SOFTWARE VERIFICATION RESEARCH CENTRE

THE UNIVERSITY OF QUEENSLAND

Queensland 4072
Australia

TECHNICAL REPORT

No. 02-28

Randomized Non-sequential Processes and Distributed Adversaries

Hagen Völzer

July 30, 2002

Phone: +61 7 3365 1003
Fax: +61 7 3365 1533
http://svrc.it.uq.edu.au

Note: Most SVRC technical reports are available via anonymous ftp, from `svrc.it.uq.edu.au` in the directory `/pub/techreports`. Abstracts and compressed postscript files are available from `http://svrc.it.uq.edu.au`

Randomized Non-sequential Processes and Distributed Adversaries

Hagen Völzer*

Abstract

We suggest a non-sequential, i.e., partial-order, semantics for randomized distributed algorithms. It is based on Petri nets and their branching processes. We introduce *randomized Petri nets* and their semantics, *probabilistic branching processes*. As a main result, we show that each probabilistic branching process defines a unique canonical probability space on the set of its maximal runs. Finally, we show that the non-sequential semantics differs from the classical sequential semantics, modelling a new adversary, called the *distributed adversary*.

*Postal address: Hagen Völzer, SVRC, The University of Queensland, Qld 4072, Australia; e-mail: voelzer@svrc.uq.edu.au; this work was supported by Deutsche Forschungsgemeinschaft: project "Konsensalgorithmen". Large parts of the work were carried out when the author was with the Institut für Informatik of Humboldt-Universität Berlin, Germany. An extended abstract of a preliminary version of this paper has appeared as [31].

Contents

1 Introduction

A *randomized distributed algorithm* is a distributed algorithm where agents may flip coins during the execution of their programs. Randomized algorithms gained more and more attention in the last twenty years since they often solve problems in a simpler and more efficient way than ordinary distributed algorithms. Some randomized algorithms solve problems that are known to be unsolvable by ordinary algorithms. Examples of such problems are symmetry breaking [12, 14], choice coordination [22], and consensus [3]. For a survey of randomized algorithms, see [10].

To model randomized distributed algorithms, a formalism for modelling distributed algorithms has to be equipped with a coin flip construct. Existing formalisms include *probabilistic I/O automata* [27], *probabilistic programs* [21], *probabilistic concurrent processes* [1], a formalism suggested by Rao [23], which extends UNITY [7], *probabilistic predicate transformers* [17], and several probabilistic extensions of process algebras. Where an operational linear-time semantics is given to these models, it is a sequential semantics, i.e., concurrency is expressed by nondeterminism and a run represents a total order of events[1].

A randomized distributed algorithm has three important features which must be accommodated by a model: (1) There is randomized choice. (2) There is nondeterministic choice. (3) There is, at least in the basic version, no assumption on timing or synchrony. To our knowledge, there is no model based on Petri nets with all three features. We suggest a Petri net based model for randomized distributed algorithms in this paper which we call *randomized Petri nets*. This model provides a basis for incorporation of randomized algorithms into existing Petri net based techniques for distributed algorithms (e.g. [24, 25]).

Furthermore, we present a non-sequential semantics for randomized Petri nets where a run represents a partial order, viz the causal order, of events, thus providing a basis for investigation of the applicability of partial-order verification methods [19] to randomized algorithms. Our semantics is directly based on branching processes of Petri nets [18, 9]. In particular, we do not use a notion of global time or global state to construct the semantics. This absence of explicit global states results in the non-sequential semantics being weaker than the classical sequential semantics of randomized distributed algorithms: there are algorithms that terminate with probability 1 in the non-sequential semantics but not in the sequential semantics. As we will show, that is because, in our non-sequential semantics, a nondeterministic choice never depends on the outcome of a coin flip that happens concurrently to that nondeterministic choice, which can be the case in the sequential semantics.

We proceed as follows. After recalling some preliminaries, we introduce randomized Petri nets in Sect. 3. The classical sequential semantics for randomized distributed algorithms is defined for randomized Petri nets in Sect. 4. In Sect. 5, we introduce the non-sequential semantics with the central notion of a *probabilistic branching process*. As a main result, we show that each probabilistic branching process defines a unique canonical probability space. Sect. 6 defines

[1] When another semantics is considered such as bisimulation semantics, the focus also lies on semantics that are on the "sequential" side of the sequential/ partial-order spectrum of semantics (cf. [29]).

the probabilistic validity of temporal-logical formulas. Sect. 7 provides some examples which directly lead to a comparison of the sequential and the non-sequential semantics in Sect. 8, where we explain why the new semantics models a new adversary. A few remarks conclude the paper.

2 Preliminaries

This section collects some preliminaries, which include Petri nets and their branching processes and a few notions from probability theory. For a motivation of these concepts we refer to the literature. In particular, we refer to Nielsen, Plotkin, and Winskel [18] and Engelfriet [9] for branching processes. The reader who is familiar with these concepts may skip the corresponding paragraphs. The only definition deviating from the literature is the definition of a conflict on page 5.

We denote the set of natural numbers by $\mathbb{N}$, the set of real numbers by $\mathbb{R}$, and the closed interval of real numbers between 0 and 1 by $[0,1]$. Let A be a set. The set of all subsets of A is denoted by 2^A. A *multiset* over A is a mapping $M : A \to \mathbb{N}$. We write $M[x]$ instead of $M(x)$ for the multiplicity of an element x in M. A multiset M is *finite* if $\sum_{x\in A} M[x]$ is finite. Inclusion and addition of multisets are defined elementwise, i.e., $M_1 \leq M_2$ if $\forall x \in A : M_1[x] \leq M_2[x]$ and $(M_1 + M_2)[x] = M_1[x] + M_2[x]$. If we have $M_1 \leq M_2$ then the difference $M_2 - M_1$ is also defined elementwise. A set $A' \subseteq A$ will be treated as a multiset over A by identifying it with its characteristic function.

Petri nets. A *Petri net* (or *net* for short) $N = (P, T, F)$ consists of two disjoint non-empty, countable sets P and T and a binary relation $F \subseteq (P \times T) \cup (T \times P)$. Elements of P, T, and F are called *places*, *transitions*, and *arcs* of the net respectively. We graphically represent a place by a circle, a transition by a square, and an arc by an arrow between the corresponding elements. An element $x \in P \cup T$ is also called an *element* of N. For each element x of N, we define the *preset* of x by ${}^\bullet x = \{y \mid (y,x) \in F\}$ and the *postset* of x by $x^\bullet = \{y \mid (x,y) \in F\}$. The set of *minimal elements* of N and the set of *maximal elements* of N is defined by ${}^\circ N = \{x \mid {}^\bullet x = \varnothing\}$ and $N^\circ = \{x \mid x^\bullet = \varnothing\}$, respectively. For each element x of N, we define the set of *predecessors* of x by $\downarrow x = \{y \mid yF^+x\}$ where F^+ denotes the transitive closure of F. We restrict our attention to nets in which, for each transition t, the preset ${}^\bullet t$ and the postset $t^\bullet$ are non-empty and finite[2]. Therefore, we have ${}^\circ N, N^\circ \subseteq P$.

A net (P', T', F') is a *subnet* of a net (P, T, F) if $P' \subseteq P$ and $T' \subseteq T$ such that, for all $t' \in T'$, we have $(p,t') \in F$ or $(t',p) \in F$ implies $p \in P'$ and $F' = F \cap ((P' \times T') \cup (T' \times P'))$.

A *marking* M of a net is a finite multiset over P. A marking is graphically represented by black tokens in the places of the net. A transition t is *enabled* in a given marking M if ${}^\bullet t \leq M$. If t is enabled in a marking M_1 then t may occur, resulting in the *follower marking* $M_2 = (M_1 - {}^\bullet t) + t^\bullet$. This is denoted $M_1 \xrightarrow{t} M_2$.

[2]This is a usual technical assumption to avoid some anomalies in the non-sequential semantics. See [5] and [9] for further explanation.

A pair $\Sigma = (N, M^0)$ of a net N and a marking M^0 of N is called a *net system*. The marking M^0 is called the *initial marking* of Σ. A set $C \subseteq T$ of transitions of a net such that $|C| > 1$ is called a *conflict* if $\bigcap_{t \in C} {}^\bullet t \neq \varnothing$. A conflict C is *maximal* if there is no conflict that contains C. A conflict C is called *free choice* if, for all $t \in C$, we have $|{}^\bullet t| = 1$, and it is called *extended free choice* if, for all $t_1, t_2 \in C$, we have ${}^\bullet t_1 = {}^\bullet t_2$. A free choice conflict is also extended free choice.

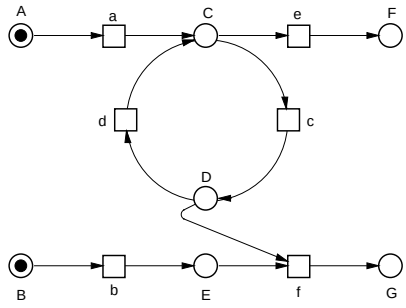


Fig. 1: A net system Σ_1

Fig. 1 shows a net system Σ_1 with its initial marking $\{A, B\}$—we write AB for short. The sets $\{e, c\}$ and $\{f, d\}$ are maximal conflicts of Σ_1 where $\{e, c\}$ is free choice and $\{f, d\}$ is not extended free choice.

A *computation tree* ϑ of $\Sigma = (N, M^0)$ is a (possibly infinite) labelled rooted tree where each node is labelled with a marking of Σ and each edge is labelled with a transition of Σ such that (i) the root is labelled with M^0, (ii) if edge (v_1, v_2) is labelled with transition t and node v_i is labelled with marking M_i for $i = 1, 2$ then we have $M_1 \xrightarrow{t} M_2$, and (iii) if edges (v, v_i) are labelled with t for $i = 1, 2$ then $v_1 = v_2$.

There is a natural prefix order on the set of all computation trees of Σ and a maximal computation tree with respect to this prefix order which is unique up to isomorphism. A *prefix* of a computation tree ϑ is a subtree of ϑ that has the same root and the same labelling as ϑ restricted to its elements. Fig. 2 shows the maximal computation tree of Σ_1. A *sequential run* is a non-branching computation tree of Σ, i.e., a path in a computation tree of Σ starting at the root. A sequential run τ is *maximal* if there is no sequential run that properly extends τ (w.r.t. the prefix order), i.e., if it is either infinite or its final marking does not enable any transition.

