_1$ in Figure 2a.
Figure 4. An example of STN $(V_j, A_j)$ for $GJ_1 (j = 1)$ in Figure 1a.

Let $A_{jr}$ denote the set of arcs involved in the use of type- $r$ resource in period $t$ in STN $(V_j, A_j)$. The capacity constraints (1), the flow balance equation (2), supply constraints (3) and demand constraints (4) must hold. Based on the spatial-temporal network STN $(V_j, A_j)$, the following problem is formulated to check the feasibility to meet product demands, $Q_j$, by the deadline $\theta$.

Feasibility Problem of Spatial-Temporal Network (FPSTN) STN $(V_j, A_j)$

\[
\min_{f_j} \sum_{a=(u,v) \in A_{jr}} w_a f_j(u,v)
\]

\[
\sum_{a=(u,v) \in A_{jr}} f_j(u,v) \leq C_{rt} \forall r \in R \forall t \in \{1, 2, ..., \Pi\}
\]

(1)

\[
\sum_{w \in F} f_j(u,w) = 0 \forall u \notin \{s, e\}
\]

(2)

\[
\sum_{w \in F} f_j(s,w) = Q_j
\]

(3)

\[
\sum_{w \in F} f_j(w,e) = Q_j
\]

(4)

$f_j(u,v) \in Z \ \forall (u,v) \in A_j$, where $Z$ is the set of nonnegative integers (5).

To check whether it is feasible to meet an order with product demands, $Q_j$, for $j \in J = \{1, 2, 3, ..., J\}$ by the deadline $\theta$, the following algorithm can be applied by solving the feasibility problem of spatial-temporal network STN $(V_j, A_j)$ for $j \in J = \{1, 2, 3, ..., J\}$ iteratively.

By solving the above problem iteratively for $j \in J = \{1, 2, 3, ..., J\}$, a solution $f = \sum f_j$ will be found, where $f_j$ is the component of $f$ in spatial-temporal network STN $(V, A)$ for type $j$ task, where $j \in J$. For each $j \in J = \{1, 2, 3, ..., J\}$, we compute the residual capacity $\overline{C}_{rt} = C_{rt} - \sum_{a=(u,w) \in A_{rt}} f_j(u,w) \forall r \in R \forall t \in \{1, 2, ..., \Pi\}$ based on the solution found in STN $(V_j, A_j)$. Obviously, the following property holds.
**Property 1:** If there exists a solution $f = \sum_{j=1}^{J} f_j$ such that the flows in all the arcs from node $N_{j+1}$ to node $e$ are all zero for all $t \in \{\theta + 1, \theta + 2, \ldots, \Pi\}$ for $j \in J = \{1, 2, 3, \ldots, J\}$, $f$ can meet the deadline $\theta$.

Based on Property 1 and the reasoning above, the algorithm below can be applied to generate a solution and check the feasibility to meet the deadline.

**Algorithm 2:** Check Feasibility of Nominal CPS

---

**Input:** $J$, $R$, $\Pi$, $C_r$, $G J_j, G^k_r, k \in \Omega^j_r = \{1, 2, \ldots, K^j_r\}$, $Q_j, r \in R \forall t \in \{1, 2, \ldots, \Pi\}$

**Output:** Feasibility Indicator, $F$

**Step 0:** $f \leftarrow 0$

$C_{rt} \leftarrow C_{rt}$

**Step 1:** $J_1 \leftarrow J$

While $J_1 \neq \emptyset$

Select $j \in J_1$

Construct $STN_j(V_j, A_j)$

Find a solution $f_j$ for the Spatial-Temporal Network $STN_j(V_j, A_j)$ based on $C_{rt} \forall r \in R \forall t \in \{1, 2, \ldots, \Pi\}$

Update $C_{rt} = C_{rt} - \sum_{a=(u, w) \in A_j} f_j(u, w) \forall r \in R \forall t \in \{1, 2, \ldots, \Pi\}$

$J_1 \leftarrow J_1 \setminus \{j\}$

$f \leftarrow f + f_j$

End While

**Step 2:** If the solution $f$ satisfies the condition of Property 1

Set Feasibility Indicator = true

Else

Set Feasibility Indicator = false

End If

---

The above optimization problem is an integer programming problem that can be solved by applying any integer programming solvers such as the CPLEX Optimizer [38]. We will analyze the complexity of Algorithm 2 later in this section. The following property states that a property can be used to check whether the order deadline $\theta$ can be met.

**Property 2:** If the Feasibility Indicator is “true” by applying the Algorithm to Check Feasibility of $G = GJ\parallel GR$ with respect to $m$, $G = GJ\parallel GR$ is feasible with respect to $\theta$.

**Proof of Property 2**

Please refer to Appendix A.

Computational complexity of the Algorithm 2 to check feasibility of nominal CPS is analyzed as follows. For each $f \in J_1$, Algorithm 2 constructs $STN_j(V_j, A_j)$, finds a solution $f_j$ for the spatial-temporal network $STN_j(V_j, A_j)$ and updates
The number of nodes constructed in the spatial-temporal network \( STN_j(V_j, A_j) \) for type \( j \) task, where \( j \in J \) is \( N_j \Pi + 2 \). The number of arcs constructed in the spatial-temporal network \( STN_j(V_j, A_j) \) for type \( j \) task, where \( j \in J \), is bounded by \( 2N_j(\Pi-1)+2+\Pi \). Therefore, the complexity to construct \( STN_j(V_j, A_j) \) and the number of nodes in \( STN_j(V_j, A_j) \) is bounded by \( O(N_j\Pi) \). The problem to find the solution \( f_j \) for the spatial-temporal network \( STN_j(V_j, A_j) \) is a minimal cost flow problem with additional constraints. Therefore, a lower bound of the computational complexity to find the solution \( f_j \) for the spatial-temporal network with capacity constraint \( \overline{C}_r \) is \( O(N_j^2\Pi^2Q_j) \). Therefore, a lower bound on the computational complexity of the algorithm to construct \( RSTN_j(V_j, A_j) \) and find a solution is \( O(JR^2\Pi^2 + N_j^2\Pi^2Q_j) \). As the number of arcs in the set \( A_{jrt} \) is no greater than \( R\Pi \), the number of arithmetic operations involved in calculating \( \sum_{a=(u,w)\in A_{jrt}} f_j(u,w) \) is bounded by \( JR\Pi \) for each \( r \in R, t \in \{1,2,...,\Pi\} \). Therefore, the overall computational complexity to compute \( \overline{C}_r - \sum_{a=(u,w)\in A_{jrt}} f_j(u,w) \forall r \in R \forall t \in \{1,2,...,\Pi\} \) is bounded by \( JR^2\Pi^2 \). Therefore, the overall computational complexity to update \( \overline{C}_r \) is \( O(JR^2\Pi^2) \).

