The University of Kansas

Information and Telecommunication Technology Center

Technical Report

On Concurrent Events and Correctness in Stochastic Models

Daniel D. Deavours

ITTC-FY2003-TR-03050-01

December 2002

Copyright © 2002:
The University of Kansas Center for Research, Inc.,
2335 Irving Hill Road, Lawrence, KS 66044-7612.
All rights reserved.

Abstract

In performance / dependability modeling, it sometimes happens that two events are scheduled to occur at the same time. In this case, the modeler has two alternatives: specify an ordering, (perhaps deterministically or probabilistically), or leave the order unspecified. If the order is unspecified, it is assumed that the order does not matter, or that the events can happen concurrently (simultaneously). It is important to be able to express concurrency, but the ability to express concurrency may also lead to the specification of ambiguous models. There are a number of checks to determine whether models are or may be ambiguous (usually implying an incorrectly specified model) with different thresholds of ambiguity. We give a survey of a number of techniques, describing their relative usefulness and characteristics, and then provide the theoretical foundation for an efficient implementation of a particularly useful check called the well-specified check.

1 Introduction

One of the important results of Einstein's theory of relativity is that simultaneity is relative to the observer. Two events may occur simultaneously with respect to one observer, but occur in some order according to another observer, and a different order according to a third observer. Thus, a fundamental aspect of nature is that simultaneity and arbitrary order is relative to the observer. In the study of discrete event systems, computer and communication systems in particular and especially distributed systems, the same principle applies.

Engineers who build complex systems want to have an understanding of the systems, including how fast and how dependable the system will perform its function. To achieve this, people build system models, which are some mathematical representation of an abstraction of the system's behavior. Models can be expressed in precise languages, called *formalisms*. Examples of formalisms include a family of queueing network formalisms (used for performance models) [1, 2], stochastic Petri nets (used for performance and dependability analysis) [3, 4], stochastic process algebras (also for performance and dependability analysis) (e.g., [5]), and fault trees (for dependability analysis) [6]. Different formalisms have different strengths, and often allow the modeler to reason formally about the model, and hence the system. For example, we may be able to formally reason that a model in some formalism never enters a deadlocked state, and therefore if the model accurately reflects the behavior of the system, the system is deadlock free.

One of the weaknesses of using formalisms is that they can be tedious to represent complex behavior. (An extreme example of this might be to try to write a word processor using a Turing machine.) For sophisticated and detailed models, engineers rely on simulation languages. While these often compromise the formal nature of formalisms, they offer in return compact representations of complex behavior. The class of simulation languages has been formally characterized using a formalism called *generalized semi-Markov process*, or GSMP [7, 8].

The particular aspect of simultaneity, sometimes called concurrency of discrete events, that we are referring to is that of two or more discrete events that occur at the same time. Here, an event can be thought of as a change in system state at a discrete point in time. One often neglected aspect of modeling is the ability to express this form of concurrency, that is,

models should to be able to express two events that may happen at the same time, or due to relativity, events in which the ordering is arbitrary. This is frequently simply not allowed in many modeling formalisms because allowing that may lead to some difficult problems (that we shall address). Leaving the order of concurrent discrete events unspecified may or may not lead to ambiguous behavior. Concurrency may lead to ambiguity, or it may not, and checking whether this is the case can be a difficult problem. Historically, this problem has largely been ignored and formalisms simply do not allow this form of concurrency.

For a simple example, if lightning strikes an aircraft, then certain electronic components may fail, leading to a failure of a non-critical system. The time between when the lightning strikes and components fail may depend exactly where the lightning strikes, but could theoretically be measured, but for reliability assessment, the time could be safely approximated as zero. The precise order in which the components fail may be irrelevant if the results are the same. In another example, two software threads may modify a common variable. The time it takes to add one to a variable is negligible compared to other delays, and the logical steps can be broken down to 1) read from the memory location, 2) add one to the value, and 3) write to the memory location. If one thread adds 1 to the variable, and concurrently, another thread sets the value to zero, then the result is ambiguous behavior. The precise order in which each step occurs does matter. This is a trivial example where being able to express concurrent events and analyze whether the resulting behavior is ambiguous would be quite useful. As models become more complex, this ability becomes more important.

In general, there have been three ways to describe how instantaneous events could be ordered: deterministically, probabilistically, or non-deterministically. It's interesting to note that very few formalisms, such as GSPNs and extensions, allow the expression of all three of these ways. Even GSMPs, for example, have been carefully crafted to only allow deterministic or probabilistic ordering. One of our contributions here is to modify GSMPs to allow for non-deterministic orderings in a way that can be useful for modelers.

Ambiguity is easy to detect in a model, but whether the ambiguity leads to ambiguous behavior in some measurable way is in general computationally expensive to detect, so we restrict ourselves primarily to ambiguity that might arise from two simultaneous discrete events. This can occur because two processes start at exactly the same time (e.g., they become enabled

from the same event) and take identical time to complete. When the process completes, the state of the model is altered to reflect the completion of the process. Another source of simultaneous events is in processes take zero time to complete. Zero-timed processes can be an approximation to processes that take very little time relative to other events in the model, or they can be one part of a succession of zero-timed processes that reflects how a model changes after a timed process completes, such as steps in an algorithm used to compute the next state.

In this paper, we begin by reviewing previous work in this area. Previous work has often focused on simultaneous events within a particular formalism, and comparing different approaches has been hindered because of the differences in formalisms. To alleviate this, we develop two formalisms in which we believe we can capture the relevant aspects of all the approaches we review. This gives us a common reference to which we can compare different approaches. It also allows us to reason formally using a common notation. We focus primarily on two approaches which we believe to be the most useful and efficient. One important result that we show here, rigorously, is that both approaches are computationally efficient.

2 Background

As we mentioned in the introduction, there are a number of formalisms that are used for performance and dependability evaluation. We will focus on a small set of these, namely stochastic process algebras (SPAs) and various extensions to stochastic Petri nets (SPNs). Both of these classes of formalisms allow for concurrent behavior, and thus, the possibility of concurrent events. Analysis of SPAs and SPNs often involve converting the model into a continuous or discrete time Markov chain and performing some numerical analysis on the Markov chain. Thus, delays in these formalisms are exponentially distributed, but they also may be immediate, i.e., take zero time. Multiple processes that take zero time to complete may be initiated by a single event, so the events corresponding to the completion of the immediate processes are concurrent. This is one of the sources of concurrent events in SPAs and SPNs.

Some extensions of SPAs and SPNs allow non-exponential delays such as deterministic times, or discrete distributions such as a geometric. Again, if two processes are started by a single event, then there is some probability that two or more processes may complete at the same time, and the corresponding events are simultaneous. In general, non-exponential delays do not translate into Markov chains (or rather, Markov chains with a countable state space), and they are generally classified under the much more general GSMP formalism. In later sections, we will examine GSMPs in greater detail.

What makes this work unique is that the models contain both timed and untimed (or more specifically, zero-timed) behavior. Among the SPA community, there have been considerable analysis of untimed probabilistic and non-deterministic systems with concurrent events (e.g., [11]). Only recently have SPA formalisms considered a mix of time, untimed probabilistic, and untimed non-deterministic behavior. MoDeST [12] is an example of a recent formalism that can express this type of behavior.

Stochastic Petri nets [4] were originally developed as a simple timed extension to Petri nets. It became apparent rather quickly that adding timed and immediate behavior was useful, and generalized stochastic Petri nets (GSPNs) were formed [3]. The state-changing process of a SPN is called a transition (graphically depicted as vertical bars). Transitions may fire (change model state) if they are enabled (according to some condition), and the delay between when

a transition becomes enabled and fire can be exponentially distributed or immediate. The transition is labeled *timed* or *immediate* accordingly.

Originally, probability distribution over competing immediate transitions were assigned by the user during a state space exploration, and hence concurrency was always probabilistically resolved. This was a tedious process, so GSPNs were subsequently revised. The revision allowed for an automatic partitioning of immediate transitions into extended conflict sets (ECSs) [3]. Because of the structure of a GSPN, a model could be analyzed so that competition among immediate transitions that belonged to different ECSs were "guaranteed" not to cause any ambiguities. Probabilistic resolution of concurrency was only required for resolving immediate transitions in the same ECS. This was later made even more formal in [13]. The primary advantage of using ECSs are stated as: "to allow a simple specification of random switches [immediate transitions], independently of the knowledge of the reachable markings" [13]. Thus, ECSs can be formed by a static analysis of the model.

There are a number of drawbacks with ECSs. First, the analysis of the GSPN to form ECSs is uses a sufficient, but not necessary condition. This is a relatively minor drawback, but it will sometimes lead to ECSs that are unnecessary. Most significantly, the construction of ECSs are closely tied to the structure of GSPNs. Different extensions that relax the structure such as stochastic reward nets [14, 15] and stochastic activity networks [16], can not use ECSs. Finally, recently a flaw has been found in the construction of ECSs that may still lead to ambiguity [17]. Reliance on ECSs to detect possible ambiguity is indeed not sufficient, and it is likely that a new revision of GSPNs will be necessary to fix this flaw. The fact that it has taken years to identify this flaw is an indication of how tricky this problem is, and how important it is to use rigorous methods. ([13] outlines a proof that is clearly incorrect.)

Concurrent to the development of GSPNs, stochastic activity networks (SANs) [16, 18] were developed. SANs also have timed activities, which are similar to timed GSPN transitions, and instantaneous activities, which are analogous to immediate GSPN transitions. Each SAN activity can have cases, which probabilistically describes different possible behaviors upon completion of the process. Among multiple immediate activities that can fire, the order in which they fire is left unspecified, and so long as the timed or measurable behavior of the model does not change, the model is said to be *well-specified*. (We define this more precisely

and formally in following sections.) While the definition of well-specified was given as early as 1988 [19], an algorithm to perform the check was not published until 1995 [20]. Under modest assumptions, a more efficient algorithm is also possible ([21], refined in later sections of this paper). In this paper, we make several new contributions related to well-specified, including the existence of an efficient algorithm that works without any assumptions.

Another approach has been through the use of the concept of *well-defined* [22]. Well-defined itself is a very strict condition and does not allow the expression of concurrent events. However, also defined in [22] is *well-defined with respect to a reward structure*, which does allow ambiguity so long as it is not measured by the reward structure. (This will be described formally and in greater detail in following sections.) The authors propose an algorithm to check for a sufficient condition for a model to be well-defined with respect to a reward condition. This check, it turns out, is quite similar to the well-specified check. In fact, under modest assumptions, they are identical. We prove this in Section 3.4.

The advantage of using the well-specified and well-defined checks have is that they can be applied efficiently. They can be applied as a model is executed, either by simulation, or by a state-space analysis. The disadvantage is that, unlike the (flawed) ECS approach, is that it cannot be applied structurally.

GSMPs have historically not considered ambiguity at all. The order of concurrent discrete events must either be deterministically or probabilistically specified; ambiguity of order, a natural result of simultaneity, is not allowed. Perhaps this is due to the historical need for efficient computer simulation. However, we show that this ambiguity can be allowed and checked with little overhead.

In the following section, we begin to present the concepts in a formal way. Because well-specified and well-defined were developed in different contexts with slightly different subtle differences, we define a representation that both concepts can be applied to equally well. We begin by describing that representation and various terminology that will help in presenting the definitions and implications. The formality is necessary to truly understand the distinctions between well-specified and well-defined, and be confident that the various algorithms we present are correct.

