Probabilistic Petri Nets and Mazurkiewicz Equivalence

Daniele Varacca[1,2] and Mogens Nielsen[1]

[1] BRICS*- *University of Aarhus, Denmark*
[2] *Computer Laboratory - University of Cambridge, UK*

Abstract. We propose a novel notion of probabilistic net systems. We define their semantics in terms of probabilistic languages. We extend the notion of Mazurkiewicz equivalence to probabilistic languages. We show a preliminary set of properties.

1 Introduction

In the theory of stochastic Petri nets, probabilistic behaviour is typically related with timed behaviour [Bal00]. Enabled transitions fire after a time interval determined by some probability distribution. Different policies can be chosen so that every conflict is also resolved probabilistically. Nevertheless there are situations where we need to account for the probabilistic behaviour, but we do not need to keep track of the time. Recently, Petri nets have been shown to be of use in the theory of security [CW01]. The protocols considered there exploit the notion of nonce, which represents a randomly generated number, but they do not involve time.

Among the various models for concurrent systems, Petri nets feature a neat separation between the notions of nondeterminism and concurrency. The idea of this paper is to exploit this feature. We propose a model where choice is always resolved probabilistically, while independent transitions can fire in any order.

First, following ideas of [dAHJ01], we define the notion of probabilistic word, as a probability distribution over strings of the same length. A probabilistic language is then a set of probabilistic words. We extend the notion of Mazurkiewicz equivalence to probabilistic words. Then we give a definition of probabilistic net systems. To the best of our knowledge, no similar definition appears in the literature. Using the notion of scheduler we define the notion of a run of a probabilistic net system as a probabilistic word over the alphabet of transitions.

The main result of the paper is that every two runs of a confusion-free probabilistic net system have Mazurkiewicz equivalent extensions. As a corollary we obtain the known result that every two finite runs of a deterministic net have extensions which are Mazurkiewicz equivalent.

On the one hand, this shows that our setting is a good lifting of the theory of Mazurkiewicz traces to the probabilistic framework. On the other hand this could eventually lead to partial order techniques for probabilistic verification.

* Basic Research in Computer Science (`www.brics.dk`),
funded by the Danish National Research Foundation.

2 Probabilistic languages

When $f : X \to \mathbb{R}^+$ is a function, for every $Y \subseteq X$, we define $f[Y] := \sum_{x\in Y} f(x)$.

Definition 1. *Let S be an alphabet. Elements of S are called* symbols*, elements of S^* are called* strings*. A* (probabilistic) letter *is a probability distribution over symbols, i.e. a function $\alpha : S \to [0,1]$ such that $\alpha[S] = 1$. A* (probabilistic) word *of* length n *is a probability distribution over strings of length n.*

We find it useful to introduce the following notation. We denote a letter α also as $(\alpha(a))_{a\in W}$ where W is the support of α, i.e. the set $\{a \in S \mid \alpha(a) > 0\}$. We denote a word ν of length n also as $(\nu(\sigma))_{\sigma\in W}$ where W is the support of ν. As usual we identify letters and words of length 1. To ease the reading we will write $(\nu(\sigma))_W$, or $(\nu)_W$ for $(\nu(\sigma))_{\sigma\in W}$, when this does not create confusion.

This notation allows us to define a notion of composition between words.

Definition 2. *Let $(\nu(\sigma))_W$ be a word, let $(\xi_\sigma(\tau))^W_{Z_\sigma}$ be a family of words, indexed by W. The composition of these two objects, denoted by $\nu; (\xi_\sigma)^W$ is the word ζ, such that $\zeta(\sigma;\tau) = \nu(\sigma)\xi_\sigma(\tau)$, where by $\sigma;\tau$ we mean the usual composition of strings.*

It is easy to check that the operation is well defined. If $\nu, (\xi_\sigma)^W$ are words, then $\nu; (\xi_\sigma)^W$ is a word. We write $\nu; \xi$ to mean $\nu; (\xi_\sigma)^W$ when for every $\sigma \in W$, $\xi_\sigma = \xi$. The neutral element on the left is the only word of length 0, which we call ϵ, as the only string in its support. The neutral elements on the right are the families with all elements equal to ϵ.