As \( J_{J_1} \) is set to \( J \) initially in Algorithm 2 and \( |J| = J \), a lower bound on the computational complexity of Algorithm 2 to check feasibility is \( O(J^2R^2\Pi^2 + JN_j^2\Pi^2Q_j) \), which is of polynomial complexity. Although the above analysis only provides a lower bound on the computational complexity of Algorithm 2, the numerical results to be presented in later shows that computational time grows polynominally, which is consistent with above lower bound on the computational complexity analyzed above.

5. Analysis of Impact of Resource Failures on CPS

Consider a perturbation vector \( \delta \) and the associated uncertainty model \( \Delta(m) = (m, \delta, \omega) \) due to resource failures. To study the influence of \( \delta \), we construct a residual spatial-temporal network \( RSTN_j(V_j, A_j) \) based on \( STN_j(V_j, A_j) \) for \( j \in J \) and the nominal solution \( f \).

The nominal solution \( f \) for the Feasibility Problem (FP) can be decomposed into several components, \( f_1, f_2, f_3, ..., f_j \), with \( f = f_1 + f_2 + f_3 + ... + f_j = \sum_{j \in J} f_j \), where \( f_j \) is the component of the solution in spatial-temporal network \( STN_j(V_j, A_j) \) for type \( j \) task, \( j \in J \). Note that \( f_j \) can be divided into two parts, \( f_{j1} \) and \( f_{j2} \), where \( f_{j1} \) denotes the flows in \( STN_j(V_j, A_j) \) not influenced by the resource failures \( \Delta(m) \) and \( f_{j2} \) represents the flows in \( STN_j(V_j, A_j) \) influenced by the resource failures \( \Delta(m) \).

Before constructing \( RSTN_j(V_j, A_j) \), we compute the residual capacity \( \overline{C}_r \) based on the nominal solution in \( STN_j(V_j, A_j) \) and resource failures first. The procedure to compute the residual capacity and construct \( RSTN_j(V_j, A_j) \) is shown as follows.
Procedure I to Compute Residual Capacity

Input: \( STN_j(V, A), f_j, j \in J, \Pi, \theta, \delta, \Delta(m) = (m, \delta, \omega), C_{rt}, r \in R, t \in \{1,2,...,\Pi}\) 

Output: \( \tilde{C}_{rt}, RSTN_j(V, A_j), Q^\delta_j \)

Step 1: Apply the Algorithm to Construct spatial-temporal network \( STN_j(V, A_j) \)

Step 2: \( RSTN_j(V, A_j) \leftarrow STN_j(V, A_j) \)

Step 3: Find the flows \( f_{j1} \) not influenced by resource failures \( \delta \) in the solution \( f_j \) of \( RSTN_j(V, A_j) \) for \( j \in J \)

Step 4: Remove all the flows \( f_{j1} \) not influenced by the resource failures \( \Delta(m) \) from \( RSTN_j(V, A_j) \) for \( j \in J \)

Step 5: Remove the flows influenced \( f_{j2} \) by the resource failures \( \Delta(m) \) from \( RSTN_j(V, A_j) \) for \( j \in J \) .

Find \( Q^\delta_j \) denote the quantity of products associated with \( f_{j2} \) influenced by resource failures \( \Delta(m) \).

Step 6: \( \tilde{C}_{rt} \leftarrow C_{rt} \) for each \( r \in R \) for each period \( t \in \{1,2,...,\Pi\} \)

Update the residual capacity \( \tilde{C}_{rt} \) for each \( r \in R \) for each period \( t \) by removing the associated resource capacities originally allocated to \( f_{j1} \) from \( C_{rt} \) for \( r \in R \) for \( j \in J \)

Update the residual capacity \( \tilde{C}_{rt} \) by removing the associated resource capacities due to resource failures \( \delta \) in \( \Delta(m) \) from \( C_{rt} \) for \( r \in R \) for \( j \in J \) :

\[
\tilde{C}_{rt} = C_{rt} - \sum_{j \in J} \sum_{a=(u,w) \in A_j} f_{j1}(u,w) \quad \forall r \in R \quad \forall t \in \{1,2,...,\Pi\}
\]

Feasibility Problem of Residual Spatial-Temporal Network (FPRSTN) \( RSTN_j(V, A) \)

\[
\min \sum_{f_{j1} = (u,v) \in A_{rt}} w_u f_{j1}(u,v)
\]

\[
\sum_{a=(u,w) \in A_{rt}} f_{j1}(u,w) \leq \tilde{C}_{rt} \quad \forall r \in R \quad \forall t \in \{1,2,...,\Pi\} \quad (6)
\]

\[
\sum_{w \in F} f_{j1}(u,w) = 0 \quad \forall u \notin \{s,e\} \quad (7)
\]

\[
\sum_{w \in F} f_{j1}(s,w) = Q_j \quad (8)
\]

\[
\sum_{w \in F} f_{j1}(w,e) = Q_j \quad (9)
\]

\( f_{j1}(u,v) \in Z \) \( \forall (u,v) \in A_j \), where is the set of nonnegative integers (10).
Consider a perturbation vector $\delta$ and $\Delta(m) = (m, \delta, \omega)$ due to resource failures. Let $J(\Delta(m))$ denote the set of task types influenced by resource failures $\delta$. We propose Algorithm 3 to iteratively check the robustness of $G = \mathcal{G} \parallel GR$ with respect to $\delta$ and $\Delta(m)$. The algorithm first applies the Procedure I to compute residual capacity $\tilde{C}_r$, construct $\text{RSTN}_r(V_j, A_j)$ according to $\text{STN}_r(V_j, A_j)$, $f_j$, $z$, $\Delta(m)$ and set the capacity constraints of the FPRSTN based on $\text{RSTN}_r(V_j, A_j)$ according to the residual capacity $\tilde{C}_r$. Finally, it finds the solution $f'_j$ for the FPRSTN for $\text{RSTN}_r(V_j, A_j)$. If the solution $f'_j$ found for each $j \in J$ satisfies the condition of Property 1, the $\text{Robustness}_{\text{Indicator}}$ will be set to “true” to indicate the solution is robust with respect to resource failures $\delta$ and $\Delta(m)$.