3 Terminology

The variety of formalisms and subtle variants and differences makes it difficult to chose a particular formalism to apply these checks in a uniform way. For example, the term *well-specified* has been applied specifically to SANs [16], while *well-defined* has been applied to GSPNs [3]. Directly comparing the two, formally, can be difficult. Furthermore, to our knowledge, no equivalent definition has been applied to GSMPs.

Our solution is two-fold. First, for GSPNs, SANs, and related formalisms such as SPAs, we develop a new formalism, which we call the *intermediate formalism representation* (IFR). For our purposes, IFR generalizes, and hence abstracts away unimportant aspects of the execution behavior of a formalism, while retaining other aspects of the execution behavior that is essential to our analysis. Thus, we can be rigorous and uniform in our application of the definitions and proofs.

Furthermore, we believe that using IFR is not a limitation. We argue (without proof, although one would be straightforward) that any GSPN or SAN model has a straightforward mapping into an equivalent IFR model. Furthermore, the definitions and proofs can be extrapolated in a straightforward manner to the other formalisms, some of which we will name later. This gives a nice mix of rigor and generalization.

We take a different approach with GSMPs. Since they are significantly different from SPNs and extensions, and considerably more sophisticated, we treat them separately. We take a similar tact in creating a simplified GSMP formalism that (we believe) retains all the essential elements of GSMPs, but is syntactically more similar to IFR. Again, this allows us a similar mix of a simpler formalism for easier rigorous application of definitions and proofs, as well as generalization.

3.1 Intermediate Formalism Representation

Definition 1. *An* Intermediate Formalism Representation *(IFR) is a tuple:* $\text{IFR} = (\Sigma, \text{T}, \text{En}, \phi, \text{W}, \text{PG}, \mu_0)$

- Σ*, a countable set of markings;*
- $\text{T} = \text{T}^T \cup \text{T}^I$*, set of timed and immediate transitions;*

- $\mathrm{En} : \Sigma \times \mathrm{T}$, *the enabled relation;*
- $\phi : \Sigma \times \mathrm{T} \rightarrow \Sigma$, *the transition firing partial function;*
- $\mathrm{W} : \Sigma \times \mathrm{T} \rightarrow \mathbb{R}^{>}$, *the weight/rate partial function;*
- $\mathrm{PG} : \Sigma \rightarrow 2^{2^{\mathrm{T}_I}}$, *the transition partition function;*
- μ_0, *the initial marking.*

A transition $\mathrm{t} in \mathrm{T}$ *is said to be* enabled *in marking* $\mu \in \Sigma$ *if* $(\mu, \mathrm{t}) \in \mathrm{En}$. *The fire function* $\phi(\mu, \mathrm{t})$ *is defined only for transitions* t *that are enabled in marking* μ. *Similarly,* $\mathrm{W}(\mu, \mathrm{t})$ *is defined only transitions* t *that enabled in marking* μ. *The transition partition function partitions the immediate transitions in each marking. I.e., if* $\mathrm{PG}(\mu) = \{g_1, g_2, \ldots, g_k\}$ *where* $g_i \subseteq \mathrm{T}_I$, *then* $\cup_i g_i = \mathrm{T}_I$, *and* $\cap_i g_i = \emptyset$.

We claim that the IFR can accept any SPN or related formalism, so it is sufficient to prove properties on IFR. It is then straightforward to apply the results to a specific formalism.

The IFR formalism specifies a probabilistic automata, and the general idea behind the execution of the IFR is as follows (this will be given more formally below). Transitions are either immediate or timed. When a timed transition is enabled, then after some exponential time (given with rate W), the transition will "fire" and the model will change marking according to the firing function ϕ. If an immediate transition becomes enabled, then it will fire immediately, i.e., in zero time. If multiple immediate transitions become enabled simultaneously, then immediate transitions are selected probabilistically among transitions of the same partition group. Among immediate transitions of different partition groups, the selection is left undefined.

The key construct to allow non-determinism is the immediate partition group PG. The concept behind this is that immediate transitions of the same partition group compete probabilistically using the weight function. Among immediate transitions of different partition groups, the order is left completely unspecified. This combination of ordering within groups and non-ordering among partitions allows us to express any combination of deterministic ordering (through enabling), probabilistic ordering (through weights), and non-deterministic ordering (among partition groups).

Before we can proceed, we must introduce some notation for convenience. Let $EN_\mu = \{t \in \mathrm{T} : (\mu, t) \in \mathrm{En}\}$ be the set of *enabled* transitions. Similarly, let $EN^I_\mu = EN_\mu \cap \mathrm{T}_i$ and $EN^T_\mu = EN_\mu \cap \mathrm{T}_T$ be the set of enabled immediate and timed transitions, respectively. Furthermore, we can define $EN^*_\mu = EN^I_\mu$ if $EN^I_\mu \neq \emptyset$, or EN^T_μ otherwise. The symbol $EN^*_\mu = EN^I_\mu$ gives us the set of enabled transitions in a marking that are competing to fire first; immediate transitions will always fire before timed transitions.

Assume first that a IFR model contains no non-specified simultaneous discrete events. This occurs when $\forall \mu \in \Sigma$, $\exists g \in \mathrm{PG}(\mu)$ such that $g \cap EN^I_\mu = EN^I_\mu$, i.e., all enabled immediate transitions belong to the same immediate partition group. Then a model M uniquely defines a stochastic process SP. Let $\mathrm{SP} : \Omega \times \mathbb{N} \to \mathrm{T} \times \Theta \times \Sigma$ be the stochastic process a IFR model M. A sample path of the SP, $\mathrm{SP}|\omega$, defines a sequence of events. We simplify by letting $E : \mathrm{T} \times \Theta \times \Sigma$ be an event. Then we can write $\mathrm{SP} : \Omega \times \mathbb{N} \to E$. For $\epsilon_i \in E$, $\epsilon_i = (\mathrm{t}_i, \theta_i, \mu_i)$, t_i is the transition (immediate or timed) that fires, θ_i is the transition firing time, and μ_i is the resulting marking. We write $\mathrm{SP}|n$, for $n \in \mathbb{N}$, to be the n-th event of the stochastic process.

To add concurrency, we can use the original definition of IFR and simply relax the condition that enabled immediate transitions belong to the same partition group. Then the order in which immediate transitions complete is left unspecified, and the assumed intention is that the order is arbitrary in some sense. Next, we describe the execution policy for IFR models.

First, notice that a IFR model $M \not\mapsto \mathrm{SP}$, but rather, a model can specify a number of possible mappings. In order to describe these possible mappings, we use a scheduler. A *scheduler* s is a mapping $s : \Sigma \to 2^{2^{\mathrm{T}}}$ where $s(\mu) \in \mathrm{PG}(\mu)$, and indicates which partition group is selected in a marking μ so that transitions of that partition group will fire before transitions of other partition groups. Thus, a IFR model M and a scheduler s does define a stochastic process. Let S_M be the set of valid schedulers for a model M. We denote the particular stochastic process defined by M and s as SP_s, and we denote the family of stochastic processes possibly defined by the model as $\mathrm{SP}_{S_M} = \{\mathrm{SP}_s : (M, s) \mapsto \mathrm{SP}_s, s \in S_M\}$. We can then ask the question: do all $\mathrm{SP}_s \in \mathrm{SP}_{S_M}$ define an "equivalent" stochastic process in some sense? If this is the case, then the orderings are safely arbitrary; otherwise, the orderings lead to ambiguity.

The execution policy formally defines how the model evolves, and hence describes the

$EP(\mu_i, \theta_i, s) \mapsto (\mathrm{t}_{i+1}, \theta_{i+1}, \mu_{i+1})$

Begin

$$\text{Let } \mathrm{T}^* = \begin{cases} s(\mu_i) \cap EN^I_{\mu_i} & EN^I_{\mu_i} \neq \emptyset\,, \\ EN^T_{\mu_i}\,, & \text{otherwise}\,. \end{cases}$$

Let $\mathrm{t}_{i+1} \in \mathrm{T}^*$ be selected with probability $\frac{\mathrm{W}(\mu_i, \mathrm{t}_{i+1})}{\sum_{\mathrm{t}_j \in \mathrm{T}^*} \mathrm{W}(\mu_i, \mathrm{t}_j)}$

$$\theta_{i+1} = \begin{cases} \theta_i & EN^I_{\mu_i} \neq \emptyset\,, \\ \theta_i + \mathbf{X}_i \sim \exp(\mathrm{W}(\mu_i, \mathrm{t}_{i+1})) & \text{otherwise}\,. \end{cases}$$

$\mu_{i+1} = \phi(\mu_i, \mathrm{t}_m)$

End

Figure 1: Execution policy for IFR.

stochastic process. The execution policy describes how, given the model enters some marking μ_i at time θ_i, the subsequent marking μ_{i+1} is reached by firing transition t_{i+1} at time θ_{i+1}. The execution policy is given in Figure 1. Note the use of the notation $\theta_{i+1} = \theta_i + \mathbf{X} \sim \exp(\mathrm{W}(\mu_i, \mathrm{t}_{i+1}))$; this means that θ_{i+1} is the previous firing time θ_i plus some exponential random variable ($\mathbf{X}_i$) with parameter $\mathrm{W}(\mu_i, \mathrm{t}_{i+1})$. I.e., let $\lambda = \mathrm{W}(\mu_i, \mathrm{t}_{i+1})$. Then $F_{\mathbf{X}_i}(t) = 1 - e^{-\lambda t}$. The random variables $\cup_i \{\mathbf{X}_i\}$ are independent. Note that the stochastic process is time-homogeneous.

The execution policy formally describes the stochastic process $\mathrm{SP}_s : \Omega \times \mathbb{N} \to E$. From this, we can induce a Borel space and a unique probability measure P. It is possible to state precisely what the Borel space and measure function is precisely, but such formality is unnecessary for our purposes here. Let $\Pr[e|s] \equiv P\{\omega \in \Omega : \mathrm{SP}_s|\omega \text{ satisfies } e\}$. For example, $\Pr[\mu_i \to \mu_j|s]$ is shorthand for writing $P\{\mu_j \in EP(\mu_i, \theta, s)$ (it is independent of θ). Furthermore, let $\Pr[e]$ be *undefined* if $\Pr[e|s]$ is not identical for all $s \in S_M$. Thus, $\Pr[\mu_i \to \mu_j]$ is undefined if $\Pr[\mu_i \to \mu_j|s_1] \neq \Pr[\mu_i \to \mu_j|s_2]$ for some $s_1, s_2 \in S_M$. If $\Pr[\mu_i \to \mu_j|s]$ is identical for all $s \in S_M$, then $\Pr[\mu_i \to \mu_j] \equiv \Pr[\mu_i \to \mu_j|s]$ for any $s \in S_M$. Also, note that $\Pr[\mu_1 \to \mu_2, \mu_2 \to \mu_3, \cdots, \mu_i \to \mu_{i+1}] = \Pr[\mu_1 \to \mu_2, \mu_2 \to \mu_3, \cdots, \mu_{i-1} \to \mu_i] \Pr[\mu_i \to \mu_{i+1}]$.

The concept of a defined or undefined probability, e.g., $\Pr[\mu_i \to \mu_j]$, already gives us

some formal way of measuring ambiguity. For example, we may require that a model have no undefined probabilities of the form $\Pr[\mu_i \rightarrow \mu_j]$. Another requirement might be that probabilities may be undefined so long as the states involved in the undefined probabilities are somehow equivalent or indistinguishable. A third might be that the probability of a model being in a state at some time must be defined, but the intermediate states the model is in for zero time may be undefined.