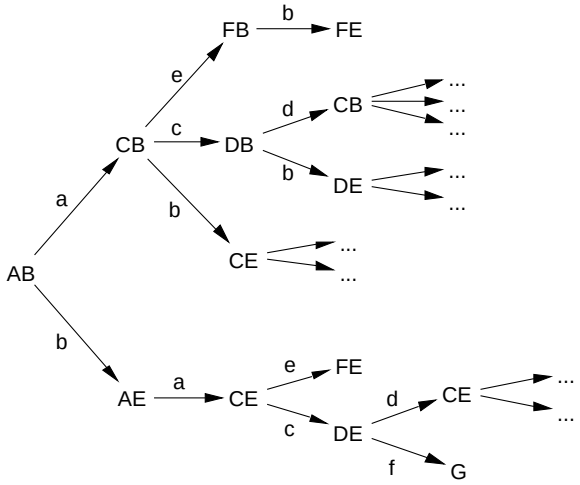


Fig. 2: The maximal computation tree of Σ_1

Branching processes of Petri nets. Let N be a net. N is *acyclic* if, for each element x of N, we have $x \notin \downarrow x$ and N is *predecessor-finite* if, for each element x of N, the set $\downarrow x$ is finite. Let $K = (B, E, \lessdot)$ be an acyclic, predecessor-finite net. A place $b \in B$ is called a *condition* and a transition $e \in E$ is called an *event*. Since K is acyclic, the transitive closure of $\lessdot$, denoted $<$, is a partial order, which we call the *causal order*. If we have $x_1 < x_2$ or $x_2 < x_1$ then we say x_1 and x_2 are *causally dependent*. If we have $x_1 < x_2$ or $x_1 = x_2$ then we write $x_1 \leq x_2$. Two elements are *in (extended) conflict*, denoted $x_1 \# x_2$, if there is a conflict $\{e_1, e_2\}$ such that $e_i \leq x_i$ for $i = 1, 2$. Two different elements x_1 and x_2 are *concurrent*, denoted x_1 co x_2, if they are neither causally dependent nor in conflict.

A predecessor-finite, acyclic net K is called an *occurrence net*, if (i) $^\circ K$ is finite, (ii) for each condition b of K we have $|^\bullet b| \leq 1$, and (iii) there is no event e of K such that $e \# e$. Let $\Sigma = (N, M^0)$ be a net system with $N = (P, T, F)$ and let $K = (B, E, \lessdot)$ be an occurrence net. Let $l : B \cup E \to P \cup T$ be a mapping such that $l(B) \subseteq P$ and $l(E) \subseteq T$. For finite $B' \subseteq B$, we define a marking $l(B')$ of Σ by $l(B')[p] = |\{b \in B' \mid l(b) = p\}|$. The pair $\pi = (K, l)$ is a *branching process* of Σ if (i) $l(^\circ K) = M^0$, (ii) for each event e of K we have $l(^\bullet e) = {}^\bullet l(e)$ and $l(e^\bullet) = l(e)^\bullet$, and (iii) for all events e_1, e_2 of π, we have ${}^\bullet e_1 = {}^\bullet e_2$ and $l(e_1) = l(e_2)$ implies $e_1 = e_2$. We often write E_π for the set of events of a branching process π.

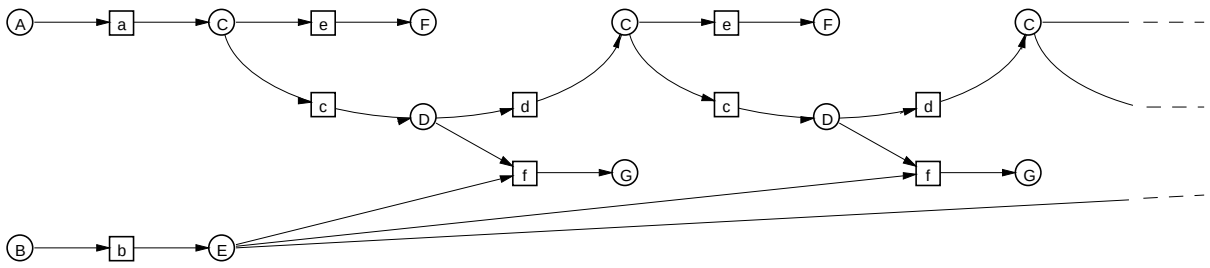


Fig. 3: The maximal branching process π of Σ_1

There is also a natural prefix order $\sqsubseteq$ on the set of branching processes of a net system Σ and a maximal branching process with respect to this prefix order which is unique up to isomorphism[3]. A branching process $\pi' = (K', l')$ is a *prefix* of a branching process $\pi = (K, l)$, denoted $\pi' \sqsubseteq \pi$, if (i) K' is a subnet of K, (ii) $^\circ K' = {}^\circ K$, and (iii) $l' = l\,|_{K'}$. Let $\pi' \sqsubseteq \pi$. We write $\pi' \overset{e}{\sqsubset} \pi$ if $E_\pi = E_{\pi'} \cup \{e\}$, i.e., π is derived from π' by appending a single event e (and its postconditions).

Fig. 3 shows the maximal branching process π of Σ_1 . A branching process of Σ without conflicts is called a *non-sequential run*[4] of Σ. We may omit the attributes "sequential" and "non-sequential" where it is clear from the context. If a run ρ is a prefix of a branching process π then ρ is called a *run of* π. By a *maximal run* of π we mean a run of π which is maximal with respect to the prefix order. We denote the set of all runs of π, the set of all finite runs of π, and the set of all maximal runs of π by $\mathfrak{R}(\pi)$, $\mathfrak{R}_{\text{fin}}(\pi)$, and $\mathfrak{R}_{\text{max}}(\pi)$, respectively. Note that, in contrast to sequential runs, a proper prefix of a non-sequential run is not necessarily finite.

[3] Engelfriet [9] shows that the prefix order constitutes a complete lattice on the isomorphism classes of all branching processes of a net system under some technical assumptions.

[4] It is also called a *process* in the literature.

A *sequential (non-sequential) temporal property* is a set of sequential (non-sequential) runs. A temporal property E such that $\rho \in E$ if and only if ρ satisfies some given formula Φ of a given linear-time temporal logic is called *temporal-logical property*.

Probability. Let Ω be a non-empty set. A family of subsets $\mathcal{A} \subseteq 2^\Omega$ with $\Omega \in \mathcal{A}$ that is closed under complementation and finite union is called a *field* on Ω. A field $\mathcal{A}$ on Ω that is also closed under countable union is called a σ-*field* on Ω. The elements of $\mathcal{A}$ are called *measurable sets*. Let $\mathcal{A}$ be a σ-field on Ω. A mapping $P : \mathcal{A} \to [0,1]$ is a *probability measure* on $\mathcal{A}$ if $P(\Omega) = 1$ and for each countable pairwise disjoint family $\mathcal{F} \subseteq 2^\Omega$ we have

$$P(\bigcup_{A \in \mathcal{F}} A) = \sum_{A \in \mathcal{F}} P(A). \tag{1}$$

The triple $(\Omega, \mathcal{A}, P)$ is called a *probability space* if $\mathcal{A}$ is a σ-field on Ω and P is a probability measure on $\mathcal{A}$; σ-fields are closed under intersection. Therefore, for a given family $\mathcal{E} \subseteq 2^\Omega$, the smallest σ-field containing $\mathcal{E}$, denoted by $\sigma(\mathcal{E})$, is well-defined; $\mathcal{E}$ is called the *generator* of $\sigma(\mathcal{E})$. For two measurable sets A and B, the *conditional probability* of A under B is defined by $P(A \mid B) = \frac{P(A \cap B)}{P(B)}$ if $P(B) \neq 0$.

3 Randomized Petri Nets

This section introduces *randomized Petri nets*, our model for randomized distributed algorithms. We model a coin flip with two outcomes *head* and *tail* by a free choice conflict[5] as in Fig. 4a (the double slashed arrows designate the absence of other arcs to the respective transition).

(a) Simple coin flip

(b) General coin flip

Fig. 4: Modelling coin flips

[5] Guided by our intuition of coin flips, we assign probabilities to free choice conflicts only. An extended free choice conflict can be refined to a free choice conflict by a simple, well-known construction (cf. Sect. 9). This way, we may interpret also an extended free choice conflict as a coin flip. Extension beyond extended free choice is discussed in Sect. 9.

Every outcome has a probability of $\frac{1}{2}$. A transition modelling one outcome of a coin flip is called *probabilistic*. A probabilistic transition is graphically distinguished by the symbol %. A general coin flip with n outcomes is depicted in Fig. 4b. The transitions $t_1, \ldots, t_n$ constitute a free choice conflict. Every t_i is equipped with a probability p_i, depicted at the arc leading to t_i, such that $\sum_{i=1}^{n} p_i = 1$.

A *randomized net* consists of a net where some transitions are distinguished as *probabilistic* and a mapping assigning a probability to each probabilistic transition.

Definition 1 (Randomized net) Let $N = (P, T, F)$ be a net and let $T^{flip} \subseteq T$ be a set of distinguished transitions, called *probabilistic transitions*. A conflict C of N such that $C \subseteq T^{flip}$ is said to be *probabilistic*. Furthermore, let $\mu : T^{flip} \to [0, 1]$ be a mapping such that $0 < \mu(t) < 1$ for all $t \in T^{flip}$. Then, the triple $N' = (N, T^{flip}, \mu)$ is called a *randomized net* if

(i) each probabilistic conflict is finite and free choice, and

(ii) for each maximal probabilistic conflict C, we have

$$\sum_{t \in C} \mu(t) = 1. \tag{2}$$

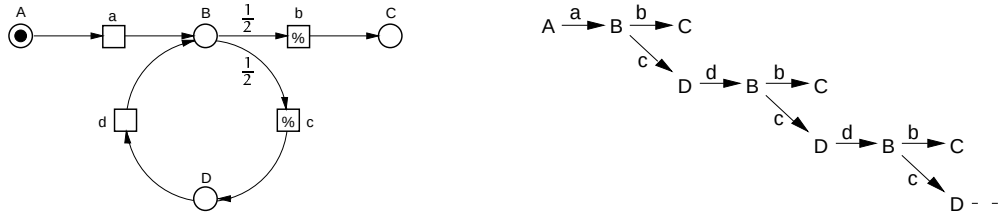
The mapping μ is called the *local coin measure* of N'. A pair $\Sigma = (N, M^0)$ of a randomized net N and a marking M^0 of N is called a *randomized net system*. ∘

Not every conflict of a randomized net system is probabilistic. Non-probabilistic conflicts are solved nondeterministically. Randomized net systems need nondeterminism to model nondeterminism of randomized distributed algorithms: there, we have usually no knowledge about the order of causally independent events and the behaviour of the environment. Moreover, nondeterminism also models freedom of implementation. Note that there may be a non-probabilistic transition in conflict with a probabilistic transition. Such a transition would model the nondeterministic removal of the coin before it is flipped.