Algorithm 3: Check Robustness with respect to Uncertainty $\Delta(m)$

Input: $\text{STN}_r(V_j, A_j)$, $f_j$, $\forall j \in J$, $\Pi$, $\theta$, $\Delta(m) = (m, \delta, \omega)$
Output: $\text{Robustness}_{\text{Indicator}}$, $f'_j$

Step 1: Find $J(\Delta(m))$
Step 2: Apply Procedure I to compute residual capacity $\tilde{C}_r$ for $r \in R \forall t \in \{1, 2, \ldots, \Pi\}$ and $Q^\delta_j$ for each $j \in J$
Step 3: While $J(\Delta(m)) \neq \Phi$
    Select $j = j_0 \in J(\Delta(m))$
    Construct $\text{RSTN}_r(V_j, A_j)$ according to $\text{STN}_r(V_j, A_j)$ and $\tilde{C}_r$
    Set the capacity constraints of the FPRSTN defined by $\text{RSTN}_r(V_j, A_j)$ according to the residual capacity $\tilde{C}_r$
    Find the solution $f'_j$ for the FPRSTN defined by $\text{RSTN}_r(V_j, A_j)$ with residual capacity constraint $\tilde{C}_r$
    Update $f'$ based on $f'_j$
    Update residual capacity $\tilde{C}_r$ by deducting the capacity assigned to
    $f'_j$ from $\tilde{C}_r$: $\tilde{C}_r \leftarrow \tilde{C}_r - \sum_{(u,w) \in A_j} f'_j(u, w)$
    If the solution $f'_j$ satisfies the condition of Property 1
        Set $\text{Robustness}_{\text{Indicator}}_j = \text{true}$
    Else
        Set $\text{Robustness}_{\text{Indicator}}_j = \text{false}$
End If
End While
$\text{Robustness}_{\text{Indicator}} \leftarrow \text{true}$
For $j \in J(\Delta(m))$
    $\text{Robustness}_{\text{Indicator}} \leftarrow \text{Robustness}_{\text{Indicator}} \land \text{Robustness}_{\text{Indicator}}_j$
End For
Return $\text{Robustness}_{\text{Indicator}}$
Property 3: If the Robustness Indicator is “true” by applying the Algorithm 3 to Check Robustness of \( G = GJ\|GR \) with respect to \( \Delta(m) \), \( G = GJ\|GR \) is robust with respect to \( \Delta(m) \).

Proof of Property 3

Please refer to Appendix B.

Computational complexity of the Algorithm 3 to check robustness is analyzed as follows. Note that the Algorithm 3 to check robustness invokes the construction of \( RSTN_j(V_j, A_j) \) based on the nominal solution and \( STN_j(V_j, A_j) \) for each \( j \in J(\Delta(m)) \).

To construct \( RSTN_j(V_j, A_j) \), Procedure I to is called to compute the residual capacity

\[
\tilde{C}_r = C_r - \sum_{j=J_u=(u,v)\in A_j} f_j(u,v) - C_r^D \quad \text{for} \quad r \in R, t \in \{1, 2, ..., \Pi\}.
\]

As the number of arcs in the set \( A_{j_u} \) is no greater than \( RPI \), the number of arithmetic operations involved is bounded by \( JRPI \) for each \( r \in R, t \in \{1, 2, ..., \Pi\} \). Therefore, the overall computational complexity to compute the residual capacity is \( O(JR^2\Pi^2) \).

Note that the number of nodes constructed in the spatial-temporal network \( STN_j(V_j, A_j) \) for type \( j \) task, where \( j \in J \) is \( N_j\Pi + 2 \). The number of arcs constructed in the spatial-temporal network \( STN_j(V_j, A_j) \) for type \( j \) task, where \( j \in J \), is bounded by \( 2N_j(\Pi-1) + 2 + \Pi \). Therefore, the complexity to construct \( STN_j(V_j, A_j) \) and the number of nodes in \( STN_j(V_j, A_j) \) is bounded by \( O(N_j\Pi) \). As the structure of \( RSTN_j(V_j, A_j) \) is the same as \( STN_j(V_j, A_j) \), the complexity to construct \( RSTN_j(V_j, A_j) \) and the number of nodes in \( RSTN_j(V_j, A_j) \) is bounded by \( O(N_j\Pi) \). The problem to find the solution \( f'_{j_2} \) for the FPRSTN (feasibility problem of residual spatial-temporal network) defined by \( RSTN_j(V_j, A_j) \) with residual capacity constraint \( \tilde{C}_r \) is a minimal cost flow problem with additional constraints. Therefore, a lower bound of the computational complexity to find the solution \( f'_{j_2} \) for the FPRSTN defined by \( RSTN_j(V_j, A_j) \) with residual capacity constraint \( \tilde{C}_r \) is \( O(N_j^2\Pi^2Q^D) \). Therefore, a lower bound on the computational complexity of the algorithm to construct \( RSTN_j(V_j, A_j) \) and find a solution is \( O(JR^2\Pi^2 + N_j^2\Pi^2Q^D) \). As \( |J(\Delta(m))| = I \) is no greater than \( J \), a lower bound on the computational complexity of the Algorithm 3 to Check Robustness is \( O(J^2R^2\Pi^2 + JN_j^2\Pi^2Q^D) \), which is of polynomial complexity. Although the above analysis only provides a lower bound on the computational complexity of the algorithm, we will show that the numerical results indicate that computational time grows polynomially, which is consistent with above lower bound on the computational complexity analyzed above.

6. Results

In this section, we illustrate the proposed method by examples and present the results to illustrate computational feasibility of the proposed method. We first illustrate the results obtained by applying the proposed method to two examples that represent two application scenarios. In Example 1, a scenario in which resource failures influence only one type of product is considered. The patterns of resource failures in the scenario of Example 2 influence two types of products. We illustrate how the proposed method works to check the robustness of CPS with respect to resource failures. Following Exam-
ple 1 and Example 2, we will present the results to study computational feasibility of the proposed method.

**Example 1:** Consider a CPS that can produce two types of products. The production processes of the two types of products and the resources are modeled by task subnets, $GJ_1$ and $GJ_2$ in Figure 1 and the resource models in Figure 2, respectively. The CPS model is shown in Figure 3. The models can be downloaded from the following link: https://drive.google.com/drive/folders/1rBiYKMoMDJOU1ktrvjdGllz_nmi1FzXOmdY?usp=sharing (Accessed on 15 May 2021)

Suppose an order which requires three type-1 products and deadline AM 9:20 has arrived. A time horizon starting from AM 8:00 is considered to handle the order. The time horizon is divided into time periods and duration of each period is ten minutes. For this example, the order deadline AM 9:20 is presented by $\theta = 8$ and we set $\Pi = 10$ in our problem formulation. The data of the order described based on the above parameters is summarized in Table 2. As the order only requires type-1 products, only the operations with associated transitions in the type-1 task subnet and the associated processing time of the resources are shown in Table 3.

![Figure 5](image-url)  
*Figure 5.* A nominal solution is represented in $STN(V_1, A_i)$, where the flows influenced by resource failure is represented in the red path.