Definition 3. *A word ν is* prefix *of a word ν' if there exists a family $(\xi_\sigma)^W$ such that $\nu; (\xi_\sigma)^W = \nu'$. We denote this by $\nu \sqsubseteq \nu'$. A* language *is a set of words. A language L is* prefix-closed *iff $\nu' \in L$ and $\nu \sqsubseteq \nu'$ implies that $\nu \in L$.*

We are now going to extend the notion of Mazurkiewicz equivalence to probabilistic words. Let $\bowtie$ be an irreflexive and symmetric relation on S. The pair $(S, \bowtie)$ is called a *concurrent alphabet.*

Definition 4. *[Maz86] Let $(S, \bowtie)$ be a concurrent alphabet. The* Mazurkiewicz equivalence $\equiv$ *is the least congruence on the monoid of strings S^* such that*

$$a \bowtie b \Longrightarrow ab \equiv ba \ .$$

We propose two possible ways of extending this notion to words. The first definition is a standard extension of a relation to a probabilistic framework. However, we also need a stronger notion of equivalence to carry out the proof of our main result.

Definition 5. *Let $\nu = (\nu(\sigma))_W$ and $\xi = (\xi(\tau))_Z$ be two words of the same length over a concurrent alphabet $(S, \bowtie)$. We say that they are* Mazurkiewicz equivalent *and write $\nu \equiv_p \xi$, if for every $\equiv$-equivalence class C, $\nu[C] = \xi[C]$. We say that they are* strongly Mazurkiewicz equivalent *and write $\nu \equiv_s \xi$, if there exists a bijection $\phi : W \to Z$ such that for every $\sigma \in W$,*

– $\sigma \equiv \phi(\sigma)$;
– $\nu(\sigma) = \xi(\phi(\sigma))$.

We call such ϕ a *witness* of the equivalence. Clearly $\equiv_s \subseteq \equiv_p$, but not viceversa.

We can relax the conditions on the witness:

Lemma 1. *Given two words $(\nu)_W, (\xi)_Z$, they are strongly Mazurkiewicz equivalent if there exists an injective function $\phi : W \to S^*$ such that for every $\sigma \in W$,*

– $\sigma \equiv \phi(\sigma)$;
– $\nu(\sigma) = \xi(\phi(\sigma))$.

Proof. For every σ in W, $\phi(\sigma)$ is in Z as $\xi(\phi(\sigma)) = \nu(\sigma)$. Moreover ϕ is surjective onto Z: first notice that

$$1 = \sum_{\sigma \in W} \nu(\sigma) = \sum_{\sigma \in W} \xi(\phi(\sigma)) \ .$$

Since ϕ is injective we have

$$\sum_{\sigma \in W} \xi(\phi(\sigma)) = \sum_{\tau \in \phi(W)} \xi(\tau) = 1 \ .$$

So $\phi(W)$ exhausts the support of ξ. □

3 Elementary Net Systems

We quickly stroll through the properties we need for our purposes. For a more complete introduction, see [RT86,Thi86].

An *elementary net* is a tuple (B, E, F), where B is a set of *conditions*, E is a set of *transitions*, E and B are disjoint, F is a subset of $(B \times E) \cup (E \times B)$, the so called *causal dependency relation*. For $x \in B \cup E$ we write ${}^\bullet x$ for $F^{-1}(x)$ and $x^\bullet$ for $F(x)$. A *case* is a subset $C \subseteq B$. A transition $e \in E$ is *enabled* at a case C if ${}^\bullet e \subseteq C$ and $e^\bullet \cap C = \emptyset$. We say that $C[e\rangle C'$ if e is enabled at C and $C' = (C \setminus {}^\bullet e) \cup e^\bullet$. For $\sigma \in E^n$ we write $C[\sigma\rangle C'$ if there exist $C_0, \ldots, C_n$ such that:

$$C = C_0[\sigma_1\rangle C_1[\sigma_2\rangle C_2 \ldots [\sigma_n\rangle C_n = C' \ .$$

An *elementary net system* is a tuple $\mathcal{N} = (B, E, F, C_0)$ where (B, E, F) is an elementary net and C_0 is the *initial case*. A case C' is *reachable* in the net system $\mathcal{N}$ if there exists $\sigma \in E^*$ such that $C_0[\sigma\rangle C'$. A string $\sigma \in E^*$ is a *run* of the system if there exists C such that $C_0[\sigma\rangle C$.