A fourth requirement might allow undefined probabilities so long that some measure of interest is defined. Let $R : \mathrm{SP} \rightarrow \mathbb{R}$ be a measure of the stochastic process. Typically, measures are constructed using a reward structure with other timing-related information. Thus, a requirement may be that for a measure R, $R(\mathrm{SP}_s)$ must be identical for all $s \in S_M$. This is the condition *well-defined with respect to measure M* [22].

Another requirement focuses only on component of a measure, impulse rewards. An impulse reward is a function, $\imath : \Sigma \times \mathrm{T} \times \Sigma \rightarrow \mathbb{R}$, which accumulates reward at each transition firing. Given the model is in marking μ_i, transition t fires, and the resulting marking is μ_{i+1}, the impulse reward is given as $\imath(\mu_i, \mathrm{t}, \mu_{i+1})$. The measure of ambiguity states that the timed behavior of the stochastic process must be defined, and the untimed behavior can be ambiguous so long as the probability of obtaining an impulse reward is defined. This is the *well-specified* condition [19].

Each measure of ambiguity leads to a different check, and hence different algorithms for performing the check. Some of these checks are significantly more computationally costly than others. The remainder of this paper, we define the well-specified and a well-defined check formally, analyzing them, and applying them to IFR and GSMP models. Unfortunately, before we can proceed, we must define some notation.

3.2 Notation

Definition 2. *We can define a pair of* goes to *($\rightarrow$) relations. Let $\rightarrow : \Sigma \times \Sigma$ be defined such that $(\mu_i, \mu_{i+1}) \in \rightarrow$ if $\exists \mathrm{t} \in EN^*_{\mu_i}$ such that $\mu_j = \phi(\mu_i, \mathrm{t})$, and we write $\mu_i \rightarrow \mu_j$. Let $\stackrel{s}{\rightarrow} : \Sigma \times \Sigma$ be defined such that $(\mu_i, \mu_{i+1}) \in \stackrel{s}{\rightarrow}$ if $\exists \mathrm{t} \in EN^*_{\mu+i} \cap (\mathrm{T}_T \cup s(\mu_j))$ such that $\mu_{i+1} = \phi(\mu_i, \mathrm{t})$, and we write $\mu_i \stackrel{s}{\rightarrow} \mu_{i+1}$.*

Definition 3. *We can define the set of* successor markings *of μ_i, and we write* $\mathrm{SM}(\mu_i)$*, as follows:* $\mathrm{SM}(\mu_i) = \{\mu_j : \mu_i \rightarrow \mu_j\}$.

Definition 4. *A marking μ_i is called a* stable marking *if $EN^I_{\mu_i} = \emptyset$, i.e., if no immediate transitions are enabled in marking μ_i. We write $\hat{\mu}_i$ to mean that marking μ_i is a stable marking. We denote $\hat{\Sigma} \subset \Sigma$ to be the subset of all markings that are stable markings.*

Definition 5. *A marking μ_i is called an* unstable marking *if $EN^I_{\mu_i} \neq \emptyset$, i.e., at least one immediate transition is enabled in marking μ_i. We write $\bar{\mu}_i$ to mean that marking μ_i is an unstable marking. We denote $\bar{\Sigma} \subset \Sigma$ to be the subset of all markings that are unstable markings.*

Definition 6. *An unstable marking $\bar{\mu}_i$ is called an* unscheduled marking *if $|\{g \in \mathrm{PG}(\bar{\mu}_i) : g \cap EN^I_{\mu_i} \neq \emptyset\}| > 1$, that is, if in marking $\bar{\mu}_i$ multiple immediate transitions belonging to different partitions are enabled. A marking is* scheduled marking *if it is not an unscheduled marking.*

Definition 7. *Let $\rightsquigarrow$: $\Sigma \times \Sigma \times S_M$ be a relation, written $\mu_i \rightsquigarrow \mu_j|s$. The relation $\mu_i \rightsquigarrow \mu_j|s$ holds if there exists a sequence of markings $\mu_i, \mu_{i+1}, \cdots, \mu_{j-1}, \mu_j$, such that $\mu_i \xrightarrow{s} \mu_{i+1}, \mu_{i+1} \xrightarrow{s} \mu_{i+2}, \cdots, \mu_{j-1} \xrightarrow{s} \mu_j$, and $\mu_{i+1}, \cdots, \mu_{j-1} \in \bar{\Sigma}$. We write $\mu_i \rightsquigarrow \mu_j$ if $\mu_i \rightsquigarrow \mu_j|s, \forall s \in S_M$.*

Informally, if μ_i is an unstable marking, then μ_i leads to μ_j only if μ_j is reachable by firing only enabled immediate transitions. If μ_i is a stable marking, then μ_i leads to μ_j only if μ_j is reachable from marking μ_i by firing at most one enabled timed transition followed by any number of enabled immediate transitions.

For convenience, we write $\mu_i \rightsquigarrow \mu_j$ as short for $\mu_i \rightsquigarrow \mu_j|S_M$; we write $\mu_i \rightsquigarrow \mu_j|s$ as short for $\mu_i \rightsquigarrow \mu_j|\{s\}$; and we write $\mu_i \rightsquigarrow \mu_j|s(\mu_k) = g$ as short for $\mu_i \rightsquigarrow \mu_j|\{s \in S_M : s(\mu_k) = g\}$.

Definition 8. *Let the set of* next stable marking *of a marking μ_i, written* $\mathrm{NM} : \Sigma \rightarrow \Sigma$*, be defined as* $\mathrm{NM}(\mu_i) = \{\hat{\mu} \in \hat{\Sigma} : \mu_i \rightsquigarrow \hat{\mu}\}$.

Definition 9. *For some marking μ_0, the set of* immediately reachable states *is given by* $\bar{\Sigma}^{\mu_0} = \{\mu \in \hat{\Sigma} : \mu_0 \rightsquigarrow \mu\}$.

3.3 Well-Defined and Well-Specified

3.3.1 Well-Defined

We now have enough notation that we can give formal definitions for different criteria for ambiguity for concurrent events in IFR models. We begin with the well-defined condition, which we will call *the first well-defined condition*, given in [22], adapted to apply to IFR models.

Definition 10. *Let M be a IFR model, and let $\epsilon_0, \ldots, \epsilon_k$ be a sequence of events generated by M, where $\epsilon_i = (\mathrm{t}_i, \theta_i, \mu_i)$. A model M is* well-defined *if $\forall n \in \mathbb{N}$, $\forall \mathrm{t} \in T$, $\forall \mu \in \Sigma$, and $\forall \theta \in \mathbb{R}^{\geq}$, $P\{\mathrm{t}_i = \mathrm{t}, \theta_i \leq \theta, \mu_i = \mu\}$ is defined.*

Note that this definition is quite restrictive. It implies that the stochastic process should not have any measurable ambiguity. If you consider the stochastic process as an indexed set of random variables, then for each $n \in \mathbb{N}$, $\{\mathrm{SP}_s | n : s \in S_M\}$ yields a set of random variables that must be identical in distribution.

This condition does not leave any room for any ambiguity. Thus, if $|\mathrm{SP}_{S_M}| > 1$, then the model is not well-specified. If somehow for SP_1, $\mathrm{SP}_2 \in \mathrm{SP}_{S_M}$, SP_1 and SP_2 are identical except that in some marking μ, scheduler 1 defines a model where an immediate transition is not enabled, and scheduler 2 defines a model where the immediate transition is enabled but with weight zero, then the model would still be well-defined. However, because of the way IFR models are constructed, this scenario is not possible, and therefore for IFR models, well-defined implies that $|\mathrm{SP}_{S_M}| = 1$.

3.3.2 Well-Defined With Respect To a Reward Variable

It might be the case that there could be some ambiguity in the stochastic process, but the ambiguity is so slight that it does not have any impact on the measure of interest. For example, ambiguity may lead to a repair being performed by an unspecified repairperson, but as long as the repair process is statistically identical among repairpeople and the model is not intended to measure repairs of a particular repairperson, then the ambiguity is acceptable. This is what a second definition of *well-defined* addresses (again, adapted to IFR models).

Definition 11. *Let M be a IFR model, and let $\epsilon_0, \ldots, \epsilon_k$ be a sequence of events generated by M. Let S_M be the set of possible schedulers for M, and let SP_{S_M} be the set of stochastic processes defined by (M, s). Let R be some measure of the stochastic process. M is* well-defined with respect to the reward variable R *if for the set of stochastic processes defined by M, SP_{S_M}, $\Pr[R(SP_s) < x]$ is identical for all $s \in S_M$, and for all $x \in \mathbb{R}$.*

This has an interpretation that, given a reward variable R, any ambiguity that occurs does not affect R in any measurable way. Recall that the reward variable $R : (\Omega \times \mathbb{N} \to E) \to (\Omega \to \mathbb{R})$ maps a stochastic process to a random variable. Let X_s be the random variable defined by R for the stochastic process SP_s. Then $\{X_s : s \in S_M\}$ defines a set of random variables, and for the model to be well-defined with respect to a reward condition R, then the set of random variables $\{X_s\}$ must be identical in distribution.

The well-defined with respect to a reward variable is perhaps the "loosest" definition in allowing ambiguity, and hence the most general. More ambiguity would render the model ambiguous so that the measure of interest is ambiguous. Note that this still has considerable utility, but this is outside the scope of this problem. Our problem focuses on whether the model is sufficiently specified so that we can learn unambiguous information about the model.

Well-defined with respect to a reward variable allows for significant amounts of ambiguity. For example, if t_1 and t_2 are immediate transitions enabled in the same marking and belonging to different partition groups, firing t_1 or t_2 resulted in the same next marking, and the reward variable does not measure the firing of t_1 or t_2, then the ambiguity is allowed under this condition. Even if t_1 or t_2 yield different next states, as long as the next states are somehow equivalent with respect to the reward variable, for example, if the two next states are bisimilar, then the ambiguity is allowed under this condition. However, this condition is significantly more general than the bisimilarity condition.

3.3.3 Sufficient Well-Defined Checks

The problem with the definition of well-defined with respect to a reward condition is that a check for this would, in general, require computing a solution the reward variable $|S_M|$ times, potentially very costly since $|S_M|$ could grow exponentially in $|\Sigma|$. Since truly exact solutions are often computationally impractical or impossible, approximate solutions would yield small

differences in the reward variable solution, it is impractical to determine whether small changes are due to the approximations or ambiguities. Thus, the well-defined with respect to a reward variable condition is impractical to implement.

The authors of [22] take a pragmatic approach and decide instead to provide an algorithm that is a sufficient, stricter definition of well-defined with respect to a reward condition. Instead of calling this algorithm *a sufficient condition to determine well-defined with respect to a reward variable*, from here on out, we lazily call it *well-defined-2*, or *the second well-defined condition.* The definition is given formally below.

Definition 12. *A model M is* well-defined-2 *if* $\forall \mu_i \in \Sigma$, $\forall \hat{\mu} \in \mathrm{NM}(\mu_i)$, *and* $\forall \imath \in Im$,

1. $\Pr[\mu_i \rightsquigarrow \hat{\mu}]$ *is defined, and*

2. $\Pr[\mu_i \rightsquigarrow \hat{\mu}, \imath]$ *is defined.*

This definition assumes that the only ambiguity that can occur in a model is which immediate transitions will be selected to fire. Indeed, IFR is constructed in such a way that this is the only form of ambiguity possible. There is no ambiguity about what the distribution of the firing time is, for example, as is the case for other formalisms ([12], for example).