In the process of producing products, resource may fail unexpectedly. Suppose a resource failed during the period from AM 8:00. Suppose the failure is expected to be recovered by AM 8:20. The resource failure information is shown in Table 4. As there are
16 places in the CPS model, for this example, we represent the resource failure by the perturbation vector \( \delta = [0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0] \) and discrete time failure interval \( \omega_{21} = [\alpha_{21} \ \beta_{21}] = [1 \ 2] \). To assess the influence of the resource failure, a residual spatial-temporal network is constructed based on \( \Delta(m) = (m, \delta, \omega) \). A solution is found by solving the FPRSTN defined by \( RSTN_i(V_j, A_j) \) with \( \Delta_i(m) = (m, \delta, \omega) \).

| Resource Type | No. of Resources Failed | Operation | Start Period | End Period |
|---------------|------------------------|-----------|--------------|------------|
|               | 1                      | 1         | 1            | 2          |

As only one type of product involved in the nominal solution, only type-1 products will be influenced. Therefore, \( J(\Delta(m)) = \{1\} \) in Algorithm 3. By solving the FPRSTN defined by \( RSTN_j(V_j, A_j) \) with residual capacity constraint for \( j = 1 \), a solution is obtained as follows: \( f_{3,14} = 1, \ f_{14,26} = 1 \) and \( f_{26,1} = 1 \). This solution satisfies the condition of Property 3. The above solution can meet the deadline as the condition of Property 1 holds. Figure 6a shows the above solution represented in \( RSTN_i(V_1, A_1) \). The solution displayed in Figure 6a is obtained by rerouting the flows in the red path of Figure 5. Figure 6b shows the overall solution obtained by combining the solution in Figure 6a (rerouted workflows) with the flows not influenced by resource failures. It indicates that the solution can meet the deadline as the flows of each arc after the deadline is zero.

Figure 6. (a) The rerouted flows due to resource failure represented in \( RSTN_i(V_1, A_1) \); (b) One solution represented in \( STN_i(V_1, A_1) \) obtained after resource failure.

Example 2: Please refer to Appendix C for the details.

To illustrate the computational feasibility of the proposed method, we conduct relevant experiments. These experiments were running on a computer with an Intel CoreTM i7-4720 HQ CPU, 2.6 G Hz, and 16 GB of RAM. Two types of experiments have been performed. The first type of experiments aims to study the influence of number of tasks that need to be rerouted due to resource failures on the computation time. The second type of experiment focuses on the scalability of the proposed method with respect to the number of transitions in a task subnet.

For the first type of experiment, we increase the number of tasks influenced, \( O^I \), due to resource failures to study how the computational time grows with the number of tasks influenced. The number of tasks influenced due to resource failures is changed from 1 to 150. In this type of experiment, the number of operations in the task subnet is fixed at 10. Therefore, the number of transitions in the first type of experiments is 20. The time
horizon, $\Pi$, is set to 722 in this type of experiment. The results are shown in Figure 7a. This indicates that the growth of computational time with respect to number of tasks influenced due to resource failures is polynomial.

For the second type of experiment, to study the scalability of the proposed algorithm, we conduct experiments by varying the number of transitions in the processes from 8 to 40. The time horizon, $\Pi$, is set to 722 in this type of experiment. The results are shown in Figure 7a. This indicates that the growth of computational time with respect to number of transitions in the task subnet is polynomial.

![Figure 7. (a) Computational time with respect to the number of tasks influenced, $Q^\delta_j$, due to resource failures; (b) Computational time with respect to number of transitions in task subnets.](image)

7. Discussion

Although CPS provides a paradigm to manage, monitor and control manufacturing resources in enterprises based on the real-time information from the IoT and ICT infrastructure, an effective scheme to deal with unexpected events must be developed to enhance robustness of CPS. Robust design of CPS becomes an important issue to achieve stable and secure CPS. It has attracted a lot of attention in the CPS research community. A comprehensive survey of the concept and strategies for robust design of CPS can be found in [5]. Although many studies have addressed the robustness issues on stability, security and systemativeness in the context of CPS, robustness of CPS with respect to resource failures is less explored in the literature. This study aims to bridge this gap through development of a methodology to model and analyze robustness of CPS with respect to resource failures.

CPS relies on an effective scheme to deal with changes in processes, resources and implementation technology. Model driven approach (MDA) provides a method to accommodate these changes [11]. MDA is an approach for developing software systems based on specifications described as models. This MDA approach is supported by the Unified Modeling Language™ [41]. Although UML is the standard for specification of software systems, it is weak in analyzing production processes with shared resources, concurrent, synchronous and asynchronous operations in CPS. Petri nets provides the capabilities to model and analyze CPS. In [8], a class of Petri nets called DTPN is proposed to model CPS. In addition, due to the lack of an efficient method to analyze DTPN, an analysis method is proposed based on transformation of the DTPN model to a network model to analyze temporal property of the DTPN model. However, the problem in [8] considers order requirements with a single type of product demand and the issue to deal with resource failures is not addressed in [8]. In terms of modeling tools and analysis methods, despite the DTPN model for CPS being proposed previously, there still lacks an analysis method to model and analyze the impact of resource failures...
on the performance of CPS. Although the development of this paper is based on the preliminary results in [15] to analyze the influence of resource failures on CPS, the content presented in this paper is different from [15], in that it includes the rigorous proof of robustness properties of CPS, computational complexity analysis and verification of complexity analysis results by experiments, which are not covered in [15]. In addition, the problem studied in this paper considers order requirements with multiple types of product demands whereas the problem in [8] considers order requirements with a single type of product demand.

In Petri net literature, there are two ways to describe time regarding transition firing. Timed Petri nets specify a duration to represent firing time for a transition [27]. Time Petri nets describe a time interval for firing transition [42]. From the perspective of theoretical development, the timed Petri net model considered in this study is different from [43,44] in that the DTPN model used in this study specifies transition firing time delay whereas the time Petri net model used in [43] specifies time intervals for transition firing. The transition firing time delay in DTPN is discrete, which makes it possible to analyze the robustness property of the system with RSTN. In [42], Akshay et al. defined a model to be robust if the set of discrete behaviors is preserved under arbitrarily small perturbations in the firing intervals of transitions. Akshay et al. showed that TPNs are not robust in general and that the problem to check robustness with respect to boundedness and safety properties is undecidable. In [45], robustness is defined as a property to measure the allowed variability of the timing delays in their neighborhood. The model used in [45] is also TPN which specifies time intervals for transition firing. The robustness property studied in this paper is not robustness with respect to firing time variation. Instead, we study the robustness with respect to perturbation in markings, which corresponds to changes in the number of resources in the system. Therefore, the study in this paper is different from the papers mentioned above.

In this study, we focus on the development of an algorithm to analyze the impact of resource failures on the operations and temporal properties of CPS, based on transformation of the DTPN model of CPS into relevant spatial-temporal network models. We illustrate the proposed algorithm by applying it to two application scenarios. In the first scenario, only one type of production process is influenced by the resource failure. The results indicate that the influenced workflows are rerouted properly to reflect the unavailability of the failed resource. In the second scenario, two types of production processes are influenced due to the resource failure. The results also show that the influenced workflows of the two production processes are rerouted properly to reflect the unavailability of the failed resource. In addition to the two application scenarios above, we also perform experiments to study how the computational time grows with respect to number of tasks influenced due to resource failures and how the computational time grows with respect to number of transitions in the processes. The results of the computational time experiments indicate that the proposed is computationally feasible and is consistent with the characteristics of the lower bound on the polynomial complexity obtained in our analysis. This study provides a foundation for the analysis of temporal properties in CPS.