Two transitions e_0, e_1 are *independent* iff $e_0 \neq e_1$ and $({}^\bullet e_0 \cup e_0^\bullet) \cap ({}^\bullet e_1 \cup e_1^\bullet) = \emptyset$. (The first condition insures that, when ${}^\bullet e = e^\bullet = \emptyset$, then e is not independent from itself.) The independence relation I is irreflexive and symmetric. We say that two transitions are in conflict at case C if they are enabled at C and not independent. For every case C the conflict relation is reflexive and symmetric.

For every e and C the set of transitions in conflict with e at C is called the *conflict set* of e at C.

We can view (E, I) as a concurrent alphabet and define Mazurkiewicz equivalence on E^*. The following result is fundamental in the theory of Mazurkiewicz traces (cf. Theorem 8 in [Maz86]):

Theorem 1. *If $\sigma, \sigma \in E^*$ and $\sigma \equiv \sigma'$ then $C[\sigma\rangle C'$ if and only if $C[\sigma'\rangle C'$.*

Definition 6. *A case C is* confused *if there are two independent enabled transitions e_0, e_1 such that, if $C[e_0\rangle C'$, then the conflict sets of e_1 at C and at C' are different. A net systems is* confusion-free *if no reachable case is confused.*

Definition 7. *A net system is called* neatly concurrent *if for every reachable case the conflict relation is an equivalence. The equivalence classes are called* actions*.*

Proposition 1. *A confusion-free net system is neatly concurrent.*

Proof. Take a net system $\mathcal{N}$ and consider a reachable case C for which the conflict relation is not an equivalence. Such relation is always reflexive and symmetric, therefore it must be non-transitive. This means that there are three enabled transitions e, e', e'' such that e is in conflict with e', e' is in conflict with e'' but e, e'' are independent. The firing of e'' disables e', thus modifying the conflict set of e: hence C is confused. $\square$

The converse of the theorem is not true: we will see later an example of a net system which is neatly concurrent, but confused.

The following lemmas single out one important feature of confusion free systems.

Lemma 2. *Let $\mathcal{N}$ be a confusion-free net system, let C be a reachable case of $\mathcal{N}$, and let Z be the set of actions at C. Take $X \in Z$ and $e \notin X$. Let C' be the case such that $C[e\rangle C'$ Let Z' be the set of actions at C'. Then $X \in Z'$.*

Proof. Take $X \in Z$ and consider $e \notin X, e' \in X$. Since e, e' belong to two different actions, they must be independent. Since the system is confusion free, the firing of e does not modify the conflict set of e'. So at C' the transition e' is still enabled, its conflict set is X, and since the system is neatly concurrent, X is still an equivalence class. $\square$

Lemma 3. *Let $\mathcal{N}$ be a confusion-free net system, let C be a reachable case of $\mathcal{N}$ and let Z be the set of actions at C. Take $X \in Z$. Suppose $C[e_0\rangle C_0 \dots [e_n\rangle C_n$ where for every i, $e_i \notin X$. Let Z_i be the set of actions at C_i. Then for every i, $X \in Z_i$.*

Proof. By induction on n, using the previous lemma. $\square$

Another useful observation is relating deterministic systems with confusion-free systems.

Definition 8. *A net system $\mathcal{N}$ is called* deterministic *if for every reachable case C and for every transition e enabled at C, the conflict set of e at C is $\{e\}$.(See also [Thi86].)*

Then it is trivial to observe

Proposition 2. *A deterministic net system is confusion-free.*

4 Probabilistic Nets

Definition 9. *A* weighted *net is a tuple (B, E, F, W) where (B, E, F) is an elementary net and W is a weight function $E \to \mathbb{R}^+$. A weighted net system is a weighted net together with an initial case C_0, such that (B, E, F, C_0) is a neatly concurrent net system and for every reachable case C and for every action X at C, $W[X] < \infty$.*

We can give a semantics of a neatly concurrent weighted net in terms of Markov Decision Processes [Put94]. For every action X, the probability of a transition $e \in X$ is obtained normalizing the weights against all the transitions in X. So every action determines the probabilities of its transitions. However, note that in general transitions are not assigned fixed probabilities, as the probability of a transition depends on the action. In what follows we will not allow transitions with weight 0. So the probability distribution which corresponds to an action X assigns non-zero probability to every transition in X.

A similar semantics could be given in terms of Segala's probabilistic automata [SL95]. However, for the purpose of this work, we do not need to do this formally. We made this observation mainly to justify our terminology (actions, schedulers, etc...).