Informally, this definition checks whether, from any marking in the marking space, the probability of reaching the next stable marking and obtaining some impulse reward is defined. Recall that $\mathrm{SP}_s : \Omega \times \mathbb{N} \rightarrow E$, so that $\mathrm{SP}_s|n$ is a random variable. For some $\epsilon \in E$ and $s_1, s_2 \in S_M$, it might be the case that $\Pr[\mathrm{SP}_{s_1}|n = \epsilon] \neq \Pr[\mathrm{SP}_{s_2}|n = \epsilon]$, or at least this is not explicitly dis-allowed. I.e., the exact nature of the sequence of events may not be identical in probability, but that is acceptable so long as the the definition is met.

There is a very limited amount of ambiguity that is tolerated under this condition. Let t_1 and t_2 be enabled immediate transitions of different partition groups in some marking μ. If the probability of selecting t_1 and t_2 from their respective partition groups is identical, $\phi(\mu, \mathrm{t}_1) = \phi(\mu, \mathrm{t}_2)$, (the result of firing t_1 and t_2 is the same), and if the reward variable does not measure t_1 or t_2 firing, then the ambiguity with regard to whether t_1 or t_2 fires is acceptable under this condition.

This definition of well-defined-2 is stricter than the definition of well-defined with respect to a reward condition, and can be best illustrated through a simple example. For some marking

μ_i, say that there are two possible next stable states: $\hat{\mu}$ and $\hat{\mu}'$, and the probability of reaching $\hat{\mu}$ and $\hat{\mu}'$ are undefined. Further, let us say that the reward variable does not distinguish between $\hat{\mu}$ and $\hat{\mu}'$, and that the future behavior of the model, given the model is in states $\hat{\mu}$ or $\hat{\mu}'$, is identical. This is the case if $\hat{\mu}$ and $\hat{\mu}'$ are bisimilar, for example. In this case, the model meets the requirement of well-defined with respect to a reward condition, but does not meet the condition of well-defined-2. Hence, some generality is lost for efficiency in performing the check.

Since the second well-defined condition allows for such a restrictive form of ambiguity, an efficient algorithm can be devised to perform this check [22].

3.3.4 Well-Specified

Another approach developed separately, which has been used exclusively on SANs, is the "well-specified" condition. First introduced in [19], it takes a slightly different approach. Formally, the definition is given below (again, adapted to IFR models).

Definition 13. *A model M is* well-specified *if* $\forall \hat{\mu}_0 \in \hat{\Sigma}$, $\forall \hat{\mu} \in \mathrm{NM}(\hat{\mu}_0)$, *and* $\forall \imath \in Im$,

1. $\Pr[\hat{\mu}_0 \rightsquigarrow \hat{\mu}]$ *is defined, and*

2. $\Pr[\hat{\mu}_0 \rightsquigarrow \hat{\mu}, \imath]$ *is defined.*

Informally, this definition is concerned about the timed evolution of the model. Inductively, if moving from a stable state to some next stable state and obtaining an impulse reward is always defined, then the model is well-specified. This is a more general definition than well-defined, and the distinction is subtle but important. For a model to be well-specified, the probability of reaching a next stable marking and obtaining an impulse reward *from a stable marking* must be defined. To be well-defined, the probability of reaching a next stable marking and obtaining an impulse reward *from any marking* must be defined.

Presumably, a model that meets the well-specified definition might have some unstable marking by which the probability of reaching some next stable marking and obtaining an impulse reward would be undefined. This is acceptable for well-specified models, but not for well-defined models. Consequently, the algorithm that was developed to determine whether a

SAN is well-specified [9, 20] was developed from the definition to enumerate all the possible paths from a stable state to the next stable state. Unfortunately, the number of paths can be exponential in the number of states involved in the set of possible paths. This yields an algorithm that is exponential in $|\Sigma^{\mu_0}|$. This is in contrast with the well-defined condition in which an algorithm exists that can perform the check that is linear in $|\Sigma^{\mu_0}|$.

An interesting result of our research is that, surprisingly (or perhaps not), the two definitions are in fact very similar. Because well-specified was defined in terms of SANs and the execution policy of SANs, while well-defined was defined in terms of GSPNs and the execution policy governing GSPNs, it was not at all clear to researchers at the time that they were so similar. In fact, under a mild condition, the two definitions are equivalent. This condition is simply this: $\Pr[\mu_i \rightarrow \mu_j] > 0$ for all $\mu_i \in \Sigma$, and for all $\mu_j \in \mathrm{SM}(\mu_i)$. In the high-level IFR model, this requirement is met if all weights are positive. I.e, there may be no transition t and marking μ such that $\mathrm{W}(\mu, \mathrm{t}) = 0$. An intuitive interpretation of this might be that a transition with zero weight (and hence zero probability of firing) should be considered as not enabled.

If there exists an immediate transition with zero weight, and hence zero probability of firing, then there exists a sample path of the stochastic process in which that transition fires, but the probability of those sample paths in which that transition fires is zero. This is key to intuitively understanding the subtle difference between the definition of well-specified and well-defined. For illustration, suppose $\Pr[\mu_i \rightarrow \mu_j] = 0$, and $\Pr[\mu_j \rightsquigarrow \hat{\mu}]$ is undefined for some $\hat{\mu} \in \mathrm{NM}(\mu_j)$. Then the model may be well-specified, but is not well-defined. The distinction is this: if μ_j is reachable with probability zero, so anything that happens from μ_j is considered irrelevant from the perspective of well-specified. However, since $\mu_j \in \Sigma$, the model is not considered well-defined.

It is not clear that one definition is better than another. The arguments about well-specified and well-defined is that if the corresponding level of ambiguity is present, then the model is not correct. Proponents of well-specified would say that ambiguity that happens with probability zero is acceptable, while well-defined proponents would argue that any ambiguity in the model is a sign that the model was constructed incorrectly. Both are valid from a perspective.

If, however, we ignore sample paths with probability zero, or correspondingly, we say that immediate transitions with zero weight are considered not enabled, then the two definitions are

in fact equivalent. In the following section we prove this formally, and is one of the contributions of this paper. Later, we present a new algorithm that accurately performs the well-specified check in linear time, also (accurately) presented here for the first time. Finally, we apply these definitions to GSMP models.

3.4 Equivalence of Well-Specified and Well-Defined

As we just mentioned, under a mild condition, the definition of well-specified and well-defined are identical.

We begin by stating three rules that are readily evident.

Rule 1. If μ_i is a scheduled marking, then $\Pr[\mu_i \rightsquigarrow \hat{\mu}] = \sum_{\mu_j \in \mathrm{SM}(\mu_i)} \Pr[\mu_j \rightsquigarrow \hat{\mu}] p_{ij}$

Rule 2. $\Pr[\hat{\mu} \rightsquigarrow \hat{\mu}] = 1$.

Rule 3. $\Pr[\mu_i \rightsquigarrow \hat{\mu} \cap \mu_i \rightsquigarrow \hat{\mu}'] = 0$, for $\hat{\mu}, \hat{\mu}' \in \mathrm{NM}(\mu_i)$, $\hat{\mu} \neq \hat{\mu}'$.

Note that we use the shorthand notation p_{ij} to mean $\Pr[\mu_i \rightarrow \mu_j]$.

Condition 1. *If a node μ_i is an unscheduled node, then $\Pr[\mu_i \rightsquigarrow \hat{\mu} | s(\mu_i) = g]$ is defined and identical $\forall g \in \mathrm{PG}(\mu_i)$, $\forall \hat{\mu} \in \mathrm{NM}(\mu_i)$.*

This leads to the first theorem.

Theorem 1. *If $\mu_i \rightarrow \mu_j$ implies that $\Pr[\mu_i \rightarrow \mu_j] > 0$, then Condition 1 is a necessary and sufficient condition for a IFR model to be well-specified with respect to the first part of the well-specified condition.*

Proof. If Condition 1 holds, then $\Pr[\mu_i \rightsquigarrow \hat{\mu}]$ is defined $\forall \mu_i \in \Sigma$, so certainly it is defined for $\mu_i \in \hat{\Sigma}$. Thus, Condition 1 is a sufficient condition.

To show that Condition 1 is a necessary condition, we assume, for contradiction, that a well-specified IFR model exists that does not meet Condition 1.

First, we note that the well-specified definition implies that $\Pr[\hat{\mu}_i \rightsquigarrow \hat{\mu}_j]$ is defined $\forall \hat{\mu}_j \in \mathrm{NM}(\hat{\mu}_i)$. Condition 1 implies that $\Pr[\mu_i \rightsquigarrow \hat{\mu}_j]$ is defined $\forall \hat{\mu}_j \in \mathrm{NM}(\mu_i)$. For a contradiction, we can assume that there exists some markings $\hat{\mu}_0$, $\bar{\mu}$, and $\hat{\mu}$ where:

1. $|\hat{\mu}_0 \rightsquigarrow \hat{\mu}| > 0$,

2. $|\hat{\mu}_0 \rightsquigarrow \bar{\mu}| > 0$,

3. $|\bar{\mu} \rightsquigarrow \hat{\mu}| > 0$,

4. $\Pr[\hat{\mu}_0 \rightsquigarrow \hat{\mu}]$ is defined, and

5. $\Pr[\bar{\mu} \rightsquigarrow \hat{\mu}]$ is undefined.

If $\Pr[\hat{\mu}_0 \rightsquigarrow \hat{\mu}]$ is defined but $\Pr[\bar{\mu} \rightsquigarrow \hat{\mu}]$ is not defined, then there must exist some marking marking $\bar{\mu}_i$ for which $|\hat{\mu}_0 \rightsquigarrow \bar{\mu}_i| > 1$ and $\Pr[\bar{\mu}_i \rightsquigarrow \hat{\mu}]$ is defined, while for some $\bar{\mu}_j \in \mathrm{SM}(\bar{\mu}_i)$, $\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}]$ is not defined. (If no such node $\bar{\mu}_i$ exists, then the contradictory assumption cannot hold.)

There are two possibilities: $\bar{\mu}_i$ is sequenced or unsequenced. If $\bar{\mu}_i$ is unsequenced, then for some $g \in \mathrm{PG}(\bar{\mu}_i)$, $\Pr[\bar{\mu}_i \rightsquigarrow \hat{\mu}|g] = \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}]$ which is undefined. If $\bar{\mu}_i$ is unsequenced, then the only way that $\Pr[\bar{\mu}_i \rightsquigarrow \hat{\mu}]$ can be defined is if $\Pr[\bar{\mu}_i \rightarrow \bar{\mu}_j] = 0$, which violates the condition of the theorem. Thus, $\bar{\mu}_i$ can not be unsequenced.

Therefore, μ_i must be a sequenced node, and we can write

$$\Pr[\mu_i \rightsquigarrow \hat{\mu}] \;\; = \sum_{\bar{\mu}_j \in \mathrm{SM}(\bar{\mu}_i)} \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}] p_{ij} \,, \tag{1}$$

where $p_{ij} = \Pr[\mu_i \rightarrow \bar{\mu}_j]$. Recall that for contradiction, we assume that for some $\bar{\mu}_j \in \mathrm{SM}(\mu_i)$, $\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}]$ is undefined.