Complexity analysis provides a formal and widely accepted way to assess the computationally feasibility of different algorithms in the literature. We also follow this approach in this paper to compare our approach with classical reachability analysis methods. Classical reachability analysis methods suffer from the exponential state explosion problem even for untimed Petri nets. In the literature, complexity of the reachability problem of untimed Petri nets is exponential space and time although it is decidable [46,47]. For timed Petri nets, the complexity is higher than untimed Petri nets due to consideration of time factor. The complexity of classical reachability analysis methods for timed Petri nets also suffers from the state explosion problem. For the STN based approach proposed in this paper, the lower bound of the computational complexity is polynomial with respect to problem size parameters. Our experimental results are con-
sistent with this lower bound. The proposed STN based approach has advantage over classical reachability analysis methods in terms of computational feasibility to deal with real problems. This advantage is due to the polynomial growth of the STN structure with respect to problem size parameters, compact representation of solutions and an effective way to search for solutions in STN.

In this paper, we limit our scope and aim to study the robustness of CPS for manufacturing systems. Depending on the domains in which CPS are applied, the challenges vary. Therefore, robustness of CPS need to be defined and studied in the context of a specific problem domain. Although the scope of this paper is target at robustness of CPS for manufacturing systems, it is promising to extend the proposed method to apply to other temporal analysis problems in CPS for manufacturing systems modelled with timed Petri nets. In addition to the issue to analyze the impact of resource failures addressed in this study, the widespread adoption of CPS raises other challenging research issues. Safety of CPS is another important issue that has attracted researchers’ attention. For example, the study [48] focuses on the safety property in which multiple CPS collaborate. Many safety properties are relevant to temporal analysis. An interesting issue to be studied is the feasibility of applying the proposed method to analyze the safety property of CPS.

8. Conclusions

With the maturity and adoption of the IoT and ICT infrastructure in CPS, availability of the real-time information from the shop floor enhances visibility of production status for manufacturers. The real-time information provided by the IoT and ICT infrastructure enables the development of an effective strategy to deal with undesirable events in CPS, such as resource failures. Resource failures may take place unexpectedly and have negative effects on the operations of CPS. Assessing the impact of resource failures on CPS is an important issue. Development of a systematic method to support analysis of the impact of resource failures on CPS is needed. This paper aims to propose a method to evaluate the effects of resource failures on operations and temporal properties of CPS by constructing residual spatial-temporal networks obtained based on transformation of a class of discrete timed Petri net models for CPS and the system state information. To evaluate the impact due to resource failures, we formulate a problem based on the residual spatial-temporal networks to check whether the order deadline is feasible after resource failures to efficiently assess the system. The proposed method is illustrated by examples. To assess practicality of the proposed method in terms of computational feasibility, we analyzed the computational complexity of the proposed algorithm and conducted experiments to study the growth of computational time required with respect to the number of tasks influenced due to resource failures and number of transitions in the processes. The analysis indicates that the lower bound on the complexity of the algorithm is polynomial with respect to problem size parameters. The numerical results of the experiments also indicate that the CPU time grows polynomially with respect to the number of tasks influenced due to resource failures and number of transitions in the processes. The numerical results are consistent with the complexity analysis and indicate that the proposed method can be applied to solve real problems. Formal models such as timed Petri nets have been widely accepted as tools for the specifying, modelling and simulation of systems. However, analysis methods of timed Petri nets are still limited. The method proposed in this paper attempts to analyze the temporal property of timed Petri nets based on transformation of the reachability problem into a problem relevant to the scheduling problem in the literature. Such transformation is nontrivial and provides an alternative approach to the analysis of timed Petri nets. One future research direction is to extend the proposed method to other types of processes. Although this study focuses on robustness of temporal property of CPS with respect to resource failures, this is not the only robustness research issue. There are other robustness issues of CPS due to the variety of uncertainties in the real world. These uncertainties include delay of raw
materials/parts, unforeseen loss of manpower, arrival/cancellation of orders and variation of processing time of operations. Robustness of CPS can be studied for each type of uncertainty. In summary, there are a lot of promising future research directions relevant to robustness of CPS.

**Funding:** This research was supported in part by the Ministry of Science and Technology, Taiwan, under Grant MOST 109-2410-H-324-001.

**Institutional Review Board Statement:** Not applicable for studies not involving humans or animals.

**Informed Consent Statement:** Not applicable for studies not involving humans.

**Data Availability Statement:** Data available in a publicly accessible repository described in the article.

**Conflicts of Interest:** The author declares no conflicts of interest.

**Appendix A**

**Proof of Property 2**

Suppose the `Feasibility Indicator` is “true” after applying Algorithm 2 to check feasibility of nominal CPS. We must show that the solution \( f = \sum_{j \in J} f_j \) forms a solution that satisfies the constraints and meets the deadline for the original problem. First, we show that solution \( f = \sum_{j \in J} f_j \) satisfies the flow balance constraints.

As \( f_j, j \in \{1,2,3,\ldots,J\} \), is obtained by solving the FPSTN of \( STN_j(V_j, A_j) \), it satisfies the following constraints for each \( j \in \{1,2,3,\ldots,J\} \):

\[
\sum_{w \in F} f_j(u,w) = 0 \quad \forall u \notin \{s,e\} \tag{A1}
\]

\[
\sum_{w \in F} f_j(s,w) = Q_j \tag{A2}
\]

\[
\sum_{w \in F} f_j(w,e) = Q_j \tag{A3}
\]

By summing up the terms on the left side and summing up the terms on the right side of constraint (2), respectively, the following equations hold:

\[
\sum_{j \in \{1,2,\ldots,J\}} \sum_{w \in F} f_j(u,w) = 0 \quad \forall u \notin \{s,e\} \tag{A4}
\]

\[
\sum_{w \in F} \sum_{j \in \{1,2,\ldots,J\}} f_j(u,w) = 0 \quad \forall u \notin \{s,e\} \tag{A5}
\]

Therefore, the flow balance equation holds for the solution \( f = \sum_{j \in J} f_j \).

Note that the solution \( f = \sum_{j \in J} f_j \) consists of \( f_j \) in the spatial-temporal network \( STN_j(V_j, A_j) \).

Therefore, \( \sum_{w \in F_j} f_j(s,w) = Q_j \) and \( \sum_{w \in F_j} f_j(s,w) = 0 \) for \( j' \in \{1,2,3,\ldots,J\} \setminus \{j\} \).

Therefore, \( \sum_{w \in F_j} \sum_{j \in J} f_j(s,w) = Q_j \). That is, the quantity of the products in the solution \( f = \sum_{j \in J} f_j \) is exactly the same as the product demands of the order.

By similar reasoning, \( \sum_{w \in F_j} \sum_{j \in J} f_j(w,e) = Q_j \). That is, the quantity of the products produced in the solution \( f = \sum_{j \in J} f_j \) is exactly the same as the product demands of the
order. Based on the above reasoning, the solution \( f = \sum_{j \in J} f_j \) satisfies the flow balance constraints.

Next, we prove that the solution \( f = \sum_{j \in J} f_j \) satisfies the capacity constraints.