4 Probabilistic Computation Trees

This section presents the traditional sequential semantics of randomized distributed algorithms, which we easily carry over from probabilistic programs [21] or probabilistic concurrent processes [1] to randomized net systems.

We begin with a randomized net system without concurrency and without nondeterminism. Fig. 5a shows such a system Σ_2. We expect the proposition "eventually C is marked", denoted $\Diamond C$, to be valid in Σ_2 with probability 1. The meaning of such a proposition is traditionally, e.g., in probabilistic transition systems, defined as in the following paragraph.



(a) Σ_2 (b) ϑ

Fig. 5: A randomized net system and its unique maximal probabilistic computation tree

Fig. 5b shows the maximal computation tree ϑ of Σ_2. Since Σ_2 has neither concurrency nor nondeterminism, every branching in ϑ represents a coin flip, i.e., the local coin measure assigns a conditional probability to every branch such that the sum of the probabilities at every branching is 1. A computation tree where every branching represents a coin flip is called *probabilistic*. A probability space $(\Omega, \mathcal{A}, P)$ is assigned to every probabilistic computation tree ϑ as follows. As a preparation, we assign a probability $p(\tau)$ to every finite sequential run $\tau = M_0 \xrightarrow{t_1} \ldots \xrightarrow{t_n} M_n$ by $p(\tau) = \prod_{i=1,\ldots,n} \mu(t_i)$ where, for $t_i \notin T^{flip}$, we set $\mu(t_i) = 1$, and we set $p(\tau) = 1$ if $n = 0$. For Σ_2, we have $p(A \xrightarrow{a} B \xrightarrow{c} D \xrightarrow{d} B \xrightarrow{c} D) = \frac{1}{4}$.

Let $\Omega = \{\tau \mid \tau \text{ is a maximal run of } \vartheta\}$. In the probability space on Ω to be constructed, the probability $p(\tau)$ is assigned to the set $K(\tau) = \{\tau' \in \Omega \mid \tau \text{ is a prefix of } \tau'\}$. Such a set $K(\tau)$ is called a *cone*. The set of all cones $\mathcal{E} = \{K(\tau) \mid \tau \text{ is a finite run of } \vartheta\}$ constitutes the generator of the σ-field $\mathcal{A}$, i.e., $\mathcal{A} = \sigma(\mathcal{E})$. There is a unique probability space $(\Omega, \mathcal{A}, P)$ such that

$$P(K(\tau)) = p(\tau) \tag{3}$$

holds true for each finite sequential run τ. Each sequential temporal-logical property E is measurable in this probability space. Then, we define E to be *valid with probability* p if $P(E) = p$. Then, $\Diamond C$ is actually valid with probability 1 in Σ_2. The probability space associated with a probabilistic computation tree ϑ is also referred to by $(\Omega_\vartheta, \mathcal{A}_\vartheta, P_\vartheta)$.

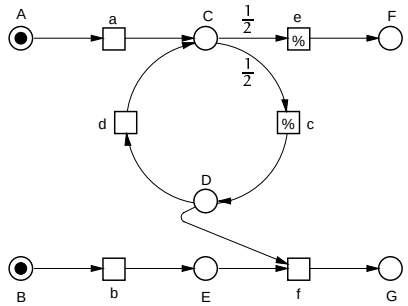


Fig. 6: Σ_3

Fig. 6 shows a randomized net system with concurrency as well as with nondeterminism. Therefore, the maximal computation tree ϑ of Σ_3 is not probabilistic. However we can choose maximal probabilistic computation trees from the prefixes of ϑ (Σ_3 has infinitely many maximal probabilistic computation trees). The choice of a maximal probabilistic computation tree ϑ and therefore of a probability space $(\Omega_\vartheta, \mathcal{A}_\vartheta, P_\vartheta)$ is nondeterministic. Therefore, a sequential temporal-logical property E is defined to be *valid with at least probability* p if, for each maximal probabilistic computation tree ϑ of Σ, we have $P_\vartheta(E) \geq p$. In Σ_3, $\Diamond F$ is valid with at least probability $\frac{1}{2}$ and $\Diamond(F \vee G)$ with (at least) probability 1, i.e., Σ_3 terminates with probability 1.

5 Probabilistic Branching Processes

In analogy to probabilistic computation trees, we introduce *probabilistic branching processes* in this section. A *probabilistic branching process* of a randomized net system Σ is a branching process of Σ such that all conflicts in it are probabilistic, i.e., there is no nondeterminism in a probabilistic branching process. As a main result, we show that there is a unique canonical probability space for each probabilistic branching process.

Definition 2 (Probabilistic branching process) Let $\Sigma = ((N, T^{flip}, \mu), M^0)$ be a randomized net system and $\pi = (K, l)$ be a branching process of Σ with events E. Let $E^{flip} = \{e \in E \mid l(e) \in T^{flip}\}$ denote the set of all *probabilistic events* of π. We carry over μ to probabilistic events: $\mu : E^{flip} \to [0,1]$ by $\mu(e) = \mu(l(e))$. We call a probabilistic conflict $C \subseteq E^{flip}$ of π *complete* if it satisfies (2), i.e., if it contains all possible outcomes of the coin flip. Then, π is called a *randomized branching process* of Σ if all maximal probabilistic conflicts of π are complete[6]. A randomized branching process π of Σ is a *probabilistic branching process* of Σ if each conflict of π is probabilistic, i.e., π does not contain nondeterministic conflicts. ∘

Since every coin flip has only finitely many outcomes, every probabilistic branching process is finitely branching. Fig. 7 shows a finite, maximal probabilistic branching process of Σ_3.

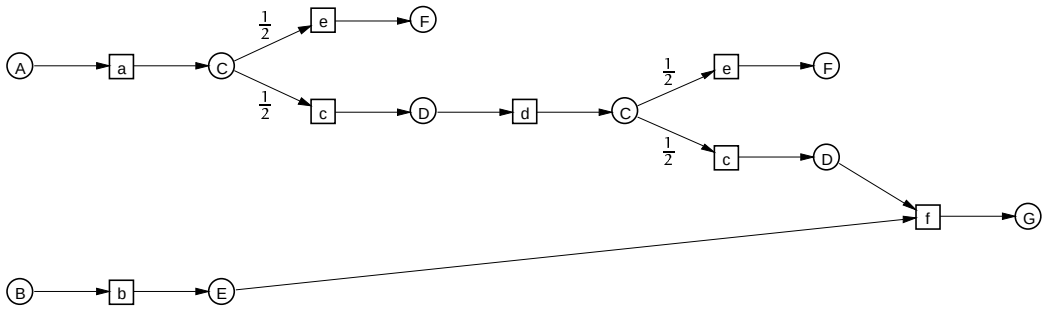


Fig. 7: A finite, maximal probabilistic branching process of Σ_3

We define a probability space for each probabilistic branching process of a randomized net system Σ which is derived from the local coin measure of Σ. To do this, we define for a probabilistic branching process π the probability $p(\alpha)$ that a finite run α of π is realized in π. As in the sequential semantics, we assume the stochastic independence of all coin flips and thus define $p(\alpha)$ to be the product of the probabilities of all probabilistic events occurring in α. Let E^{flip} be the set of probabilistic events of α and μ be defined on E^{flip} as in Def. 2. We set $p(\alpha) = 1$ if $E^{flip} = \varnothing$ and

$$p(\alpha) = \prod_{e \in E^{flip}} \mu(e) \tag{4}$$

otherwise.

[6]Then, the triple (K, E^{flip}, μ) is a randomized (occurrence) net.

Theorem 1 (Probability space of a probabilistic branching process) Let Σ be a randomized net system and π be a probabilistic branching process of Σ. Let $\Omega = \mathfrak{R}_{\max}(\pi)$ and for every finite run α of π, let $K(\alpha) = \{\rho \in \Omega \mid \alpha \sqsubseteq \rho\}$ be the set of maximal runs of π that extend α. Furthermore, let $\mathcal{E} = \{K(\alpha) \mid \alpha \in \mathfrak{R}_{\text{fin}}(\pi)\}$. Then, there exists a unique probability space $(\Omega, \mathcal{A}, P) = (\Omega_\pi, \mathcal{A}_\pi, P_\pi)$ such that $\mathcal{A} = \sigma(\mathcal{E})$ and, for all finite runs α of π, we have

$$P(K(\alpha)) = p(\alpha). \tag{5}$$

The proof of Theorem 1 is given in Appendix A. It mainly exploits the lattice properties of branching processes. The proof rests upon the fact that all conflicts in a probabilistic branching process are free choice and finite.

We give a proof sketch here. A set $K(\alpha)$ for a finite run α of π is called a *cone*. We have to construct a probability measure P over $\sigma(\mathcal{E})$, which is already defined on $\mathcal{E}$, i.e., on all cones, by (5). For that we have to show that P can be extended to all complements of cones as well as to unions of cones and cone complements. The key step in the proof is the identification of a field on Ω, that contains $\mathcal{E}$ and that generates $\sigma(\mathcal{E})$. The field $\mathcal{M}$ that we are looking for contains exactly all finite unions of pairwise disjoint cones, i.e.,

$$\mathcal{M} = \{\bigcup_{A \in \mathcal{F}} A \mid \mathcal{F} \subseteq \mathcal{E} \text{ is finite and pairwise disjoint}\}. \tag{6}$$

The mapping P can now be extended to the field $\mathcal{M}$ by (1). To show that this extension is well-defined requires the biggest effort in the proof. Further extension of P from $\mathcal{M}$ to the σ-field $\sigma(\mathcal{M}) = \sigma(\mathcal{E})$ is an application of the extension theorem, a standard theorem of measure theory. The uniqueness of the probability space follows from its construction.

6 Probabilistic Validity of Temporal Properties

In this section, we define what it means for a non-sequential temporal property to be valid with at least probability p in a given randomized net system. To do so, we have to restrict our attention to *measurable temporal properties*. A temporal property E is *measurable* in a randomized net system Σ if for every probabilistic branching process π of Σ, E is measurable in the probability space associated with π.

Proposition 1 Every temporal-logical property is measurable in each randomized net system.

As in the sequential semantics, Proposition 1 is easily shown by induction over the structure of temporal-logical formulas. The proof is given in Appendix B. We now define the probabilistic validity of measurable temporal properties.