To proceed, let us refine the set of states that $\bar{\mu}_j$ could be. Let $\Sigma^{\hat{\mu}_0} = \{\mu \in \Sigma : \hat{\mu}_0 \rightsquigarrow \mu\}$. Thus, $\Sigma^{\hat{\mu}_0}$ gives us the subset of states on which we can focus our attention. Furthermore, let us define a partitioning of the schedulers that are unique to the scheduling of partitions in $\Sigma^{\hat{\mu}_0}$. Let $\{\varsigma_i\}$ be a partitioning of S_M such that $\varsigma = \{s \in S_M : s(\mu)\text{is identical} \forall \mu \in \Sigma^{\hat{\mu}_0}\}$. Thus, $\forall s \in \varsigma_i$, the model behaves identically in its evolution from $\hat{\mu}_0$ until the next stable marking. For convenience, we can write $\varsigma(\mu)$ to mean $s(\mu)$ for some $s \in \varsigma$, since the value is identical for all $s \in \varsigma$. In particular, we write $\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma]$ for $\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|s]$, for $s \in \varsigma$.

Note that $\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_i]$ must be different for at least two partitions of the scheduler, because if the probability was identical for all schedulers, then the probability would be defined. Furthermore, we can deduce that there must exist partitions of the scheduler ς_a and ς_b such that

$\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_a] \neq \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_b]$ and $|\{\mu \in \Sigma^{\hat{\mu}_0} : \varsigma_a(\mu) \neq \varsigma_b(\mu)\}| = 1$. To see this, consider the a finite, orthogonal, discrete space defined by $|\Sigma^{\hat{\mu}_0}|$ dimensions, where each $\mu \in \Sigma^{\hat{\mu}_0}$ defines an axis (called $\vec{\mu}$). Note that $||\vec{\mu_i}|| = |\mathrm{PG}(\mu_i)|$. One can trace a path from a point in the subspace defined by ς_a to a point in an adjacent subspace defined by ς_b that moves in directions parallel an axis. At some way along the path, one point lies in ς_a, and the next point along the path, moving only in one dimension, lies outside of ς_a. Let that dimensioned be labeled μ_d. I.e., $\varsigma_a(\mu_d) \neq \varsigma_b(\mu_d)$, and $\varsigma_a(\mu) = \varsigma_b(\mu)$, $\forall \bar{\mu} \in \Sigma^{\hat{\mu}_0} \setminus \{\mu_d\}$.

Recall that ς_a and ς_b were chosen so that $\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_a] \neq \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_b]$, where $\Pr[\mu_i \rightsquigarrow \hat{\mu}]$ is defined, but for some $\bar{\mu}_j \in \mathrm{SM}(\mu_i)$, $\Pr[\mu_j \rightsquigarrow \hat{\mu}]$ is undefined. We can partition the set of paths $\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_a$ into paths that pass through marking μ_d and those that don't. Thus,

$$\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_a] \;=\; \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_a] + \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \not\rightsquigarrow \mu_d|\varsigma_a]$$

Also, note the identity

$$\begin{aligned}\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_a] &= \Pr[\bar{\mu}_j \rightsquigarrow \mu_d \cap \mu_d \rightsquigarrow \hat{\mu}|\varsigma_a] \\ &= \Pr[\bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_a]\Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_a]\,.\end{aligned}$$

(The latter equality follows directly from the execution policy.) Since $\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \not\rightsquigarrow \mu_d|\varsigma_a] = \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \not\rightsquigarrow \mu_d|\varsigma_b]$, we can deduce that

$$\Pr[\bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_a]\Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_a] \;\neq\; \Pr[\bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_b]\Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_b] \tag{2}$$

There are two important facts to note about (2). First,

$$\Pr[\bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_a] \;=\; \Pr[\bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_b] > 0\,. \tag{3}$$

The first equality can be easily seen because the stochastic process behaves identically under schedulers ς_a and ς_b on paths from $\bar{\mu}_j$ to μ_d; only at μ_d does it behave differently. The probability of the stochastic process evolving from $\bar{\mu}_j$ to μ_d is also non-zero; if $\Pr[\bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_a] = 0$, then we can deduce that $\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_a] = \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_b]$. This is a contradiction of the way we picked μ_d, ς_a, and ς_b. Secondly, given (2) and (3), we know

$$\Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_a] \;\neq\; \Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_b]\,. \tag{4}$$

This makes intuitive sense; since $\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_a] \neq \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_b]$, and since ς_a and ς_b differ only in its behavior at μ_d, then $\Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_a] \neq \Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_b]$. We will use this result later.

Going back to (1), we can write

$$\Pr[\bar{\mu}_i \rightsquigarrow \hat{\mu}] = \sum_{\bar{\mu}_j \in \mathrm{SM}(\bar{\mu}_i)} \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}] p_{ij} \,,$$

where some terms on the right hand side are undefined, as two separate equations:

$$\Pr[\bar{\mu}_i \rightsquigarrow \hat{\mu}] = \sum_{\bar{\mu}_j \in \mathrm{SM}(\bar{\mu}_i)} \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_a] p_{ij} \,, \tag{5}$$

$$\Pr[\bar{\mu}_i \rightsquigarrow \hat{\mu}] = \sum_{\bar{\mu}_j \in \mathrm{SM}(\bar{\mu}_i)} \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_b] p_{ij} \,. \tag{6}$$

Now all the probabilities are defined. Taking the difference between (5) and (6), we get

$$0 = \sum_{\bar{\mu}_j \in \mathrm{SM}(\bar{\mu}_i)} (\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_a] - \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_b]) p_{ij} \,. \tag{7}$$

Again, we can partition the set of events $\bar{\mu}_j \rightsquigarrow \hat{\mu}$ into two disjoint sets that include and exclude μ_d: $\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \rightsquigarrow \mu_d$ and $\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \not\rightsquigarrow \mu_d$. Thus, we can write

$$\begin{aligned}
\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_a] &= \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_a] + \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \not\rightsquigarrow \mu_d|\varsigma_b] \,, \\
\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu}|\varsigma_b] &= \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_b] + \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \not\rightsquigarrow \mu_d|\varsigma_b] \,.
\end{aligned}$$

Since

$$\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \not\rightsquigarrow \mu_d|\varsigma_a] = \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \not\rightsquigarrow \mu_d|\varsigma_b] \,,$$

we can rewrite (7) as

$$0 = \sum_{\bar{\mu}_j \in \mathrm{SM}(\bar{\mu}_i)} (\Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_a] - \Pr[\bar{\mu}_j \rightsquigarrow \hat{\mu} \cap \bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_b]) p_{ij}$$

Substituting as before, we get

$$\begin{aligned}
0 &= \sum_{\bar{\mu}_j \in \mathrm{SM}(\bar{\mu}_i)} (\Pr[\bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_a] \Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_a] - \Pr[\bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_b] \Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_b]) p_{ij} \\
&= \left(\Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_a] \sum_{\bar{\mu}_j \in \mathrm{SM}(\bar{\mu}_i)} \Pr[\bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_a] p_{ij} \right) - \left(\Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_b] \sum_{\bar{\mu}_j \in \mathrm{SM}(\bar{\mu}_i)} \Pr[\bar{\mu}_j \rightsquigarrow \mu_d|\varsigma_d] p_{ij} \right)
\end{aligned}$$

Since $p_{ij} > 0$, and we know for all $\bar{\mu}_j \in \mathrm{SM}(\mu_i)$, the equality of (3) holds, and the for some $\bar{\mu}_j \in \mathrm{SM}(\mu_i)$, the inequality of (3) holds, we can simply write

$$\Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_a] = \Pr[\mu_d \rightsquigarrow \hat{\mu}|\varsigma_b]\,.$$

This contradicts (4).

■

This proof shows that under the condition $\Pr[\mu_i \rightarrow \mu_j] > 0$, the first part of the well-specified and well-defined checks are identical. A proof of nearly identical form can be used to show that under the same condition, the second part of the well-specified and well-defined checks are identical. Hence, under the condition, well-specified and well-defined are equivalent on IFR models.

3.5 Well-defined algorithm

In this section, we present the well-defined algorithm adapted to IFR models. This is essentially a recasting of the algorithm presented in [21] to apply to IFR models, and to use our notation.

For this algorithm to work, then the following assumptions must hold for all $\hat{\mu}_0 \in \hat{\Sigma}$:

1. $|\Sigma^{\hat{\mu}_0}|$ is finite.

2. The graph is acyclic.

The first requirement is a "hard" requirement for any algorithm-based approach that must examine every state in $\Sigma^{\hat{\mu}_0}$ and terminate. The second requirement is necessary for the algorithm we present, but not strictly necessary for any algorithm. This could be relaxed so that no probability-1 cycles exist, and then a more sophisticated algorithm similar to solving for the steady-state behavior of a DTMC (see [23], for example) could be used. We chose to present the simpler algorithm for clarity, because cycles are rare in models, and because we believe the application is straightforward but contains issues not immediately pertinent.

For some marking $\hat{\mu}_0$, let $Im(\hat{\mu}_0) = \{\imath : \Pr[\hat{\mu}_0 \rightsquigarrow \hat{\mu}, \imath] > 0, \hat{\mu} \in \mathrm{NM}(\hat{\mu}_0)\}$, the set of measurable impulse rewards obtained in a stable step from marking $\hat{\mu}_0$. We use a vector $\vec{P} : \mathbb{R}^{|\Sigma^{\hat{\mu}_0}| \times |Im(\hat{\mu}_0)|}$ that is indexed by markings and impulse rewards, i.e., $\vec{P}(\mu, \imath) \in \mathbb{R}$. Vector

addition and vector-scalar multiplication works in the normal way. However, we use a somewhat unorthodox notation in the equation $\vec{P}_k(\mu_i, \circ) = \vec{P}_k(\mu_i, \circ) + \vec{P}'(\mu_j, \circ - i)p_{ij}$. This is interpret simply as $\vec{P}_k(\mu_i, \imath) = \vec{P}_k(\mu_i, \imath) + \vec{P}'(\mu_j, \imath - i)p_{ij}$, $\forall \imath \in Im$. Figure 2 gives the algorithm. Essentially, it performs a depth-first search of the nodes in $\Sigma^{\hat{\mu}_0}$, and if at any time it finds that $\Pr[\mu_i \rightsquigarrow \hat{\mu}, \imath|\varsigma]$ is not identical for all ς, it declares the model is not well-defined. This algorithm can be applied to all stable markings, but need only be applied to stable markings for which some successor markings are unstable.

3.6 Well-specified algorithm

A novel result of above proof is that we can develop a linear algorithm that will definitively perform the well-defined check with without the condition that $\Pr[\mu_i \rightarrow \mu_j] > 0$. Otherwise, the algorithm does have the same require as the well-defined algorithm. Specifically, for the algorithm to be correct, then the following assumptions must hold for all $\hat{\mu}_0 \in \hat{\Sigma}$:

1. $|\Sigma^{\hat{\mu}_0}|$ is finite.

2. The graph is acyclic.

Again, the first restriction is a "hard" restriction for any algorithm that enumerates $\Sigma^{\hat{\mu}_0}$. The second restriction could be relaxed, so long as any cycles do not cycle with probability 1, and a more sophisticated algorithm applied.