As \( f_j, \ j \in \{1,2,3,...,J\} \), is obtained by solving the FPSTN of \( STN_j(V_j, A_j) \), the following capacity constraints must be satisfied:

For \( j = 1 \), the following constraints must be satisfied:

\[
\sum_{a=(u,w) \in A_u} f_1(u,w) \leq C_a = C_a \forall r \in R \forall t \in \{1,2,...,\Pi\}
\]  
(A6)

\[
\sum_{w \in F} f_1(u,w) = 0 \forall u \notin \{s,e\}
\]  
(A7)

\[
\sum_{w \in F} f_1(s,w) = Q_1
\]  
(A8)

\[
\sum_{w \in F} f_1(w,e) = Q_1
\]  
(A9)

The following hold for \( j \in \{2,3,...,J\} \)

\[
\sum_{a=(u,w) \in A_u} f_j(u,w) \leq C_a = C_a - \sum_{j' \in \{1,2,...,j-1\}} \sum_{a=(u,w) \in A_u} f_j(u,w) \forall r \in R \forall t \in \{1,2,...,\Pi\}
\]  
(A10)

\[
\sum_{w \in F} f_j(u,w) = 0 \forall u \notin \{s,e\}
\]  
(A11)

\[
\sum_{w \in F} f_j(s,w) = Q_j
\]  
(A12)

\[
\sum_{w \in F} f_j(w,e) = Q_j
\]  
(A13)

\[
\sum_{a=(u,w) \in A_u} f_j(u,w) \leq C_a - \sum_{j' \in \{1,2,...,j-1\}} \sum_{a=(u,w) \in A_u} f_j(u,w) \forall r \in R \forall t \in \{1,2,...,\Pi\}
\]  
(A14)

Therefore,

\[
\sum_{a=(u,w) \in A_u} f_j(u,w) - \sum_{j' \in \{1,2,...,J\}} \sum_{a=(u,w) \in A_u} f_j(u,w) \leq C_a \forall r \in R \forall t \in \{1,2,...,\Pi\}
\]  
(A15)

Hence the following holds:

\[
\sum_{j' \in \{1,2,...,J\}} \sum_{a=(u,w) \in A_u} f_j(u,w) \leq C_a \forall r \in R \forall t \in \{1,2,...,\Pi\}
\]  
(A16)

The capacity constraints are satisfied.

Based on the above reasoning, the solution \( f = \sum_{j \in J} f_j \) satisfies the flow balance constraints as well as the capacity constraints. It is feasible to meet the order product demands \( Q_j \) for \( j \in \{1,2,3,...,J\} \) by the deadline \( \theta \).

**Appendix B**

**Proof of Property 3:**

Suppose the Robustness Indicator is “true” after applying Algorithm 3 to check robustness of the nominal CPS. We prove by showing that the solution \( f'_{j2} \) obtained by solving the FPRSTN of \( RSTN_j(V_j, A_j) \) with residual capacity constraint \( \tilde{C}_{j2} \) for each \( j \in J(\Delta(m)) \) and the solution \( f'_{j1} \) (corresponding to the flows in \( STN_j(V_j, A_j) \) not in-
fluenced by the resource failures \( \Delta(m) \) for each \( j \in J \) form a solution that satisfies the constraints and meet the deadline for the original problem.

We first show that \( f'_{j_2} \) for each \( j \in J(\Delta(m)) \) and \( f'_{j_1} \) for each \( j \in J \) satisfy the capacity constraints.

Suppose \( J(\Delta(m)) = \{ j_k | k \in \{1,2,\ldots,\Gamma \} \} \). As the Robustness Indicator is “true” by applying the Algorithm 3 to check robustness of \( G = G_1 \bigcup G_2 \) with respect to \( \Delta(m) \), Robustness Indicator \( j = “true” \) for \( j = j_k \), where \( k \in \{1,2,\ldots,\Gamma \} \).

As Robustness Indicator \( j = “true” \), where \( j = j_k \) with \( k \in \{1,2,\ldots,\Gamma \} \), it implies that the solution \( f'_{j_2} \) for the FPRSTN defined by RSTN \(_j(V_j, A_j) \) with residual capacity constraint \( \tilde{C}_{rt} \) satisfies the condition of Property 1.

That is, the flows associated with the solution \( f'_{j_2} \) in the arcs from node \( N_j, \Pi + t \) to node \( e \) are all zero for all \( t \in \{\theta + 1, \theta + 2, \ldots, \Pi \} \) and \( f'_{j_2} \) can meet the due date \( \theta \). Note that the solution \( f'_{j_2} \) is found by solving the FPRSTN defined by RSTN \(_j(V_j, A_j) \) with residual capacity constraint \( \tilde{C}_{rt} \).

Without loss of generality, suppose \( j = j_1 \) is selected from \( J(\Delta(m)) \) for the first iteration. The following inequalities hold:

\[
\sum_{(u, w) \in A_{j_1}} f'_{j_2}(u, w) \leq \tilde{C}_{rt} = C_{rt} - \sum_{j \in J} \sum_{w \in A_{j_1}} f_{j}(u, w) - C^0_{rt} \forall r \in R \forall t \in \{1, \ldots, \Pi\}
\]

After processing \( j = j_1 \), the residual capacity is updated by

\[
\tilde{C}_{rt} = C_{rt} - \sum_{j \in J} \sum_{w \in A_{j_1}} f_{j}(u, w) - C^0_{rt} \forall r \in R \forall t \in \{1, \ldots, \Pi\}
\]

\[
\tilde{C}_{rt} \leftarrow \tilde{C}_{rt} - \sum_{(u, w) \in A_{j_1}} f'_{j_2}(u, w) = C_{rt} - \sum_{j \in J} \sum_{w \in A_{j_1}} f_{j}(u, w) - C^0_{rt} - \sum_{(u, w) \in A_{j_1}} f'_{j_2}(u, w)
\]

The set \( J(\Delta(m)) \) is updated as follows:

\[
J(\Delta(m)) \leftarrow J(\Delta(m)) \setminus \{ j_1 \}
\]

The algorithm proceeds to the second iteration.
Suppose \( j = j_2 \) is selected from \( J(\Delta(m)) \) for the second iteration.
Suppose the solution \( f'_{j_2} \) is found by solving the FPRSTN defined by RSTN \(_j(V_j, A_j) \) with residual capacity constraint

\[
\tilde{C}_{rt} = C_{rt} - \sum_{j \in J} \sum_{w \in A_{j_1}} f_{j}(u, w) - C^0_{rt} - \sum_{(u, w) \in A_{j_1}} f'_{j_2}(u, w)
\]

The solution \( f'_{j_2} \) must satisfy the capacity constraint:

\[
\sum_{(u, w) \in A_{j_1}} f'_{j_2}(u, w) \leq \tilde{C}_{rt} = C_{rt} - \sum_{j \in J} \sum_{w \in A_{j_1}} f_{j}(u, w) - C^0_{rt} - \sum_{(u, w) \in A_{j_1}} f'_{j_2}(u, w)
\]