Definition 10. *Take a weighted net system $\mathcal{N}$, one of its reachable cases C, and an action X at C. The probabilistic letter of X, ν_X, is defined as follows*

$$\nu_X(e) = \begin{cases} \frac{W(e)}{W[X]} & \textit{if } e \in X \\ 0 & \textit{otherwise} \end{cases} .$$

In the following we identify an action X at C with its probabilistic letter. That is we will write X for ν_X, and $X(e)$ will denote a probability. Also we make the convention that σ will denote the case C such that $C_0[\sigma\rangle C$. If a string σ is such that $\not\exists C. C_0[\sigma\rangle C$, then σ denotes the empty case.

For purely technical reasons we find it convenient to consider nets with a special transition ∂ symbolizing a delay event. The transition ∂ is such that ${}^\bullet\partial = \partial^\bullet = \emptyset$. Thus ∂ is always enabled, it is independent of any other transition and its firing does not change the case. The set $\{\partial\}$ is an action at every case. The introduction of the delay event is convenient if we want to consider only probability distributions over strings of the same length. Otherwise it would be difficult to account for scheduling policies that decide to stop the computation, or to account for situations when one branch of a probabilistic computation cannot progress, while another branch can.

Definition 11. *Let* $E_\partial = E \uplus \{\partial\}$. *If* $\mathcal{N} = (B, E, F, C_0)$ *is a net system, then* $\mathcal{N}_\partial$ *is defined as* (B, E_∂, F, C_0).

The system $\mathcal{N}$ is neatly concurrent iff $\mathcal{N}_\partial$ is; $\mathcal{N}$ is confusion free iff $\mathcal{N}_\partial$ is.

In what follows, $P^+(X)$ denotes the set of nonempty subsets of a set X.

Definition 12. *A* scheduler *of a neatly concurrent weighted net system* $\mathcal{N}_\partial$ *is a function* $f : (E_\partial)^* \to P^+(E_\partial)$ *such that* $f(\sigma)$ *is an action at* σ. *A* partial scheduler f_n *of length* n *is the restriction of a scheduler* f *to strings of length* $< n$. *A* fragment *scheduler* $f_{n,m}$ *is the restriction of a scheduler* f *strings of length* l *such that* $n \leq l < m$. *The number* $m - n$ *is the length of the fragment scheduler. A fragment scheduler* f' extends *a partial scheduler* f, *if* $f \uplus f'$ *is a partial scheduler. Using the notation above* $f_{n',m}$ *extends* f_n *if and only if* $n = n'$. *In that case we write* $f; f'$ *for* $f \uplus f'$. *The length of a scheduler* f *is denoted by* $|f|$.

Note that our schedulers take strings as input. This means that they decide knowing not only the case, but the whole history of a process.

Definition 13. *The* language $L_f = \{\nu_f^n \mid n \in N\}$ *of a scheduler* f *is defined as follows:*

$$\nu_f^0(\epsilon) = 1 \ ;$$

$$\nu_f^{n+1}(\sigma; e) = \nu_f^n(\sigma) \cdot f(\sigma)(e) \ .$$

Similarly we define the language $L_{f_m} = \{\nu_{f_m}^n \mid n \leq m\}$ *of a partial scheduler* f_m.

A scheduler f *is* reduced *if for any string* σ *of length* n,

$$\nu_f^n(\sigma) = 0 \Longrightarrow f(\sigma) = \{\partial\} \ .$$

The word *of a partial scheduler* f_m *is the only word of length* m *in* L_{f_m}.

It is easy to show that L_f is indeed a set of words, that is for every n, ν_f^n is a probabilistic word of length n. Moreover if $g = f; f'$ where $|f| = n, |g| = m$, then $\nu_f^n \sqsubseteq \nu_g^m$.

In the following we will identify a partial scheduler with the word it generates. When we want to stress that a word f is a probability distribution over strings, we will use the expression ν_f. We will also identify a word with its support. A type discipline will allow the reader to disentangle this overloading.