The well-specified algorithm uses values in $\mathbb{R}^{\geq} \cup \{\eta\}$, where η is considered "undefined" or "not a number." For example, if a probability is undefined, we can represent the probability as η. We represent addition and multiplication using operators $\oplus$ and $\odot$. Normal addition and multiplication holds over the reals. Adding and subtracting η yields a η, i.e., $3 \oplus \eta = \eta$. Multiplication by η follows the following rule.

$$x \odot y = \begin{cases} 0 & x = 0 \vee y = 0, \\ xy & x > 0 \wedge y > 0, \\ \eta & \text{otherwise.} \end{cases}$$

The key here is that $0 \odot \eta = 0$, and $x \odot \eta = \eta$, for $x > 0$.. This captures the essential difference between well-specified and well-defined: ambiguity (undefined probabilities) are acceptable for

$\vec{P}$ = well-defined-check($\mu_i : \Sigma, \text{root} : \textit{bool}$)

if ($\mu_i \in \hat{\Sigma} \cap \text{root} = \textit{false}$)

Let $\vec{P} = \vec{0}$

$\vec{P}(\mu_i, 0) := 1$

return $\vec{P}$

$\forall g_k \in \text{PG}(\mu_i)$

Let $\vec{P}_k = \vec{0}$

$\forall \text{t} \in g_k \cap EN^*_{\mu_i}$

Let $\mu_j = \phi(\mu_i, \text{t})$

Let $p_{ij} = \frac{\text{W}(\mu_i, t)}{\sum_{t_j \in g_k \cap EN^I_{\mu_i}} \text{W}(\mu_i, t_j)}$

Let $i = Im(\mu_i, \text{t})$

Let $\vec{P}' = \vec{0}$

$\vec{P}'$ = well-defined-check($\mu_j, \textit{false}$)

$\vec{P}_k(\mu_i, \circ) = \vec{P}_k(\mu_i, \circ) + \vec{P}'(\mu_j, \circ - i)p_{ij}$

End $\forall t$

End $\forall g_k$

If $|\{\vec{P}_k : g_k \in \text{PG}(\mu_i)\}| \neq 1$

Failed well-defined check

Else

return $\vec{P}_1$

End algorithm

Figure 2: Well-defined algorithm for IFR models.

well-defined models if the probability of ambiguity is zero.

We can extend the operator $\odot$ to vectors. Vector addition $\vec{z} = \vec{x} \oplus \vec{y}$ is defined element-wise as $\vec{z}_i = \vec{x}_i \oplus \vec{y}_i$. Scalar multiplication is defined similarly: $\vec{z} = \vec{x}y$ is defined element-wise as $\vec{z}_i = \vec{x}_i \odot y$. Normal precedence rules apply. The algorithm is given in Figure 3.

Theorem 2. *The well-specified algorithm performs the well-specified check.*

If one believes that $\vec{P}(\hat{\mu}, \imath)$ correctly computes $\Pr[\hat{\mu}_0 \rightsquigarrow \hat{\mu}, \imath]$ if it is defined, or η if it is not defined, then the proof is straightforward. The proof that $\vec{P}(\hat{\mu}, \imath)$ correctly computes $\Pr[\hat{\mu}_0 \rightsquigarrow \hat{\mu}, \imath]$ can be derived from first principles.

well-specified-check($\mu_i : \Sigma) : \vec{P}$

Declare $\vec{P}$ = well-specified-recurse(μ_i, true)

If $\eta \in \vec{P}$

    Failed well-specified check

Else

    return $\vec{P}$

well-specified-recurse($\mu_i : \Sigma, \text{root} : \mathit{bool}) : \vec{P}$

if ($\mu_i \in \hat{\Sigma} \cap \text{root} = \mathit{false}$)

    Let $\vec{P} = \vec{0}$

    $\vec{P}(\mu_i, 0) := 1$

    return $\vec{P}$

$\forall g_k \in \text{PG}(\mu_i)$

    Let $\vec{P}_k = \vec{0}$

    $\forall \text{t} \in g_k \cap EN^*_{\mu_i}$

        Let $\mu_j = \phi(\mu_i, \text{t})$

        Let $p_{ij} = \frac{\text{W}(\mu_i, t)}{\sum_{t_j \in g_k \cap EN^*_{\mu_i}} \text{W}(\mu_i, t_j)}$

        Let $i = Im(\mu_i, \text{t})$

        Let $\vec{P}'$ = well-specified-recurse(μ_j, false)

        $\vec{P}_k(\mu_i, \circ) = \vec{P}_k(\mu_i, \circ) \oplus \vec{P}'(\mu_j, \circ - i) \odot p_{ij}$

    End $\forall t$

End $\forall g_k$

return $\vec{P}_1$

Figure 3: Well-specified algorithm for IFR models.

4 Application to GSMPs

A GSMP is a formalism used to describe a class of simulation languages. There have been a number of attempts to formally construct a GSMP formalism, and the results are several very closely related but slightly different constructs [8, 7].

GSMPs use different terminology, and to remain consistent, we take some liberty and continue use SPN terminology when applied to GSMPs. For those familiar with GSMP terminology, we use "transition" instead of "event," "enabled" instead of "active," and a "transition fires" instead of an "event occurs."

Formally, a GSMP (using our terminology) is a tuple:

- Σ : countable set of markings,
- T : finite set of transitions,
- En : $\Sigma \times$ T : enabled relation,
- $p : \Sigma \times 2^T \times \Sigma \to [0, 1]$: given marking σ and set of transitions T^* with the same minimum completion time, $p(\mu, T^*, \mu')$ is the probability that the resulting marking is μ'.
- $r : \Sigma \times \mathrm{T} \to [0, \infty)$; for $r(\mu, t)$, the rate that the clock of t runs in marking μ. This is irrelevant to our discussion so will be dropped.
- $F : \Sigma \times \mathrm{T} \times \Sigma \times \mathrm{T} \to (\mathbb{R} \to [0, 1])$; for $F(\mu, \mathrm{t}, \mu', \mathrm{t}')$ gives the probability distribution function of t′ when t fires in μ resulting in μ'.
- $\mu_0 : \Sigma \to [0, 1]$; an initial probability distribution over the markings. Irrelevant for our use.

There are a number of things that are unsuitable for using this for well-specified and well-defined. First, and most importantly, it leaves no room for any ambiguity in the model. If multiple transitions have the same firing time, then the function p resolves any ambiguity by simply defining the behavior of what will happen when T^* have the same minimum completion time. This is convenient for resolving any ambiguity, but can be very tedious and impractical to define, a priori, all possible conflicts. It suffers from all the drawbacks of over-specification

that we argued in Section 1, where we argued that ambiguity is sometimes an important and useful behavior to capture in models. Further, synchronization on immediate transitions can be used to construct larger models in a modular way, so the way the way ambiguity is resolved may depend on how the model is used in context with other models, so a priori definitions are impractical.

To extend the well-specified and well-defined definitions to GSMPs, we took the liberty of modifying and simplifying the definition of GSMP. Our goal was to modify it in such a way that it still retains the essential expressive nature, eliminates inessential features for our analysis, and to introduce a mechanism for specifying ambiguity. First, we removed the function p and replaced it with more GSPN-like constructs: ϕ, W, and PG. This gives us the ability to specify ambiguous behavior. Second, we eliminated r as it is inessential to our analysis. The result is what we call a "modified" GSMP, or MGSMP.

Definition 14. *A* modified GSMP*, or* MGSMP*, is a tuple* $(\Sigma, \mathrm{T}, \mathrm{En}, \phi, F, \mathrm{W}, \mathrm{PG}, \mu_0)$*:*

- Σ *is a countable set of markings.*
- T *: a set of (generally timed) transitions.*
- $\mathrm{En} : \Sigma \times \mathrm{T}$ *; the enabled relation.*
- $\phi : \Sigma \times \mathrm{T} \to \Sigma \cup \emptyset$*, the transition firing function.*
- $F : \Sigma \times \mathrm{T} \to (\mathbb{R} \to [0, 1])$*; for* $F(\mu, \mathrm{t})$ *gives the probability distribution function of* t *when the model is marking* μ*.*
- $\mathrm{W} : \mu \times \mathrm{T} \to \mathbb{R}^{\geq}$*, the weight function. If* $\mathrm{W}(\mu, t) > 0$*, then* t *is said to be enabled in* μ*; otherwise it is not enabled.*
- $\mathrm{PG} : \Sigma \to 2^{2^{\mathrm{T}}}$*, a partitioning of the transitions into partition groups.*
- $\mu_0 \in \Sigma$*, the initial marking of the model.*

The *state* of a GSPN is the marking of the GSPN, but since delays (given by F) are not exponential, the state of a GSMP is the marking along with the "clock" values of the GSMP. The clock value is used to measure the residual time to firing of a transition. The state of

a GSPN is given by $\sigma = (\mu_i, \vec{c}_i)$, where $\vec{c}_i$ is a vector of non-negative real values indexed by transitions. Thus, c_i^{t} is the value of the clock for transition t in state σ_i. The state space of a GSMP is then $\Xi = \{\sigma\}$, and Ξ is in general uncountable.

One important difference between MGSMPs and GSPNs is that MGSMPs allow general firing time distributions, which may include mixed or discrete distributions. Thus, it is possible that multiple timed transitions are scheduled to fire simultaneously. Consequently, immediate and timed transitions may be in competition, and partition groups must be defined over all transitions, not just immediate transitions.

While a GSMP defines a unique stochastic process, the partition groups in MGSMPs allow for some ambiguity and the need for a scheduler to define a stochastic process. Let a scheduler $s : \Sigma \to 2^{2^{\mathrm{T}}}$, where $s(\mu) \in \mathrm{PG}(\mu)$, and indicates which partition group is selected in marking μ so that transitions of that partition group will fire before transitions of other partition groups.

For a given MGSMP model M, let S_M be the set of schedulers possible in the model. A MGSMP model M and a scheduler $s \in S_M$ defines a stochastic process $\mathrm{SP}_s : \Omega \times \mathbb{N} \to \mathrm{T} \times \theta \times \Xi$. We define an *event* of a MGSMP as $\varepsilon_i = (\mathrm{t}_i, \theta_i, \sigma_i)$, and the set of all events is given as $\mathcal{E} : \mathrm{T} \times \Theta \times \Xi$.

The execution policy for a MGSMP is more complex than a GSMP because of clock values. Let us define for some state $\sigma_i = (\mu_i, \vec{c}_i)$, let $EN^*_{\mu_i} = \{\mathrm{t} : \mathrm{t} \in EN_{\mu_i}, c_i^{\mathrm{t}} \in \vec{c}_i\}$, the set of enabled transitions competing to fire next. The new execution policy is given in Figure 4.

It is possible to take the well-specified and well-defined conditions and apply them directly to MGSMP models on states instead of markings. Validating a MGSMP model would be difficult, however, because the state space of a MGSMP in general is uncountable. This is not insurmountable because, in general, only sample paths of the stochastic process are generated, and the checks can be applied on sample paths (which visit a finite number of states). However, applying the check in this way has two deficiencies. First, it slows down the generation of sample paths, and secondly it only validates that no disallowed ambiguities were encountered by a sample path, not that disallowed ambiguities do not exist.

Fortunately, there is a better way to deal with the well-specified and well-defined conditions. Since the number of transitions is finite, the number of combinations of transitions competing in a marking is finite. The actual value of all the clock values are irrelevant; the only relevant

$EP(\sigma_i, \theta_i, s) \mapsto (\mathrm{t}_{i+1}, \theta_{i+1}, \sigma_{i+1})$

Let $\mathrm{T}^* = \{\mathrm{t} : c^{\mathrm{t}} \in \min \vec{c}_i\}$.