After processing \( j = j_2 \), the residual capacity is updated by
By following similar procedure, after finishing the iteration \( I-1 \), the solution \( f'_{j(k)} \) must satisfy the capacity constraint:

\[
\sum_{(u,w) \in A_{jrt}} f'_{j(k,2)}(u,w) \leq C_{rt} = C_{rt} - \sum_{j \in J} \sum_{a=(u,w) \in A_{jrt}} f_j(u,w) - C^{\delta} - \sum_{j \in \{j_1, j_2, \ldots, j_l\}} \sum_{(u,w) \in A_{jrt}} f'_{j}(u,w)
\]

Therefore, the following constraint holds:

\[
\sum_{j \in J} \sum_{a=(u,w) \in A_{jrt}} f_j(u,w) + \sum_{j \in \{j_1, j_2, \ldots, j_l\}} \sum_{(u,w) \in A_{jrt}} f'_{j}(u,w) \leq C_{rt} = C_{rt} - \sum_{j \in \{j_1, j_2, \ldots, j_l\}} \sum_{(u,w) \in A_{jrt}} f'_{j}(u,w)
\]

Thus the capacity constraint is satisfied.

Therefore, \( \sum_{j \in J} f_j + \sum_{j \in J(\Delta(m))} f'_{j(k)} \) is a solution that satisfies the capacity constraints after resource failures of \( \delta \in \Delta(m) \).

Next, we show that \( f'_{j(k)} \), \( j_k \in J(\Delta(m)) \), and \( f_j \), \( j \in J \), satisfy the flow balance constraints.

Let \( Q^\delta_j \) denote the quantity of the products influenced by the resource failure \( \delta \). Note \( f_j \) must satisfy the following constraints:

\[
\sum_{w \in F} f_j(u,w) = 0 \quad \forall u \notin \{s,e\} \quad \forall j \in J \quad (A17)
\]

\[
\sum_{w \in F} f_j(s,w) = Q_j - Q^\delta_j \quad \forall j \in J \quad (A18)
\]

\[
\sum_{w \in F} f_j(w,e) = Q_j - Q^\delta_j \quad \forall j \in J \quad (A19)
\]

Note \( f'_{j(k)} \) must satisfy the following constraints:

\[
\sum_{w \in F} f'_{j(k,2)}(u,w) = 0 \quad \forall u \notin \{s,e\} \quad \forall j \in J \quad (A20)
\]

\[
\sum_{w \in F} f'_{j(k,2)}(s,w) = Q^\delta_j \quad \forall j \in J \quad (A21)
\]

\[
\sum_{w \in F} f'_{j(k,2)}(w,e) = Q^\delta_j \quad \forall j \in J \quad (A22)
\]

The following equations hold:

\[
\sum_{w \in F} (f_j(u,w) + f'_{j(k,2)}(u,w)) = 0 \quad \forall u \notin \{s,e\} \quad \forall j \in J \quad (A23)
\]

\[
\sum_{w \in F} (f_j(s,w) + f'_{j(k,2)}(s,w)) = Q_j - Q^\delta_j + Q^\delta_j = Q_j \quad \forall j \in J \quad (A24)
\]
\[ \sum_{w \in F} [f_{j_1}(w, e) + f'_{j_2}(u, w)] = Q_j - Q'_{j_2} + Q'_{j_1} = Q_j \quad \forall j \in J \] (A24)

Hence \( f'_{j_2} \) and \( f_{j_1}, j \in J \), satisfy the flow balance constraints.

By following reasoning, \( f'_{j_2} \) for each \( j \in J(\Delta(m)) \) and \( f_{j_1} \) for each \( j \in J \) satisfy the flow balance constraints.

Finally, we show that \( f'_{j_2}, j \in J(\Delta(m)) \), and \( f_{j_1}, j \in J \), meet the deadline. Note that the flows \( f_{j_1}, j \in J \), are the parts of the nominal solution corresponding to the flows in \( STN_f(V, A_f) \) not influenced by the resource failures \( \Delta(m) \). As the nominal solution meets the deadline, \( f_{j_1}, j \in J \), meet the deadline. We show that \( f'_{j_2}, j \in J(\Delta(m)) \), meet the deadline. Note that \( f'_{j_2} \) is obtained by solving the FPRSTN of \( RSTN_f(V_f, A_f) \) with residual capacity constraint \( C_{r_f} \) for each \( j \in J(\Delta(m)) \). As Robustness _Indicator\( _f = \text{"true"} \) for each \( j \in J(\Delta(m)) \), where \( k \in \{1,2,3,\ldots \} \), it implies that the solution \( f'_{j_2} \) meets the deadline for each \( j \in J(\Delta(m)) \). Therefore, \( f'_{j_2}, j \in J(\Delta(m)) \) and \( f_{j_1} \) for each \( j \in J \) can meet the deadline.

Based on the discussions above, if the Robustness _Indicator\( _f \) is “true” by applying the Algorithm 3 to check robustness of \( G = G_f \| GR \) with respect to \( \Delta(m) \), \( f'_{j_2}, j \in J(\Delta(m)) \), and \( f_{j_1}, j \in J \), form a solution that satisfies the capacity constraints, flow balance constraints as well as the deadline.

**Appendix C**

Example 2: Consider a CPS that can produce two types of products. The production processes of the two types of products and the resources are modeled by Figure 1 and Figure 2, respectively. The CPS model of the type-1 product is shown in Figure 3. Suppose an order has arrived. The order requires three type-1 products, one type-2 product and deadline AM 9:40. A time horizon starting from AM 8:00 is considered to handle the order. The time horizon is divided into time periods and the duration of each period is ten minutes. For this example, the order deadline AM 9:40 is presented by \( \theta = 10 \) and we set \( \Pi = 10 \) in our problem formulation. The data of the order described based on the above parameters is summarized in Table A1. The operations with associated transitions in the type-1 task subnet and the associated processing time of the resources are shown in Table 3. The operations with associated transitions in the type-2 task subnet and the associated processing time of the resources are shown in Table A2.

**Table A1.** Order Data.

| Type of Product | Demands (Quantity) | Deadline |
|-----------------|--------------------|----------|
| 1               | 3                  | 10       |
| 2               | 1                  | 10       |

**Table A2.** Data for Operations, Transitions and Resources involved in Type-2 Tasks.

| Resource Type | Capacity | Operation | Transitions | Processing Time |
|---------------|----------|-----------|-------------|-----------------|
| 1             | 1        | 3         | \( t_5, t_6 \) | 2               |
| 2             | 1        | 4         | \( t_7, t_8 \) | 1               |
| 3             | 1        | 5         | \( t_9, t_{10} \) | 1               |

By applying Algorithm 1 to check feasibility of nominal CPS, we obtain a nominal solution for type-1 products as follows: \( f_{1,12} = 1 \), \( f_{4,15} = 1 \), \( f_{5,16} = 1 \), \( f_{12,24} = 1 \), \( f_{15,27} = 1 \) and \( f_{16,28} = 1 \). To present the solution clearly, the above solution is shown in a
spatial-temporal network $STN_i(V_1,A_i)$ in Figure A1a. We also obtain a nominal solution for type 2 products as follows: $f_{2,14} = 1, f_{14,25} = 1, f_{25,36} = 1$. To present the solution clearly, the above solution is shown in a spatial-temporal network $STN_2(V_2,A_2)$ in Figure A1b. Obviously, the above solution can meet the deadline of the order.