Definition 3 (Probabilistic validity) Let Σ be a randomized net system and E be a temporal property that is measurable in Σ. We say that E *is valid with at least probability* p if, for every maximal probabilistic branching process π of Σ, we have[7]

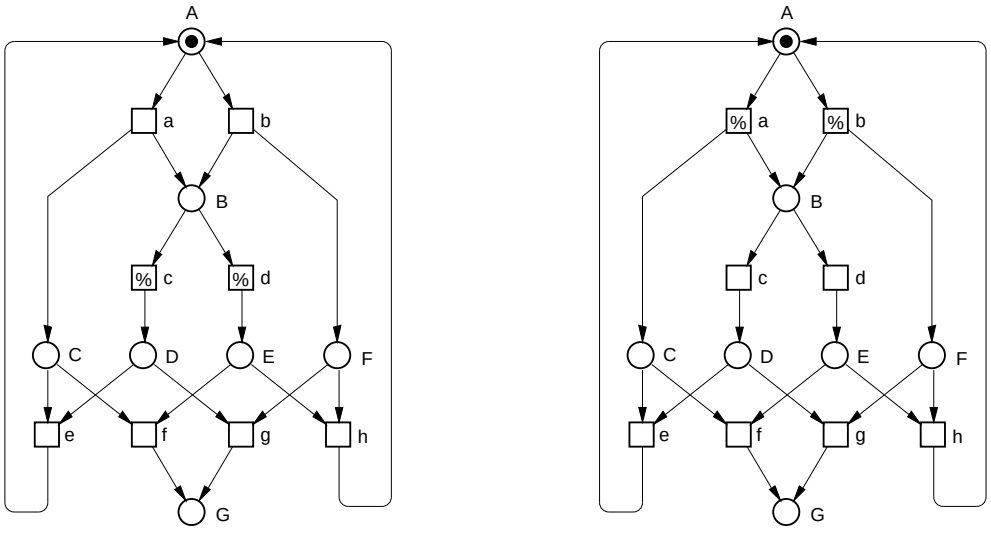
$$P_\pi(E) \geq p, \tag{7}$$

where P_π is the probability measure for π defined in Theorem 1. ∘

In Def. 3, we take maximality of non-sequential runs[8] as the basic liveness assumption of our semantics. To change to another notion of admissible runs, call a probabilistic branching process admissible if all its runs are admissible[9] and substitute "maximal" in Def. 3 by "admissible".

7 Examples

In the following, we omit the graphical representation of concrete probabilities because they do not play any role anymore. In each of the randomized net systems Σ_4 and Σ_5, depicted in Fig. 8, two free choice conflicts are solved, one after the other. Dependent on the solution of those conflicts, exactly one of the four transitions e, f, g, and h is then enabled.



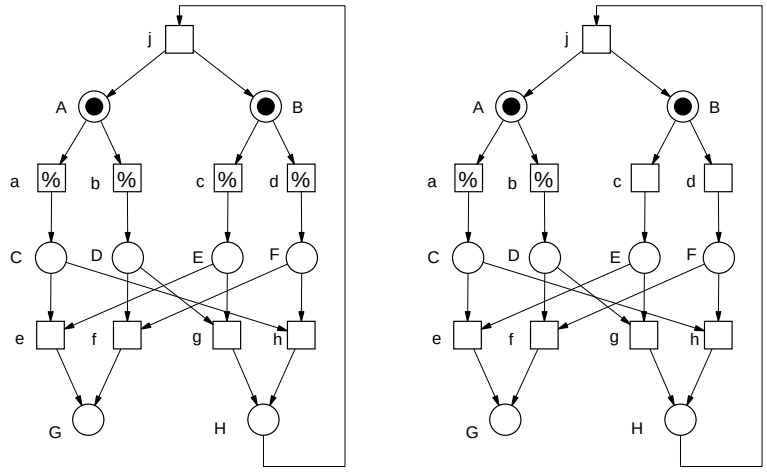
(a) Σ_4 (b) Σ_5

Fig. 8: Two randomized net systems

[7] We write here and in the following $P_\pi(E)$ for $P_\pi(E \cap \mathfrak{R}_{\max}(\pi))$.

[8] Also sometimes called *progress* in the literature.

[9] Or alternatively, if the set of all admissible runs has probability 1; the difference between these notions is discussed in [11].



(a) Σ_6 (b) Σ_7

Fig. 9: Coordination of concurrent decisions through randomization

In Σ_4, the conflict between c and d is probabilistic. Therefore, whether c or d occurs does not depend on whether a or b occurred before; Σ_4 terminates with probability 1, i.e., $\Diamond G$ is valid with probability 1 in Σ_4.

In Σ_5, not the conflict between c and d, but the conflict between a and b is probabilistic. The solution of the nondeterministic conflict between c and d may depend on whether a or b occurred before: There is a maximal probabilistic branching process π of Σ_5 such that each occurrence of a is followed by an occurrence of c and each occurrence of b is followed by an occurrence of d. In that π, no run is finite, and therefore, Σ_5 does not terminate with probability 1.

Σ_6 and Σ_7 in Fig. 9 are similar to the randomized net systems in Fig. 8. Again, two free choice conflicts are solved but this time not sequentially but concurrently to each other. In Σ_6, both conflicts are probabilistic; Σ_6 terminates with probability 1. This is a prominent example for randomization in distributed algorithms—both conflicts will eventually be solved *coordinated* to each other—without synchronization. Such iterated concurrent coin flips occur in many randomized distributed algorithms in the literature.

In Σ_7, only one of the two free choice conflicts is probabilistic; Σ_7 terminates with probability 1, but only in the non-sequential semantics. In contrast to Σ_6, Σ_7 does not terminate with probability 1 in the sequential semantics, because there is a probabilistic computation tree of Σ_7 where every maximal run is infinite. This discrepancy between the sequential and the non-sequential semantics is further explained in the next section.

8 Non-sequential vs. Sequential Semantics

In this section, we explain the discrepancy between the sequential semantics and the non-sequential semantics of randomized net systems.

Sometimes, it is useful to adjust a semantics for randomized distributed algorithms. Differences between differently adjusted semantics are usually described by help of the notion of an *adversary* (sometimes also called *scheduler*). By that we imagine that we have two players, a coin flipper and an adversary, who interact to determine a run of a given randomized algorithm. For every finite interleaving $\tau = M_0, \ldots, M_n$, the adversary chooses either a non-probabilistic transition enabled in M_n—which then occurs—or a maximal probabilistic conflict—which is then resolved by the coin flipper by flipping a coin. Each behaviour of the adversary generates exactly one probabilistic computation tree.

An adjustment of the sequential semantics restricts the behaviour of an adversary. Usually, two kinds of restrictions are distinguished (cf.[27]): *execution-based adversaries* and *adversaries with partial on-line information*. For an execution-based adversary, we restrict the interleavings that are generated by the adversary. For example, we may postulate that the adversary is fair with respect to the solution of nondeterministic conflicts. For an adversary with partial on-line information, we restrict the knowledge the adversary is able to base its decisions upon. For example, we may postulate that the adversary does not know the entire history but only the current state. (If additionally, the adversary does not know the outcome of prior coin flips then the randomized net system in Fig. 8b terminates with probability 1 under this adversary.)

The following example suggests that the adversary associated with the non-sequential semantics can be viewed as an adversary with partial on-line information with respect to the adversary associated with the sequential semantics. This adversary, we call it the *distributed adversary*, has in contrast to the sequential adversary, no knowledge about non-causal order of events. In particular, the distributed adversary cannot make the solution of a nondeterministic conflict depend on the outcome of coin flips that happen concurrently to that conflict, i.e., the distributed adversary can only make the solution of a nondeterministic conflict depend on the causal history of that conflict. To see this, we consider Fig. 10.

Fig. 10 shows the randomized net system Σ_8, its two maximal probabilistic branching processes π_1 and π_2 and its six probabilistic computation trees ϑ_1 to ϑ_6. From each probabilistic branching process, we can derive a set of probabilistic computation trees by sequentialization. From π_1, we derive ϑ_1 and ϑ_2 and from π_2, we derive ϑ_3 and ϑ_4. Σ_8 shows that not every probabilistic computation tree can be derived from a probabilistic branching process: ϑ_5 and ϑ_6 can neither be assigned to π_1 nor to π_2. The computation trees ϑ_5 and ϑ_6 represent a behaviour of the sequential adversary where a choice depends on the outcome of a coin flip that is concurrent to that choice. A similar behaviour of the sequential adversary prevents the termination of the randomized net system in Fig. 9b when c always occurs concurrently to b and d always occurs concurrently to a.

The characteristic property of the probability spaces of π_1, π_2 and ϑ_1 to ϑ_4 which distinguishes them from the probability spaces of ϑ_5 and ϑ_6 is the following. Let α be a finite run of a