Let $g_i = s(\mu_i)$

Let $\mathrm{t}_{i+1} \in EN^*_{\mu_{i+1}}$ be selected with probability $\frac{\mathrm{W}(\mu_i, \mathrm{t}_{i+1})}{\sum_{\mathrm{t}_j \in g_i \cap EN^*_{\mu_i}} \mathrm{W}(\mu_i, \mathrm{t}_j)}$

$\mu_{i+1} = \phi(\mathrm{t}, \mu_i)$

$\theta_{i+1} = \theta_i + c^{\mathrm{t}}_i$

If $\mathrm{t} \in EN_{\mu_{i+1}}$

$\quad c^{\mathrm{t}}_{i+1} = \mathbf{X} : F_{\mathbf{X}}(t) = F(\mu_{i+1}, \mathrm{t}_{i+1})$

Else

$\quad c^{\mathrm{t}}_{i+1} = 0$

$\forall \mathrm{t}_j \in \mathrm{T} \setminus \{\mathrm{t}\}$

$$c^{\mathrm{t}_j}_{i+1} = \begin{cases} \mathbf{X} : F_{\mathbf{X}}(t) = F(\mu_{i+1}, \mathrm{t}_j) & \mathrm{t}_j \notin EN_{\mu_i} \wedge \mathrm{t}_j \in EN_{\mu_{i+1}} \\ c^{\mathrm{t}_j}_i - c^{\mathrm{t}}_i & \mathrm{t}_j \in EN_{\mu_i} \wedge \mathrm{t}_j \in EN_{\mu_{i+1}} \\ 0 & \mathrm{t}_j \notin EN_{\mu_i} \wedge \mathrm{t}_j \notin EN_{\mu_{i+1}} \\ 0 & \mathrm{t}_j \in EN_{\mu_i} \wedge \mathrm{t}_j \notin EN_{\mu_{i+1}} \end{cases}$$

End $\forall$

Figure 4: Conflict-free execution policy for MGSMP.

factor is the set of transitions with the same minimum clock value, and the combination of those is finite. Therefore, a well-specified or well-defined check could, in theory, be performed on each marking and every subset of transition enabled in the marking.

Many of the definitions no longer apply to markings, but to states.

Definition 15. *We can define a pair of* goes to *(→) relations. Let* $\rightarrow: \Xi \times \Xi$ *be defined such that* $(\sigma_i, \sigma_{i+1}) \in \rightarrow$ *if* $\exists \omega \in \Omega$, $n \in \mathbb{N}$, *and* $s \in S_M$ *such that* $\sigma_i \in \mathrm{SP}_s(\omega, n)$ *and* $\sigma_{i+1} \in \mathrm{SP}_s(\omega, n+1)$. *Let* $\xrightarrow{s}: \Xi \times \Xi$ *be defined such that* $(\sigma_i, \sigma_{i+1}) \in \xrightarrow{s}$ *if* $\exists \omega \in \Omega$ *and* $n \in \mathbb{N}$ *such that* $\sigma_i \in \mathrm{SP}_s(\omega, n)$ *and* $\sigma_{i+1} \in \mathrm{SP}_s(\omega, n+1)$. *For convenience, we write* $\sigma_i \rightarrow \sigma_{i+1}$.

Definition 16. *A state* σ_i *is called a* stable state *if* $\forall \mathrm{t} \in EN_{\mu_i}$, $c_i^{\mathrm{t}} > 0$, *i.e., if the next transition firing happens in non-zero time. We write* $\hat{\sigma}_i$ *to mean that the state* σ_i *is a stable state. We denote* $\hat{\Xi} \subseteq \Xi$ *to be the subset of all states that are stable states.*

Definition 17. *A* leads to *relation, written* $\rightsquigarrow: \Xi \times \Xi$ *is defined such that* $(\sigma_i, \sigma_j) \in \rightsquigarrow$ *if* $\exists \omega \in \Omega$, $n, k \in \mathbb{N}$, *and* $s \in S_M$ *such that for* $\varepsilon_n = \mathrm{SP}(\omega, n)$, $\sigma_i \in \varepsilon_n$, $\sigma_j \in \varepsilon_{n+k}$, *and if* $0 = \min \vec{c}_n$, *then* $\theta_n = \theta_{n+k}$, *else* $\theta_{n+1} = \theta_{n+k}$. *For convenience, we write* $\sigma_i \rightsquigarrow \sigma_j$.

Definition 18. *Let the state space of a model* M *with initial marking* μ_0 *and corresponding initial state* σ_0 *be defined as the transitive closure of* $\rightsquigarrow$ *with* σ_0. *I.e., if* $\rightarrow^*$ *is the transitive closure of* $\rightarrow$, *then* $\Xi = \{\sigma : \sigma_0 \rightarrow *\sigma\}$. *Note that in general,* Ξ *is uncountable.*

Definition 19. *A state* σ_i *is called an* unstable state *if* $\exists t \in EN_{\mu_i}$ *such that* $c_i^t = 0$, *i.e., the next transition fires in zero time. We write* $\bar{\sigma}_i$ *to mean that state* σ_i *is an unstable marking. We denote* $\bar{\Xi}$ *to be the subset of all states that are unstable states.*

Definition 20. *A state* σ_i *is called a* concurrent state *if for* $\sigma_i = (\mu_i, \vec{c}_i)$, $|\{t : c_i^t \in \min \vec{c}_i\}| > 1$, *i.e., transitions competing for minimum firing time.*

Concurrent states play the same role as unstable markings in GSPNs.

Definition 21. *A MGSMP model* M *is* well-defined *if* $\forall \bar{\sigma}_i \in \bar{\Xi}$, $\forall \hat{\sigma} \in \mathrm{NS}(\bar{\sigma}_i)$, *and* $\forall \imath \in Im$,

1. $\Pr[\bar{\sigma}_i \rightsquigarrow \hat{\sigma}]$ *is defined, and*

2. $\Pr[\bar{\sigma}_i, \rightsquigarrow \hat{\sigma}, \imath]$ *is defined.*

Definition 22. *A MGSMP model M is* well-specified *if* $\forall \hat{\sigma}_0 \in \hat{\Xi}$, $\forall \hat{\sigma} \in \mathrm{NS}(\tilde{\sigma}_0)$, *and* $\forall \imath \in Im$,

1. $\Pr[\hat{\sigma}_0 \rightsquigarrow \hat{\sigma}]$ *is defined, and*

2. $\Pr[\hat{\sigma}_0 \rightsquigarrow \hat{\sigma}, \imath]$ *is defined.*

Similar to the way we defined $\Sigma^{\hat{\mu}_0}$, we can define $\Xi^{\hat{\sigma}_0} = \{\sigma \in \Xi : \hat{\sigma}_0 \rightsquigarrow \sigma\}$. Thus, $\Sigma^{\hat{\mu}_0}$ gives us the subset of states which are reachable from $\hat{\mu}_0$ by firing at most one timed transition followed by any number of transitions that fire in zero time.

The well-specified and well-defined algorithms are identical except that σ is used in place of μ. See Figures 2 and 3. However, one problem with using the algorithms as presented is that $|\Xi|$ is in general uncountable, even if $|\Sigma|$ is finite. Thus, determining whether a MGSMP model M is well-defined or well-specified may be undecidable. In one sense, this may be acceptable. Whatever analysis of the MGSMP model that involves in some way "executing" the model (e.g., generating sample paths, used in Monte Carlo simulation), the well-specified or well-defined checker can be used. As a result, one can say that in the course of analysis, no state was encountered in which the model behaved in a sufficiently ambiguous way.

This is also an unsatisfactory approach, especially when some states are unlikely to be reached. It could be that the model is not well-specified or well-defined and hiding a serious flaw, but since the state that leads to ambiguity is never reached, the flaw is never found. Furthermore, if $|\Sigma|$ is finite, surely there must exist some check that will determine whether the model is well-specified or well-defined. Finally, even though $|\Sigma|$ may be countably infinite, often the model can be modified to make $|\Sigma|$ finite and exercise all relevant behavior. For example, an open queueing network may simply bound the number of customers in the queue. If the model is correct for finite customers, it is unlikely that it becomes incorrect with infinite customers.

In the next section, we explore such an approach. Unfortunately, we are not aware of an exact means of determining well-specified or well-defined, but a reasonable sufficient condition does exist.

4.1 New MGSMP checks

We propose a new pair of checks that can be applied as sufficient conditions for a MGSMP to be well-defined or well-specified. These checks we call *sufficient well-defined check* and *sufficient well-specified check*, and a model that passes these checks are *sufficient-well-defined* and *sufficient-well-specified* respectively.

Definition 23. *Let* T^D *be the set of transitions* t *such that for any markings* $\mu \in \Sigma$, $F(\mu, t)$ *is a discrete or mixed distribution function. I.e.,* T^D *is the set of all transitions that might have a discrete or mixed distribution function.*

We use T^D to restrict the number of transitions that have the same completion times. In general, $\mathrm{T}^D = \mathrm{T}$. However, the by limiting T^D to the smallest set of transitions that may share the same completion time, we can make the algorithm more efficient, and eliminate conditions that are clearly impossible.

The set of transitions $\mathrm{T}^D \cap EN_{\hat{\mu}_i}$ are the set of transitions that we can conservatively assume may be enabled and tied for the earliest completion time in stable marking $\hat{\mu}$. Let $\hat{\mu} \in \mathrm{NM}(\hat{\mu}_i)$, and let $\hat{\mu}_i \rightsquigarrow \hat{\mu}|\mathrm{T}^D \cap EN_{\hat{\mu}_i}$ be the set of event sequences that, given the model is in marking $\hat{\mu}_i$, the model evolves from $\hat{\mu}_i$ to next stable marking $\hat{\mu}$, assuming that any transition in $\mathrm{T}^D \cap EN_{\hat{\mu}_i}$ could fire first. Let $\mathrm{T}' \subset \mathrm{T}^D \cap EN_{\hat{\mu}_i}$, and thus $\hat{\mu}_i \rightsquigarrow \hat{\mu}|\mathrm{T}' \subseteq \hat{\mu}_i \rightsquigarrow \hat{\mu}|\mathrm{T}^D \cap EN_{\hat{\mu}_i}$. If we know that $\Pr[\hat{\mu}_i \rightsquigarrow \hat{\mu}|\mathrm{T}^D \cap EN_{\hat{\mu}_i}]$, then certainly $\Pr[\hat{\mu}_i \rightsquigarrow \hat{\mu}|\mathrm{T}']$ will be defined.

The essential strategy behind the well-specified check is to simply determine whether $\Pr[\hat{\mu}_i \rightsquigarrow \hat{\mu}|\mathrm{T}^D \cap EN_{\hat{\mu}_i}]$ is defined. Note that this is a sufficient condition, but not necessary. There could be some transition $t \in \mathrm{T}^D \cap EN_{\hat{\mu}_i}$ that, due to the nature of the model, could never compete to fire first in marking $\hat{\mu}_i$. Without actually executing the model, there is no way we can know this. Consequently, we assume the worst case. Similarly, the well-defined check checks that for all states $\{\mu : \hat{\mu}_i \rightsquigarrow \mu\}$, $\Pr[\mu \rightsquigarrow \hat{\mu}|\mathrm{T}^D \cap EN_{\hat{\mu}_i}]$ is defined. Again, all the states that are checked may not be in fact reachable, but without executing the model, there is no way of knowing this for sure (in general). Thus, we make the conservative assumption.

An attempt at creating an algorithm that performs these sufficient checks for well-defined-2 and well-specified are given in Figures 5 and 6.