Suppose a resource failed during the period from AM 8:00. Suppose the failure is expected to be recovered by AM 8:20. The resource failure information is shown in Table A3. As there are 16 places in the CPS model, we represent the resource failure by the perturbation vector $\delta = [0 \ 1 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0 \ 0]$ and discrete time interval $\omega_{21} = [\alpha_{21} \beta_{21}] = [1 \ 2]$. To assess the influence of the resource failure, a residual spatial-temporal network is constructed based on $\Delta(m) = (m, \delta, \omega)$. A solution is found by solving the FPRSTN defined by $RSTN_i(V_1,A_i)$ with $\Delta(m) = (m, \delta, \omega)$.

![Figure A1. (a) A nominal solution represented in $STN_1(V_1,A_1)$; (b) The nominal flows of the second type of tasks influenced by the resource failure represented in $STN_2(V_2,A_2)$.](image)

| Resource Type | No. of Resources Failed | Operation | Start Period | End Period |
|---------------|-------------------------|-----------|--------------|------------|
| 1             | 1                       | 1         | 1            | 2          |

As there are two types of products involved in the nominal solution, the algorithm first finds the set $\mathcal{J}(\Delta(m))$ of process types that will be influenced due to the failure. As both types of processes are involved with the use of the failed resource during the failure interval, therefore, $\mathcal{J}(\Delta(m)) = \{1, 2\}$ in Algorithm 3. By solving the FPRSTN defined by $RSTN_j(V_j,A_j)$ with residual capacity constraint for $j = 1$, a solution is obtained as follows: $f_{1,14} = 1, f_{14,25} = 1$ and $f_{25,36} = 1$. Figure A2a shows the above solution represented in $RSTN_1(V_1,A_1)$ by rerouting the flows in Figure A1a. This solution satisfies the condition of Property 3. By solving the FPRSTN defined by $RSTN_j(V_j,A_j)$ with residual capacity constraint for $j = 2$, the solution with $f_{6,18} = 1, f_{18,29} = 1$ and $f_{29,40} = 1$ in Figure A2b is obtained by rerouting the flows in Figure A1b. The above solution can meet the deadline as the flows of each arc after the deadline is zero and Property 1 holds. Figure A3a and Figure A3b show the overall solution obtained by combining the solution (rerouted flows) with the uninfluenced flows for both types of products. The order deadline can be met by this solution as the flows of each arc after the deadline is zero.
Figure A2. (a) Rerouted flows due to resource failure in $RSTN_1(V_1, A_1)$; (b) Rerouted flows due to resource failure in $RSTN_2(V_2, A_2)$.

Figure A3. (a) A solution represented in $STN_1(V_1, A_1)$ for the first type of tasks obtained after resource failure; (b) A solution represented in $STN_2(V_2, A_2)$ for the second type of tasks obtained after resource failure.

References

1. Syed, A.S.; Sierra-Sosa, D.; Kumar, A.; Elmaghraby, A. IoT in Smart Cities: A Survey of Technologies, Practices and Challenges. *Smart Cities* 2021, 4, 429–475.
2. National Institute of Standards and Technology. Available online: https://www.nist.gov/el/cyber-physical-systems (accessed on 21 February 2019).
3. Cardin, O. Classification of cyber-physical production systems applications: Proposition of an analysis framework. *Comput. Ind. 2019*, 104, 11–21.
4. Vodyaho, A.; Abbas, S.; Zhukova, N.; Chervoncnev, M. Model Based Approach to Cyber–Physical Systems Status Monitoring. *Computers* 2020, 9, 47.
5. Hu, F.; Lu, Y.L.; Vasilakos, A.V.; Hao, Q.; Ma, R.; Patil, Y.; Zhang, T.; Lu, J.; Li, X.; Xiong, N.N. Robust Cyber-Physical Systems: Concept, models, and implementation. *Future Gener. Comput. Syst. 2016*, 56, 449–475.
6. Ali, N.; Hong, J.E. Failure Detection and Prevention for Cyber-Physical Systems Using Ontology-Based Knowledge Base. *Computers* 2018, 7, 68.
7. Shin, D.; He, S.; Zhang, J. Robust, Secure, and Cost-Effective Design for Cyber-Physical Systems. *IEEE Intell. Syst.* 2020, 29, 66–69.
8. Hsieh, F.S. A Dynamic Context-Aware Workflow Management Scheme for Cyber-Physical Systems Based on Multi-Agent System Architecture. *Appl. Sci. 2021*, 11, 2030.
9. Berthomieu, B.; Diaz, M. Modeling and verification of time dependent systems using time Petri nets. *IEEE Trans. Softw. Eng.* 1991, 17, 259–273.
10. Hadjidje, R.; Boucheneb, H. Efficient Reachability Analysis for Time Petri Nets. *IEEE Trans. Comput.* 2011, 60, 1085–1099.
11. Model Driven Architecture. Available online: https://www.omg.org/mda/ (accessed on 9 May 2021).
12. Mejia, G.; Lefebvre, D. Robust scheduling of flexible manufacturing systems with unreliable operations and resources. *Int. J. Prod. Res.* 2020, 58, 6474–6492.
Akshay, S.; Hélouët, L.; Jard, C.; Lime, D.; Roux, O.H. Robustness of Time Petri Nets under Architectural Constraints. In Formal Modeling and Analysis of Timed Systems. FORMATS 2012; Lecture Notes in Computer Science; Jurdziński, M., Ničković, D., Eds; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7595, pp. 11–26.

Akshay, S.; Hélouët, L.; Jard, C.; Neyrier, P.A. Robustness of Time Petri Nets under Guard Enlargement. Fundam. Inform. 2016, 143, 207–234, doi:10.3233/FI-2016-1312.
45. André, É.; Pellegrino, G.; Petrucci, L. Precise Robustness Analysis of Time Petri Nets with Inhibitor Arcs. In *Formal Modeling and Analysis of Timed Systems;* FORMATS 2013; Lecture Notes in Computer Science; Braberman, V., Fribourg, L., Eds.; Springer: Berlin/Heidelberg, Germany, 2013; Volume 8053, pp. 1–15.

46. Mayr, E.W. An Algorithm for the General Petri Net Reachability Problem. *SIAM J. Comput.* 1984, 13, 441–460.

47. Lipton, R.J. The Reachability Problem requires exponential space. In *Department of Computer Science;* Research Report 62; Yale University: New Haven, CT, USA, 1976.

48. Ali, N.; Hussain, M.; Hong, J.E. Analyzing Safety of Collaborative Cyber-Physical Systems Considering Variability. *IEEE Access* 2020, 8, 162701–162713.