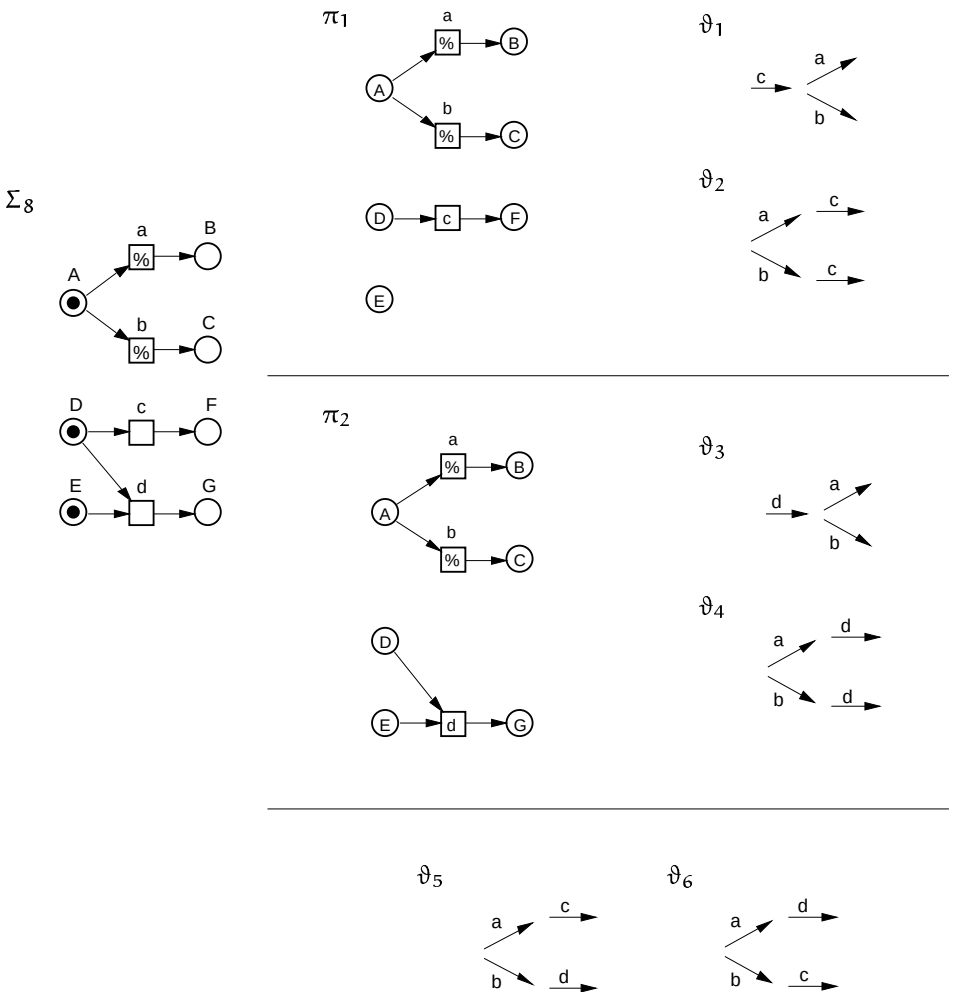


Fig. 10: Non-sequential vs. sequential semantics

probabilistic branching process π and let e be an event of π that is enabled at the end of α, i.e., α can be extended to some finite run β of π by appending e. Then, the occurrence of a concurrent event e' does not influence the conditional probability of e. More precisely, if e' is another event that is enabled in α such that e and e' are concurrent and we denote the resulting run when e' is appended to α by α' then we have

$$P(e \mid \alpha) = P(e \mid \alpha'), \tag{8}$$

where $P(e \mid \alpha)$ denotes the conditional probability that e occurs provided that α is realized[10]. Equation (8) holds in the probability space of each probabilistic branching process but not necessarily in the probability space of a probabilistic computation tree. That means that the

[10]Formally, e stands here for the measurable set $\{\rho \in \Omega \mid e \text{ is an event of } \rho\}$ and α stands for the measurable set $K(\alpha)$, defined in Thm. 1.

order of concurrent events influences probability in the sequential semantics. We extend this discussion in Section 9.

Note that it is easy to simulate the sequential semantics in the non-sequential semantics by sequentializing the randomized net system, i.e., by adding an initially marked place p to the net and connecting each transition t of the net with p by a loop, i.e., adding (p,t) and (t,p) to F. That means that the non-sequential semantics discriminates more systems than the sequential semantics does. This is of course already the case for non-probabilistic systems.

Besides the different interaction of probabilistic choice and randomized choice in sequential and non-sequential semantics, there is another technical difference between the two semantics. While Definition 3 uses non-sequential maximality as the basic liveness assumption, sequential semantics uses sequential maximality, which is, in general, weaker than non-sequential maximality. In order to focus on the different interaction of randomized and nondeterministic choice, we will use below a liveness assumption in the sequential semantics that corresponds to the liveness assumption in the non-sequential semantics.

We now make the relationship between the sequential and the non-sequential semantics precise. In order to do so, we have to define a correspondence of sequential and non-sequential temporal properties. The obvious way to do this is the sequentialization of non-sequential runs. Let ρ be a non-sequential run. An alternating sequence $\tau = \alpha_0, e_1, \alpha_1, \ldots$ of finite prefixes and events of ρ is an *interleaving* of ρ if α_0 is the minimal prefix of ρ, we have $\alpha_{i-1} \stackrel{e_i}{\sqsubset} \alpha_i$ for all positions $i > 0$ of τ, and for all events e of ρ there exists a position i such that $e = e_i$. Each interleaving τ of $\rho = (K, l)$ represents a sequential run $l(\tau) = l(\alpha_0^\circ), l(e_1), \ldots$ If ρ is maximal so is τ but not necessarily vice versa. A sequential run that is represented by some interleaving of a maximal run is called *progressive*. The set of sequential runs that is represented by a set Q of non-sequential runs is denoted by $\mathrm{seq}(Q)$.

Let Q be a set of non-sequential runs of Σ, called *admissible runs*. A branching process π is *admissible* if each run of π is admissible. A sequential run τ of Σ is *admissible* if $\tau \in \mathrm{seq}(Q)$. A computation tree is *admissible* if each sequential run of it is admissible. We write $\Sigma \stackrel{p}{\models} E$ if a non-sequential temporal property E is measurable and satisfied with at least probability p in each admissible probabilistic branching process of Σ and we write $\Sigma \stackrel{p}{\vdash} E$ if a sequential temporal property E is measurable and satisfied with at least probability p in each admissible probabilistic computation tree of Σ.

Theorem 2 The distributed adversary is weaker than the classical adversary, that is, if E is a non-sequential temporal property, we have

$$\Sigma \stackrel{p}{\vdash} \mathrm{seq}(E) \quad \Rightarrow \quad \Sigma \stackrel{p}{\models} E. \tag{9}$$

The proof of Theorem 2 is given in Appendix C. The converse of (9) is not true as the system in Fig. 9b proves. This is in contrast to a corresponding result for non-probabilistic systems Σ, where we have

$$\Sigma \vdash \mathrm{seq}(E) \quad \Leftrightarrow \quad \Sigma \models E, \tag{10}$$

where $\Sigma \vdash E$ ($\Sigma \models E$) means that all admissible sequential runs (non-sequential runs) of Σ satisfy a sequential (non-sequential) temporal property E. (10) means that the two semantics do not differ if there is no randomization. They also do not differ for systems without nondetermistic choice, for which we have "$\Leftrightarrow$" in (9), which follows from the results in Appendix C. The two semantics thus differ in the interaction of randomization and non-determinism.

9 Conclusion

We have shown that it is possible to give a semantics to randomized distributed algorithms that separates concurrency and nondeterminism and that does not rely on some ordering of concurrent events. That semantics lead to the notion of a distributed adversary. The distributed adversary is weaker than the classical "global" adversary in that it knows only local states and their histories while the classical adversary knows global states and their histories.

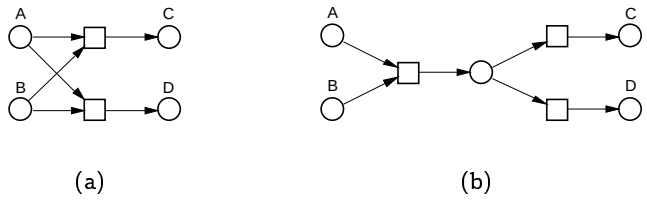


Fig. 11: Refinement of an extended free choice conflict

Motivated by the application domain of randomized distributed algorithms, we restricted probabilistic conflicts to be free choice conflicts, which implies that all outcomes of the same coin flip have the same causal history. As already noted, we could easily include probabilistic extended free choice conflicts as well. Fig. 11 shows the well-known construction (e.g. [4]) to transform an extended-free choice conflict into a free choice conflict. However, if we want to go beyond extended free choice then we have to deal with *confusion* (cf. [26, 28]), that is a situation where the enabling of a conflict depends on some order of concurrent events. We would then inevitably need a notion of time, i.e., we have to order concurrent events, and we have to give up property (8) to construct a probability space. To see this, consider the branching process π in Fig. 12, which contains a conflict that is not extended free choice. If we want to construct a probability space on all maximal runs of π such that, for each event e of π, we have $P(e) \neq 0$ then, clearly, we have $P(d \mid \alpha_1) = 1$ and $P(d \mid \alpha_0) \neq 1$ because $P(b) \neq 0$ and $P(c) \neq 0$, hence $P(d \mid \alpha_1) \neq P(d \mid \alpha_0)$. Thus, in such a probability space, the conditional probabilities of events have to depend on some ordering of concurrent events. Then we would come closer to models motivated by performance analysis such as stochastic Petri nets [16] where we also find models combining causality and probability (e.g. [6]).

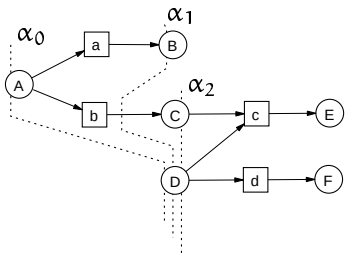


Fig. 12: A branching process π with confusion

As the distributed adversary[11] is weaker than the sequential adversary, it might admit a more efficient solution to a given synchronization- or coordination problem. This conjecture will be the subject of future work. To assume a distributed adversary rather than a sequential adversary may result in a more faithful model in many cases. However, we must not introduce new causality when implementing an algorithm which was proven correct with respect to the non-sequential semantics–unless we take some measures such as encryption to hide the knowledge to the adversary that is provided by the new causality.

The adoption of verification techniques developed for the sequential semantics such as *extreme fairness* [20], *α-fairness* [15], fairness for labels [1], and *computable fairness* [13] to the non-sequential semantics is straight-forward because these techniques do not depend on specific properties of sequential runs. However, these adoptions will reflect the discrepancy between the two semantics.

In further research, we would like to study the behaviour of the semantics under refinement and composition. In independent work, de Alfaro, Henzinger, and Jhala [8] describe a compositional semantics for probabilistic systems and their adversary is similar to our distributed adversary. Furthermore, we would like to study the applicability of partial-order verification techniques [19] to randomized net systems.

Finally, since the non-sequential semantics strictly separates nondeterminism, randomization, and concurrency, we hope to better understand the subtle interaction of these three components in randomized distributed algorithms which is often blamed for the difficulty to verify these algorithms. This hope is already substantiated as we were able to prove [30], with the new semantics, two new impossibility results with respect to the power of randomized distributed algorithms, where one of these results was discovered during the development of the semantics.

Acknowledgement. We thank Stefan Haar for substantial comments on earlier versions of this material.

[11] Actually, we get a class of distributed adversaries since we can further restrict the knowledge of the distributed adversary.

References

[1] Christel Baier and Marta Kwiatkowska. On the verification of qualitative properties of probabilistic processes under fairness constraints. *Information Processing Letters*, 66:71–79, 1998.

[2] Heinz Bauer. *Probability theory and elements of measure theory.* Academic Press, 2nd edition, 1981.

[3] Michael Ben-Or. Another advantage of free choice: Completely asynchronous agreement protocols. In *Proceedings of the 2nd Annual ACM Symposium on Principles of Distributed Computing*, pages 27–30. ACM, 1983.

[4] Eike Best. Adequacy properties of path programs. *Theoretical Computer Science*, 18:149–171, 1982.