Note that the sufficient well-specified and sufficient well-defined checks can be used in

well-defined-check($\hat{\mu}_0 : \hat{\Sigma}$)

well-defined-recurse($\hat{\mu}_0$, $\mathrm{T}^D \cap EN_{\mu_0}$, *true*)

well-defined-recurse($\mu_i : \Sigma$, $T_i^c : 2^T$, root : *bool*) : $\vec{P}$

If $\mu_i \in \hat{\Sigma} \cap$ root = *false*

Let $\vec{P} = \vec{0}$

$\vec{P}(\mu_i, 0) := 1$

return $\vec{P}$

Else

$\forall g_k \in \mathrm{PG}(\mu_i)$

Let $\vec{P}_k = \vec{0}$

$\forall \mathrm{t} \in g_k \cap \mathrm{T}_i^c$

Let $\mu_j = \phi(\mu_i, \mathrm{t})$

Let $p_{ij} = \frac{\mathrm{W}(\mu_i,t)F(\mu_i,t)(0)}{\sum_{t_j \in g_k \cap \mathrm{T}_i^c} \mathrm{W}(\mu_i,t_j)F(\mu_i,t_j)(0)}$

Let $i = Im(\mu_i, \mathrm{t})$

Let $\mathrm{T}_k^c = (\mathrm{T}_i^c \cup \mathrm{T}_{\mu_j}^I) \cap EN_{\mu_j}$

Let $\vec{P}' =$ well-defined-recurse(μ_j, T_k^c, *false*)

$\vec{P}_k(\mu_i, \circ) = \vec{P}_k(\mu_i, \circ) + \vec{P}'(\mu_j, \circ - i)p_{ij}$

End $\forall t$

End $\forall g_k$

If $|\{\vec{P}_k : g_k \in \mathrm{PG}(\mu_i)\}| > 1$

Failed well-defined check

Else

return $\vec{P}_1$

End if

End if

Figure 5: Sufficient well-defined check for MGSMP models.

well-specified-check($\hat{\mu}_i : \hat{\Sigma}$)

Let $\vec{P}$ = well-specified-recurse(μ_i, $\mathrm{T}^D \cap EN_{\mu_i}$, *true*)

If $\eta \in \vec{P}$

Failed well-specified check

End if

well-specified-recurse($\mu_i : \Sigma$, $\mathrm{T}_i^c : 2^{\mathrm{T}}$, root : *bool*) : $\vec{P}$

if $\mu_i \in \hat{\Sigma} \cap$ root = *false*

Let $\vec{P} = \vec{0}$

$\vec{P}(\mu_i, 0) := 1$

return $\vec{P}$

Else

$\forall g_k \in \mathrm{PG}(\mu_i)$

Let $\vec{P}_k = \vec{0}$

$\forall \mathrm{t} \in g_k \cap \mathrm{T}_i^c$

Let $\mu_j = \phi(\mu_i, \mathrm{t})$

Let $p_{ij} = \frac{\mathrm{W}(\mu_i,t)F(\mu_i,t)(0)}{\sum_{t_j \in g_k \cap \mathrm{T}_i^c} \mathrm{W}(\mu_i,t_j)F(\mu_i,t_j)(0)}$

Let $i = Im(\mu_i, \mathrm{t})$

Let $\mathrm{T}_k^c = (\mathrm{T}_i^C \cup \mathrm{T}_{\mu_j}^I) \cap EN_{\mu_j}$

Let $\vec{P}'$ = well-specified-recurse(μ_j, T_k^c, *false*)

$\vec{P}_k(\mu_i, \circ) = \vec{P}_k(\mu_i, \circ) \oplus \vec{P}'(\mu_j, \circ - i) \odot p_{ij}$

End $\forall t$

End $\forall g_k$

return $\vec{P}_1$

End if

Figure 6: Well-specified algorithm for IFR models.

conjunction with the well-specified and well-defined checks as a two-level screening process. A model can be subjected to the sufficient well-specified or sufficient well-defined checks first. If the model passes the checks, then the model is well-specified or well-defined. If the check fails and the modeler believes the model to be correct, then the well-specified or well-defined check can be used while the model is executing to ensure correctness.

5 Conclusion

The theoretical foundations for expressing simple forms of non-determinism in probabilistic models has much to be desired. Competing definitions, checks, and formalisms have made the progress and analysis difficult. Furthermore, assertions without proofs have led to widely-used and incorrect definitions. Two non-structural definitions that are used in the community are *well-specified* and *well-defined*. Unfortunately, lack of rigor and a common formalism have led to misunderstandings and unnecessary complexity.

In this work, we address the issues in a much more rigorous way. We begin by formally describing the formalism and stochastic process, so we can reason formally about it. We then apply the definitions of well-specified and well-defined, and then (formally) show under what conditions they are equivalent.

Furthermore, the widely used class of simulation languages fall under the GSMP formalism, which is carefully constructed so that it does not allow for any non-determinism. As we argue from physics, this is an artificial and unrealistic representation of reality. We modify the definition of GSMP to allow for some forms of ambiguity, and then apply the well-specified and well-defined definitions. It is difficult to check for either condition in GSMPs due to the uncountable number of states, so we propose a sufficient condition that can be used.

There is significant future work to be done on this problem. First, there are a number of "holes" that need to be removed. For example, the proof that the well-specified and well-defined algorithm are correct needs to be performed. While we are confident that it is true, and a proof would be straightforward, it should still be done, since previous experience has shown that confidence is not a good indicator of correctness. Second, an algorithm that accepts cycles should be explored. Currently, the algorithms work (terminate) on cycle-free unstable markings.

The work with modified GSMPs needs to be completed. While a sufficient condition is stated, no proof of its sufficiency is offered. Furthermore, there is a possibility of using more information (if available) to make a tighter sufficient condition. Frequently, transitions in models are either timed or immediate, i.e., the time to completion is always greater than zero or equal to zero. That information could be used to build tighter sufficient conditions.

We should also provide proofs for classes of formalisms that do allow for some non-determinism that the IFR really is an intermediate formalism representation, and thus that the results for IFR will also hold for the higher-level formalisms, such as SANs and GSPNs.

Finally, we should explore broader forms of non-determinism, such as that found in bisimulation.

Acknowledgments

I gratefully acknowledge the helpful input by Holger Hermanns.

References

[1] F. Baskett, K. M. Chandy, R. R. Muntz, and F. G. Palacios, "Open, closed, and mixed networks of queues with different classes of customers," *Journal of the Association for Computing Machinery*, vol. 22, pp. 248–260, Apr. 1975.

[2] C. H. Sauer and E. A. MacNair, *Simulation of Computer Communication Systems*. Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1983.

[3] M. Ajmone Marsan, G. Balbo, and G. Conte, "A class of generalized stochastic Petri nets for the performance evaluation of multiprocessor systems," *IEEE Transactions on Computers*, vol. 2, pp. 93–122, 1984.

[4] M. K. Molloy, "Performance analysis using stochastic Petri nets," *IEEE Transactions on Computers*, vol. 31, pp. 913–917, Sept. 1982.

[5] J. Hillston, *A Compositional Approach to Performance Modelling*. Cambridge: Cambridge University Press, 1996.

[6] M. L. Shooman, *Probabilistic Reliability: An Engineering Approach*. New York: McGraw-Hill, 1968.

[7] C. G. Cassandras, *Discrete Event Systems: Modeling and Performance Analysis*. Homewood, Illinois: Aksen Associates Incorporated Publishers, 1993.

[8] P. W. Glynn, "A GSMP formalism for discrete event systems," *Proceedings of the IEEE*, vol. 77, pp. 14–23, Jan. 1989.

[9] R. German, A. van Moorsel, M. A. Qureshi, and W. H. Sanders, "Algorithms for the generation of state-level representations of stochastic activity networks with general reward structures," *IEEE Transactions on Software Engineering*, vol. 22, pp. 603–614, Sept. 1996.

[10] L. Kant and W. H. Sanders, "Analysis of the distribution of consecutive cell losses in an atm switch using stochastic activity networks," *Special Issue of International Journal of Computer Systems Science & Engineering on ATM Switching*, vol. 12, pp. 117–129, Mar. 1997.

[11] R. Segala and N. Lynch, "Probabilistic simulations for probabilistic processes," in *Proc. 5th Int. Conf. on Concurrency (CONCUR '94)* (B. Jonsson and J. Parrow, eds.), vol. 836 of *LNCS*, Uppsala, Sweeden, pp. 481–496, Springer Verlag, 1994.

[12] P. R. D'Argenio, H. Hermanns, J. P. Katoen, and R. Klaren, "MoDeST - a modelling and description language for stochastic timed systems," in *Process Algebra and Probabilistic Methods* (L. de Alfaro and S. Gilmore, eds.), vol. 2165 of *LNCS*, Aachen, Germany, pp. 87–104, Springer Verlag, Sept. 2001.

[13] G. Chiola, M. Ajmone Marsan, G. Balbo, and G. Conte, "Generalized stochastic Petri nets: A definition at the net level and its implications," *IEEE Trans. on Software Engineering*, vol. 19, pp. 89–107, Feb. 1993.

[14] G. Ciardo, *Analysis of Large Stochastic Petri Net Models*. PhD thesis, Duke University, Department of Computer Science, Durham, North Carolina, 1993.

[15] G. Ciardo, A. Blakemore, P. F. J. Chimento, J. K. Muppala, and K. S. Trivedi, "Automated generation and analysis of Markov reward models using stochastic reward nets," in *Linear Algebra, Markov Chains, and Queueing Models* (C. Meyer and R. J. Plemmons, eds.), pp. 141–191, Heidelberg: Springer-Verlag, 1993.

[16] J. F. Meyer, A. Movaghar, and W. H. Sanders, "Stochastic activity networks: Structure, behavior, and application," in *Proc. Int. Workshop on Timed Petri Nets*, Torino, Italy, pp. 106–115, July 1985.

[17] E. Teruel, G. Franceschinis, and M. D. Pierro, "Clarifying the priority specification of gspn: Detached priorities," in *Proc. 8th Int. Workshop on Petri Nets and Performance Models (PNPM '99)* (P. Buchholz, ed.), Zaragosa, Spain, pp. 114–123, IEEE Comp. Soc. Press, Sept. 1999.

[18] A. Movaghar and J. F. Meyer, "Performability modeling with stochastic activity networks," in *Proc. 1984 Real-Time Systems Symposium*, Austin, Texas, pp. 215–224, Dec. 1984.

[19] W. H. Sanders, *Construction and Solution of Performability Models Based on Stochastic Activity Networks*. PhD thesis, University of Michigan, Ann Arbor, Michigan, 1988.

[20] M. A. Qureshi, W. H. Sanders, A. P. A. van Moorsel, and R. German, "Algorithms for the generation of state-level representations of stochastic activity networks with general reward structures," in *6th Int. Wksp. on Petri Nets and Performance Models (PNPM '95)*, Durham, North Carolina, pp. 180–190, IEEE Comp. Soc. Press, Oct. 1995.

[21] D. D. Deavours and W. H. Sanders, "An efficient well-specified check," in *Proc. 8th Int. Workshop on Petri Nets and Performance Models (PNPM '99)* (P. Bucholz and M. Silva, eds.), Zaragosa, Spain, pp. 124–133, IEEE Comp. Soc. Press, Sept. 1999.

[22] G. Ciardo and R. Zijal, "Well-defined stochastic Petri nets," in *Proc. 4th Int. Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'96)*, San Jose, California, pp. 278–284, Feb. 1996.

[23] V. G. Kulkarni, *Modeling and Analysis of Stochastic Systems*. Chapman & Hall, 1995.