[5] Eike Best and César Fernández. *Nonsequential Processes*, volume 13 of *EATCS Monographs on Theoretical Computer Science.* Springer-Verlag, 1988.

[6] Ed Brinksma, Joost-Pieter Katoen, Rom Langerak, and Diego Latella. A stochastic causality-based process algebra. *The Computer Journal*, 38(7):552–565, 1995.

[7] K. Mani Chandy and Jayadev Misra. *Parallel Program Design: A Foundation.* Addison-Wesley, 1988.

[8] Luca de Alfaro, Thomas A. Henzinger, and Ranjit Jhala. Compositional methods for probabilistic systems. In *Proc. CONCUR 2001 – 12th International Conference on Concurrency Theory, Aalborg, Denmark*, volume 2154 of *LNCS*, pages 351–365. Springer, August 2001.

[9] Joost Engelfriet. Branching processes of Petri nets. *Acta Informatica*, 28:575–591, 1991.

[10] Rajiv Gupta, Scott A. Smolka, and Shaji Bhaskar. On randomization in sequential and distributed algorithms. *ACM Computing Surveys*, 26(1):7–86, March 1994.

[11] Sergiu Hart, Micha Sharir, and Amir Pnueli. Termination of probabilistic concurrent programs. *ACM Transactions on Programming Languages and Systems*, 5(3):356–380, July 1983.

[12] Alon Itai and Michael Rodeh. Symmetry breaking in distributive networks. In *Proc. 22nd Annual Symposium on Foundations of Computer Science*, pages 150–158. IEEE, 1981.

[13] Manfred Jaeger. Fairness, computable fairness, and randomness. In *Proc. 2nd PROBMIV Int. Workshop on Probabilistic Methods in Verification.* Technical Report CSR-99-8 School of Computer Science, University of Birmingham, 1999.

[14] Ralph E. Johnson and Fred B. Schneider. Symmetry and similarity in distributed systems. In *Proceedings of the 4th Annual ACM Symposium on Principles of Distributed Computing.* ACM, 1985.

[15] Orna Lichtenstein, Amir Pnueli, and Lenore Zuck. The glory of the past. In *Proc. Workshop on Logics of Programs*, volume 193 of *LNCS*, 1985.

[16] M. Ajmone Marsan, G. Balbo, G. Conte, S. Donatelli, and G. Franceschinis. *Modelling with Generalized Stochastic Petri Nets.* Series in Parallel Computing. Wiley, 1995.

[17] Carroll Morgan, Annabelle McIver, and Karen Seidel. Probabilistic predicate transformers. *ACM Transactions on Programming Languages and Systems*, 18(3):325–353, May 1996.

[18] Mogens Nielsen, Gordon Plotkin, and Glynn Winskel. Petri nets, event structures and domains, part I. *Theoretical Computer Science*, 13:85–108, 1981.

[19] Doron A. Peled, Vaughan R. Pratt, and Gerard J. Holzmann. *Partial Order Methods in Verification, DIMACS Workshop, Proceedings*, volume 29 of *DIMACS Series in Discrete Mathematics and Theoretical Computer Science*. AMS, 1996.

[20] Amir Pnueli. On the extremely fair treatment of probabilistic algorithms. In *Proc. 15th Annual Symposium on Theory of Computing (STOC)*, pages 278–290. ACM, 1983.

[21] Amir Pnueli and Lenore D. Zuck. Probabilistic verification. *Information and Computation*, 103:1–29, 1993.

[22] Michael O. Rabin. The choice coordination problem. *Acta Informatica*, 17:121–134, 1982.

[23] Josyula Rao. Reasoning about probabilistic algorithms. In *Proceedings of the 9th Annual ACM Symposium on Principles of Distributed Computing*, pages 247–264. ACM, 1990.

[24] Wolfgang Reisig. *Elements of Distributed Algorithms: Modeling and Analysis with Petri Nets*. Springer, 1998.

[25] Wolfgang Reisig, Ekkart Kindler, Tobias Vesper, Hagen Völzer, and Rolf Walter. Distributed algorithms for networks of agents. In *Lectures on Petri Nets II: Applications*, volume 1492 of *LNCS*, pages 331–385. Springer, 1998.

[26] Grzegorz Rozenberg and Joost Engelfriet. Elementary net systems. In Wolfgang Reisig and Grzegorz Rozenberg, editors, *Lectures on Petri Nets I: Basic Models*, volume 1491 of *LNCS*, pages 12–121. Springer, 1998.

[27] Roberto Segala. *Modeling and Verification of Randomized Distributed Real-Time Systems*. Technical report mit/lcs/tr-676, MIT, Laboratory for Computer Science, June 1995.

[28] Einar Smith. On the border of causality: contact and confusion. *Theoretical Computer Science*, 153:245–270, 1996.

[29] Rob van Glabbeek, Scott A. Smolka, and Bernhard Steffen. Reactive, generative and stratified models of probabilistic processes. *Information and Computation*, 121(1):59–80, 1995.

[30] Hagen Völzer. *Fairneß, Randomisierung und Konspiration in Verteilten Algorithmen*. PhD thesis, Humboldt-Universität zu Berlin, Institut für Informatik, December 2000. in German, available via `http://dochost.rz.hu-berlin.de/dissertationen/voelzer-hagen-2000-12-08`.

[31] Hagen Völzer. Randomized non-sequential processes. In *Proc. CONCUR 2001 – 12th International Conference on Concurrency Theory, Aalborg, Denmark*, volume 2154 of *LNCS*, pages 184–201. Springer, August 2001.

A Construction of the Probability Space of a Probabilistic Branching Process

First we recall the theorem we want to prove. Let Σ be a randomized net system and π be a probabilistic branching process of Σ. Let $\Omega = \mathfrak{R}_{\max}(\pi)$ and for every finite run α of π, let $K(\alpha) = \{\rho \in \Omega \mid \alpha \sqsubseteq \rho\}$ be the set of maximal runs of π that extend α. Furthermore, let $\mathcal{E} = \{K(\alpha) \mid \alpha \in \mathfrak{R}_{\text{fin}}(\pi)\}$. Then, there exists a unique probability space $(\Omega, \mathcal{A}, P)$ such that $\mathcal{A} = \sigma(\mathcal{E})$ and for all finite runs α of π we have

$$P(K(\alpha)) = p(\alpha), \tag{5}$$

where p was defined by (4).

In order to prove the theorem, we provide the necessary notions of elementary measure theory in Sect. A.1. In the succeeding sections, we construct the probability space. We start with some notations.

Let $\pi_i \sqsubseteq \pi$ for $i = 1, 2$ be two prefixes of π. Let $\inf(\pi_1, \pi_2)$ denote the greatest prefix κ of π such that $\pi_i \sqsubseteq \kappa$ for $i = 1, 2$ and let $\sup(\pi_1, \pi_2)$ denote the smallest prefix κ of π such that $\kappa \sqsubseteq \pi_i$ for $i = 1, 2$. Likewise, let sup and inf also be defined for finite sets of prefixes. Two runs ρ_1, ρ_2 of π are *compatible*, denoted $\rho_1 \parallel \rho_2$, if $\sup(\rho_1, \rho_2) \in \mathfrak{R}(\pi)$. Otherwise they are *incompatible*, denoted $\rho_1 \nparallel \rho_2$. The smallest prefix α_0 of π, i.e., the run representing the initial marking of Σ, is also called the *eventless prefix* or the *eventless run* of π.

A.1 Preliminaries

The following notions and results can, for example, be found in [2]. Let Ω be a non-empty set. For a subset $A \subseteq \Omega$, let $\overline{A} = \Omega \setminus A$ denote its complement.

Let $\mathcal{E} \subseteq 2^\Omega$ be a family of subsets of Ω. A mapping $P : \mathcal{E} \to \mathbb{R} \cup \{\infty\}$ is called a *set function*, if $P(A) \geq 0$ for all $A \in \mathcal{E}$. A set function P is *finitely additive* if for each finite, pairwise disjoint family $\mathcal{F} \subseteq \mathcal{E}$, we have

$$P\Big(\bigcup_{A \in \mathcal{F}} A\Big) = \sum_{A \in \mathcal{F}} P(A). \tag{11}$$

P is σ-*additive* if (11) holds also for countable, pairwise disjoint families $\mathcal{F}$. A finitely additive set function P on a field $\mathcal{M}$ is called *content* on $\mathcal{M}$ if

$$P(\varnothing) = 0. \tag{12}$$

A σ-additive content on $\mathcal{M}$ is called *premeasure* on $\mathcal{M}$. A premeasure P on $\mathcal{M}$ is *finite* if for each $A \in \mathcal{M}$, we have $P(A) < \infty$. A premeasure defined on a σ-field $\mathcal{A}$ is called a *measure* on $\mathcal{A}$.

Theorem (Extension theorem) Every finite premeasure P on a field $\mathcal{M}$ has a unique extension to $\sigma(\mathcal{M})$, i.e., there exists a unique measure P' on $\sigma(\mathcal{M})$ such that $P'(A) = P(A)$ for all $A \in \mathcal{M}$.

A measure P such that

$$P(\Omega) = 1 \tag{13}$$

is called a *probability measure*.

We construct the desired probability space in two steps. First we construct a field $\mathcal{M}$ on Ω that generates $\mathcal{A}$ in Sect. A.2. Then, we construct a finite premeasure on $\mathcal{M}$ in Sect. A.3. We then use the extension theorem to conclude the existence of a unique extension of the premeasure to $\sigma(\mathcal{M}) = \mathcal{A}$.

A.2 Construction of the Field

Let π, Ω and $\mathcal{E}$ be as in Theorem 1. In this section, we construct the smallest field that contains $\mathcal{E}$. First we observe that $\Omega \in \mathcal{E}$ since $\Omega = K(\alpha_0)$ where α_0 denotes the eventless run of π. A set $K(\alpha) \in \mathcal{E}$ and the empty set is also called a *cone*. The family of all cones is denoted by $\mathcal{K} = \mathcal{E} \cup \{\varnothing\}$.

We prove now some propositions on cones. Let $\alpha, \beta \in \mathfrak{R}_{\text{fin}}(\pi)$. Then, it follows directly from the definition of compatibility that

$$\alpha \not\parallel \beta \quad \Leftrightarrow \quad K(\alpha) \cap K(\beta) = \varnothing. \tag{14}$$

Lemma 1 The intersection of two cones is a cone.

Proof: If one of the two cones to be intersected is the empty set then the claim is trivial. Let $K(\alpha), K(\beta) \in \mathcal{E}$ be two non-empty cones. Then, we have either $\alpha \not\parallel \beta$ or $\alpha \parallel \beta$. In the former case, $K(\alpha) \cap K(\beta) = \varnothing$ follows. In the latter case, $K(\alpha) \cap K(\beta) = K(\sup(\alpha, \beta))$ follows. □

Lemma 2 The complement of a cone equals the union of some finite and pairwise disjoint family of cones.

Proof: Let α be a finite run of π. Let $\overline{\alpha} = \{\beta \in \mathfrak{R}_{\text{fin}}(\pi) \mid \beta \not\parallel \alpha \text{ and } \exists e : \inf(\alpha, \beta) \overset{e}{\sqsubset} \beta\}$ the set of all minimal finite runs that are incompatible with α. (If κ is the branching process that contains α and all its immediate conflicts in π then $\overline{\alpha} = \mathfrak{R}_{\max}(\kappa) \setminus \{\alpha\}$.)

Illustration: Fig. 13 depicts an example for a finite run α together with its conflicts in π (conflicting events dashed). For this example, we have $\overline{\alpha} = \{\{b, f, g\}, \{a, d, f, g\}, \{a, c, e\}\}$.

We obtain a run $\beta \in \overline{\alpha}$ if one event in α is replaced by a conflicting event. For example, we obtain $\{a, c, e\}$ from α by replacing f by e. The event g is deleted by that because g is causally

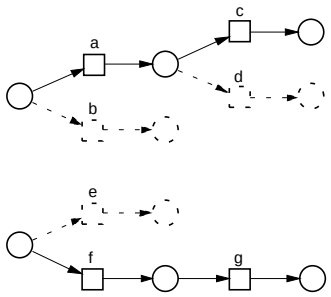


Fig. 13: A finite run α and its conflicts

dependent on f. Thus we have for each $\beta \in \overline{\alpha}$: There is exactly one event in β that does not belong to α but there may be several events in α that do not belong to β. **End of illustration.**

Then, we have

$$\overline{K(\alpha)} = \bigcup_{\beta \in \overline{\alpha}} K(\beta). \tag{15}$$

Furthermore, we have: $\overline{\alpha}$ is finite and the associated family of cones $\{K(\beta) \mid \beta \in \overline{\alpha}\}$ is pairwise disjoint. First we show (15).

1a) $\subseteq$: Let $\rho \in \overline{K(\alpha)}$. Then, we have $\rho \not\parallel \alpha$. Therefore, there is an event e of ρ that is in immediate conflict to some event of α, hence there is a finite run β such that $\inf(\alpha, \rho) \overset{e}{\sqsubset} \beta \sqsubseteq \rho$. We have $\beta \in \overline{\alpha}$ and $\rho \in K(\beta)$.

1b) $\supseteq$: If $\rho \in K(\beta)$ for $\beta \in \overline{\alpha}$ then, because of $\alpha \not\parallel \beta$, and hence $\alpha \not\parallel \rho$, we have $\rho \notin K(\alpha)$ by (14).

2) The set $\overline{\alpha}$ is finite since α is finite and π is finitely branching, i.e., each conflict in π is finite.

3) Let $\beta_i \in \overline{\alpha}$ such that $\inf(\alpha, \beta_i) \overset{e_i}{\sqsubset} \beta_i$ for $i = 1, 2$. Since $\alpha \not\parallel \beta_i$, each e_i is in conflict with each event of $\alpha \setminus \inf(\alpha, \beta_i)$. Let e be the event of α such that e_1 is in immediate conflict with e. If, in case (a), e is an event of $\inf(\alpha, \beta_2)$ then e is also an event of β_2 and therefore $\beta_1 \not\parallel \beta_2$. If, in case (b), e is an event of $\alpha \setminus \inf(\alpha, \beta_2)$ then $e \# e_2$, and because e is in immediate conflict with e_1 and because of the conflict relation being transitive due to free choice, also $e_1 \# e_2$. Hence $\beta_1 \not\parallel \beta_2$. From $\beta_1 \not\parallel \beta_2$ it follows that $K(\beta_1) \cap K(\beta_2) = \varnothing$.

□

Corollary 4 The difference and the union of two cones equal the union of some finite and pairwise disjoint family of cones.

Proof: The claim follows from the Lemmas 1 and 2 by application of DeMorgan's laws:

$$K(\alpha) \setminus K(\beta) \;=\; K(\alpha) \cap \overline{K(\beta)} \;=\; K(\alpha) \cap \bigcup_{\gamma \in \overline{\beta}} K(\gamma) \;=\; \bigcup_{\gamma \in \overline{\beta}} (K(\alpha) \cap K(\gamma))$$

$$K(\alpha) \cup K(\beta) \;=\; (K(\alpha) \setminus K(\beta)) \;\cup\; (K(\beta) \setminus K(\alpha)) \;\cup\; (K(\alpha) \cap K(\beta)).$$ □

We can now present the field we were looking for.

Lemma 3 The family

$$\mathcal{M} = \{ \bigcup_{A \in \mathcal{F}} A \mid \mathcal{F} \subseteq \mathcal{K} \text{ is finite and pairwise disjoint} \}$$

is a field on Ω.

Proof: We have $\Omega \in \mathcal{M}$. From Corollary 4, it follows that $\mathcal{M}$ is closed under finite union and complementation. □

We will now construct the premeasure on $\mathcal{M}$ and the measure on $\sigma(\mathcal{M})$.

A.3 Construction of the Measure

We now begin to construct the measure P on $\sigma(\mathcal{E}) = \sigma(\mathcal{M})$. Let P be defined on $\mathcal{E}$ by (5), i.e., $P_0 : \mathcal{E} \to [0,1]$ is defined by $P_0(K(\alpha)) = p(\alpha)$. We have $P_0(\Omega) = 1$. We extend P_0 to $\mathcal{K}$, i.e., $P_1 : \mathcal{K} \to [0,1]$ by $P_1(A) = P_0(A)$ for $A \in \mathcal{E}$ and $P_1(\varnothing) = 0$. P_1 is well-defined since $\varnothing \notin \mathcal{E}$.

In order to extend P_1 to $\mathcal{M}$ we show the finite additivity of P_1. Obviously, it suffices to show the finite additivity of P_0. We need a few preliminaries. For each two finite runs α and β of π, we have

$$K(\alpha) = K(\beta) \quad \Rightarrow \quad p(\alpha) = p(\beta), \tag{16}$$

since α and β can only differ in non-probabilistic events.

Lemma 4

(i) If $\kappa \sqsubseteq \pi$ is a finite prefix of π and α is a finite run of κ then, there is a $\beta \in \mathfrak{R}_{\max}(\kappa)$ such that $\alpha \sqsubseteq \beta$.

(ii) Let A be a set of runs. We have $\beta \in \mathfrak{R}_{\max}(\sup A) \Rightarrow \inf A \sqsubseteq \beta$.

Proof: (i) Remove in κ all conflicts with α, to obtain β. (ii) We have immediately $\inf A \parallel \beta$; $\inf A \sqsubseteq \beta$ follows then from the maximality of β. □

Let α be a finite run of π. A finite prefix κ of π is said to be α-*complete*, if

$$\bigcup_{\beta \in \mathfrak{R}_{\max}(\kappa)} K(\beta) = K(\alpha). \tag{17}$$

Lemma 5 If κ is α-complete, so is $\inf(\alpha, \kappa)$.

Proof: Since α is a run, so is $\alpha' = \inf(\alpha, \kappa)$. Therefore, we have to show $K(\alpha') = K(\alpha)$. The direction $\supseteq$ is trivial. For $\subseteq$, let $\rho \in K(\alpha')$. From Lemma 4(i) it follows that there is a $\beta \in \mathfrak{R}_{\max}(\kappa)$ such that $\alpha' \sqsubseteq \beta$, hence $\rho \in K(\beta)$ and, by (17), $\rho \in K(\alpha)$. □

Lemma 6 If κ is α-*complete* then

$$p(\alpha) = \sum_{\beta \in \mathfrak{R}_{\max}(\kappa)} p(\beta). \tag{18}$$

Proof: We lead the proof by induction over the prefixes of κ: (a) Let α' be a minimal α-complete branching process. By minimality and Lemma 5, $\alpha' = \inf(\alpha, \alpha')$ and therefore α' is a run and hence $K(\alpha) = K(\alpha')$. The claim follows from (16). (b) Let κ be a non-minimal α-complete branching process and inductively assume (18) for all α-complete proper prefixes of κ. Let κ' be a maximal α-complete proper prefix of κ. Then, there is a maximal set E' of pairwise immediately conflicting events (either a complete probabilistic conflict or a single non-probabilistic event) such that

$$\mathfrak{R}_{\max}(\kappa) = \{\beta \mid \exists \beta' \in \mathfrak{R}_{\max}(\kappa') : \exists e \in E' : \beta' \stackrel{e}{\sqsubset} \beta\}.$$

Then, we have

$$\sum_{\beta \in \mathfrak{R}_{\max}(\kappa)} p(\beta) = \sum_{e \in E'} \mu(e) \cdot \sum_{\beta' \in \mathfrak{R}_{\max}(\kappa')} p(\beta') = 1 \cdot p(\alpha) = p(\alpha).$$ □

We prove now the finite additivity of P_0.

Lemma 7 P_0 is finitely additive, i.e., if A is a finite set of pairwise incompatible finite runs of π such that

$$\bigcup_{\alpha \in A} K(\alpha) = K(\gamma), \tag{19}$$

then, we have

$$\sum_{\alpha \in A} p(\alpha) = p(\gamma). \tag{20}$$

Proof: We prove some helpful propositions first.

1. The finite branching process $\sup A$ is γ-complete. For $\subseteq$ in (17), let $\beta \in \mathfrak{R}_{\max}(\sup A)$ and $\rho \in K(\beta)$. Assume $\rho \notin K(\gamma)$. Then, $\rho \notin K(\alpha)$ for all $\alpha \in A$, by (19), and therefore $\rho \nparallel \alpha$, hence $\rho \nparallel \inf A$, a contradiction to $\inf A \sqsubseteq \beta \sqsubseteq \rho$, which follows from Lemma 4(ii). For $\supseteq$ in (17), let $\rho \in K(\gamma)$. Then, there is, due to (19), an $\alpha \in A$ such that $\rho \in K(\alpha)$. Due to Lemma 4(i) there is a $\beta \in \mathfrak{R}_{\max}(\sup A)$ such that $\rho \in K(\beta)$.

2. For each $\alpha \in A$, there is an α-complete prefix κ_α such that $\kappa_\alpha \sqsubseteq \sup A$. (Due to 1. and $K(\alpha) \subseteq K(\gamma)$; remove in $\sup A$ all conflicts with α to obtain κ_α, i.e., $\kappa_\alpha = \sup \{\rho \in \mathfrak{R}_{\max}(\sup A) \mid \rho \parallel \alpha\}$).

3. For different $\alpha_1, \alpha_2 \in A$, we have $\mathfrak{R}_{\max}(\kappa_{\alpha_1}) \cap \mathfrak{R}_{\max}(\kappa_{\alpha_2}) = \varnothing$. *Proof:* $\mathfrak{R}_{\max}(\kappa_{\alpha_1}) \cap \mathfrak{R}_{\max}(\kappa_{\alpha_2}) \neq \varnothing$ implies $\bigcup_{\beta \in \mathfrak{R}_{\max}(\kappa_{\alpha_1})} K(\beta) \cap \bigcup_{\beta \in \mathfrak{R}_{\max}(\kappa_{\alpha_2})} K(\beta) \neq \varnothing$. From this follows $K(\alpha_1) \cap K(\alpha_2) \neq \varnothing$, a contradiction to $\alpha_1 \nparallel \alpha_2$.

4. We have $\bigcup_{\alpha \in A} \mathfrak{R}_{\max}(\kappa_\alpha) = \mathfrak{R}_{\max}(\sup A)$. *Proof:* From the definition of κ_α follows $\mathfrak{R}_{\max}(\kappa_\alpha) \subseteq \mathfrak{R}_{\max}(\sup A)$. Let $\beta \in \mathfrak{R}_{\max}(\sup A)$. Because of $K(\beta) \subseteq K(\gamma)$ and (19), there is an $\alpha \in A$ such that $K(\beta) \subseteq K(\alpha)$. Then, we have $\beta \in \mathfrak{R}_{\max}(\kappa_\alpha)$ by (17).

We show (20) now.

$$
\begin{aligned}
&\sum_{\alpha \in A} p(\alpha) \\
&= \quad \{\text{2. and (18)}\} \\
&\sum_{\alpha \in A} \sum_{\beta \in \mathfrak{R}_{\max}(\kappa_\alpha)} p(\beta) \\
&= \quad \{\text{3.}\} \\
&\sum_{\beta \in \bigcup_{\alpha \in A} \mathfrak{R}_{\max}(\kappa_\alpha)} p(\beta) \\
&= \quad \{\text{4.}\} \\
&\sum_{\beta \in \mathfrak{R}_{\max}(\sup A)} p(\beta) \\
&= \quad \{\text{1. and (18)}\} \\
&p(\gamma)
\end{aligned}
$$

□

Lemma 8 There is a unique content P_2 on $\mathcal{M}$ such that $P_2(K(\alpha)) = p(\alpha)$.

Proof: We set $P_2(\bigcup_{A \in \mathcal{F}} A) = \sum_{A \in \mathcal{F}} P_1(A)$ for all finite and pairwise disjoint families $\mathcal{F} \subseteq \mathcal{K}$. P_2 is due to Lemma 7 well-defined. Furthermore, every content is finitely additive and P_2 hence unique.

We are now ready to prove Theorem 1.

Proof: (*of Theorem 1*) The σ-additivity of P_2 follows from its finite additivity since no cone can be partitioned into infinitely many cones because π is finitely branching. Therefore, P_2 is a premeasure. Obviously, P_2 is finite and has therefore a unique extension P to $\sigma(\mathcal{M}) = \sigma(\mathcal{E}) = \mathcal{A}$ due to the extension theorem. P satisfies (13) and is hence a probability measure. □

B Measurability of Temporal-Logical Properties

Let H be a countable set of *history formulas* and $\models \subseteq \mathfrak{R}_{\mathrm{fin}}(\pi) \times H$ a satisfaction relation. Let ρ be a run of π and α be a finite prefix of ρ. If φ is a history formula then φ is also a *temporal formula*, and we define $\rho, \alpha \models \varphi$ if $\alpha \models \varphi$. If Φ is a temporal formula, so is $\Diamond\Phi$, and we define $\rho, \alpha \models \Diamond\Phi$ if there is a finite β such that $\alpha \sqsubseteq \beta \sqsubseteq \rho$ and $\rho, \beta \models \Phi$.

If Φ and Ψ are temporal formulas, so are $\neg\Phi$ and $\Phi \vee \Psi$ with the usual meaning. Φ is *satisfied* in ρ if $\rho, \alpha_0 \models \Phi$ where α_0 denotes the eventless prefix of ρ.

Proof: (of Proposition 1) Let Σ be a randomized net system and π a probabilistic branching process of Σ with probability space $(\Omega, \mathcal{A}, P)$. Let Φ be a temporal formula. Then we have, for all finite runs α of π, that the set $\{\rho \in \Omega \mid \alpha \sqsubseteq \rho \text{ and } \rho, \alpha \models \Phi\}$ is measurable in π. We prove this by induction over the structure of temporal formulas:

1. If $\Phi = \varphi$ is a history formula then $\{\rho \in \Omega \mid \alpha \sqsubseteq \rho \wedge \rho, \alpha \models \varphi\}$ equals $K(\alpha)$ if $\alpha \models \varphi$ and equals $\varnothing$ otherwise, hence it is measurable in π.
2. Assume $\{\rho \in \Omega \mid \alpha \sqsubseteq \rho \wedge \rho, \alpha \models \Phi\}$ is measurable for all α. Then $\{\rho \in \Omega \mid \alpha \sqsubseteq \rho \wedge \rho, \alpha \models \Diamond\Phi\} = \bigcup_{\alpha \sqsubseteq \beta} \{\rho \in \Omega \mid \beta \sqsubseteq \rho \wedge \rho, \beta \models \Phi\}$ is measurable, because there are at most countably many finite runs of π.
3. Boolean combinations are measurable according to the definition of a σ-field.

C Comparison of Distributed and Classical Adversary

We prove Theorem 2 in this section. We start with some more notions from elementary measure theory. Let $\mathcal{A}_i$ be a σ-field over Ω_i for $i = 1, 2$ and let $f : \Omega_1 \to \Omega_2$ be a mapping; f is *measurable* (with respect to $\mathcal{A}_1$ and $\mathcal{A}_2$) if for all $A \in \mathcal{A}_2$, we have

$$f^{-1}(A) \in \mathcal{A}_1. \tag{21}$$

Let $\mathcal{E}_2$ be a generator of $\mathcal{A}_2$. The mapping f is measurable if and only if (21) holds for all $A \in \mathcal{E}_2$. If f is measurable then every measure P_1 on $\mathcal{A}_1$ defines a measure P_2 on $\mathcal{A}_2$ by

$$P_2(A) = P_1(f^{-1}(A)), \tag{22}$$

for all $A \in \mathcal{A}_2$.

Let π be a probabilistic branching process with events E_π. A *schedule* for π is a mapping $s : \mathfrak{R}_{\text{fin}}(\pi) \to 2^{E_\pi}$ such that

i. $e \in s(\alpha) \Rightarrow \exists \alpha' : \alpha \overset{e}{\sqsubset} \alpha'$

ii. $s(\alpha) = \varnothing \Rightarrow \alpha$ is maximal

iii. $|s(\alpha)| > 1 \Rightarrow s(\alpha)$ is a complete probabilistic conflict

For a schedule s of π, we define a probabilistic computation tree $s(\pi)$ as follows. The vertices of the tree are sequences over $\mathfrak{R}_{\text{fin}}(\pi)$, which are labelled by its final markings, the root of the tree is the eventless prefix of π, which is labelled by the initial marking and there is an edge from $\tau\alpha$ to $\tau\alpha\alpha'$ labelled by transition t if there is an event $e \in s(\alpha)$ with $\alpha \overset{e}{\sqsubset} \alpha'$ such that e represents the occurrence of t. A schedule s is *progressive* if every run in $s(\pi)$ converges to a maximal non-sequential run of π, i.e., it is represented by an interleaving of a maximal non-sequential run of π. For each π, there is a progressive schedule (we obtain a progressive schedule if we always schedule events with a minimal number of causal predecessors). A progressive schedule s defines, for each maximal run ρ of π, a sequentialization $s(\rho) = \alpha_0, s(\alpha_0) \cap E_\rho, \alpha_1, \ldots$ of ρ, where E_ρ denotes the set of events of ρ.

Lemma 9 If s is a progressive schedule then $s : \Omega_\pi \to \Omega_{s(\pi)}$ is measurable.

Proof: We have to show that for each finite run τ of the computation tree $s(\pi)$, we have $s^{-1}(K(\tau)) \in \mathcal{A}_\pi$. First we observe that the mapping $s : \Omega_\pi \to \Omega_{s(\pi)}$ is injective and $s^{-1}(\tau)$ is the non-sequential run that is obtained by removing the non-causal order from τ. Then, we have $s^{-1}(K(\tau)) \subseteq K(s^{-1}(\tau))$. Since s is progressive, we also have $s^{-1}(K(\tau)) \supseteq K(s^{-1}(\tau))$. Hence $s^{-1}(K(\tau)) = K(s^{-1}(\tau)) \in \mathcal{A}_\pi$. □

It follows that

$$P'_{s(\pi)}(E) = P_\pi(s^{-1}(E)) \tag{23}$$

defines a probability measure on $\Omega_{s(\pi)}$. $P'_{s(\pi)}$ coincides with $P_{s(\pi)}$ on all cones:

$$P_{s(\pi)}(K(\tau)) = p_{s(\pi)}(\tau) = p_\pi(s^{-1}(\tau)) = P_\pi(K(s^{-1}(\tau)) = P_\pi(s^{-1}(K(\tau))) = P'_{s(\pi)}(K(\tau))$$

and hence $P'_{s(\pi)}$ coincides with $P_{s(\pi)}$ everywhere. Consequently, we have

$$P_{s(\pi)}(E) = P_\pi(s^{-1}(E)) \tag{24}$$

for each measurable sequential property E.

Proof: (of Theorem 2) Let π be an admissible probabilistic branching process and s be a progressive schedule of π. Then $s(\pi)$ is an admissible probabilistic computation tree. If $\mathrm{seq}(E)$ is measurable in $s(\pi)$ then $E = s^{-1}(\mathrm{seq}(E))$ is measurable by Lemma 9. We have

$\Sigma \overset{p}{\vdash} \mathrm{seq}(E)$ implies

$P_{s(\pi)}(\mathrm{seq}(E)) \geq p$, which implies

$P_{\pi}(s^{-1}(\mathrm{seq}(E))) \geq p$ by (24), which implies

$P_{\pi}(E) \geq p$. □