– When we think of f as a scheduler, we can apply it to strings and get actions. Thus $f(\sigma)$ is an action. Also actions can be interpreted as either sets of transitions or letters. So if we write $e \in f(\sigma)$, we interpret f as a scheduler and $f(\sigma)$ as a set of transitions. If we write $f(\sigma)(e)$, we interpret f as a scheduler and $f(\sigma)$ as letter, so that $f(\sigma)(e)$ is a probability. Since we identify letters as words of length 1, and we identify words with their support, the expression $e \in f(\sigma)$ could be interpreted as "e is in the support of $f(\sigma)$". This possible confusion is avoided by our convention that all the transitions in an action get non-zero probability. This being the case, e is in the support of $f(\sigma)$ if and only if $e \in f(\sigma)$.

– When we think of f as a word, we write $\sigma \in f$ meaning "σ is in the support of f". Another ambiguity could arise in the expression $f(\sigma)$, as this could also represent a probability. We stipulate that the expression $f(\sigma)$ always represents an action.
– When we think of f as a word, and we want to use it as a probability distribution, we will use the expression ν_f.

Proposition 3. *For every scheduler f there exists a reduced scheduler f' such that $L_f = L_{f'}$.*

Definition 14. *A* partial run *of a weighted net system $\mathcal{N}$ under a partial scheduler f_n is the word of f_n.*

Definition 15. *The* language *of a weighted net systems is the set of its partial runs.*

The first observation is a straightforward generalization of Theorem 1

Definition 16. *Let f be a fragment scheduler of a weighted net system $\mathcal{N}$ with initial state C_0. For every case C of $\mathcal{N}$ we define*

$$d_f(C) := \sum_{C_0[\sigma\rangle C} \nu_f(\sigma) \ .$$

Then the following proposition is an easy corollary of the definition of $\equiv_s$ and of Theorem 1.

Proposition 4. *Let $\mathcal{N}$ be a weighted net system and let f, g be two partial schedulers in its language. If $f \equiv_s g$ then $d_f(C) = d_g(C)$ for every reachable case C.*

Notice that Proposition 4 is somewhat weaker than Theorem 1, in that if f is a scheduler, not all words in the equivalence class of the word of f correspond to a scheduler.

Consider now the neatly concurrent weighted net system represented in Figure 1 using the standard graphical representation of nets. In the initial case there are two possible actions: $\{a\}$ and $\{c\}$. If the first is chosen, then a is fired with probability 1. This leads to a case where only the action $\{b, c\}$ is enabled. This scheduling policy generates the word that assigns probability $\frac{1}{2}$ to both ab and ac.

If in the initial case the second action is chosen, then c is fired with probability 1, leading to a case where only the action $\{a\}$ is enabled. This scheduling policy generates the word which assigns probability 1 to ca. These two words are not Mazurkiewicz equivalent, nor can they be extended to two Mazurkiewicz equivalent words. Notice that the above system is not confusion-free. The main result of our paper states that a confusion free net system has only one probabilistic run, up to Mazurkiewicz equivalence.

Theorem 2. *Let L be the language of a confusion-free weighted net system. Then for all partial schedulers $f, g \in L$ there exist two schedulers $f', g' \in L$ such that $f \sqsubseteq f'$, $g \sqsubseteq g'$ and $f' \equiv_s g'$.*

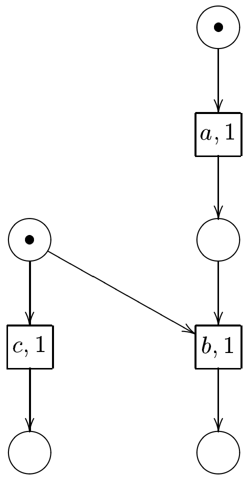

Fig. 1.

Let us see one example. In the system represented in Figure 2 we can first schedule the action $\{a\}$ and we get the word f assigning probability 1 to the string a. Or we can schedule the action $\{b, c\}$ and get the word g which assigns probability $\frac{1}{2}$ to both b, c. Then we can extend f to f' with $\nu_{f'}(ab) = \nu_{f'}(ac) = \frac{1}{2}$. Also we can extend g to g' with $\nu_{g'}(ba) = \nu_{g'}(ca) = \frac{1}{2}$. Clearly $f' \equiv_s g'$, the witness being $\phi(ab) = ba, \phi(ac) = ca$. We can now extend f' to f'' with $\nu_{f''}(abd) = \nu_{f''}(ac\partial) = \frac{1}{2}$, and extend g' to g'' with $\nu_{g''}(bad) = \nu_{g''}(ca\partial) = \frac{1}{2}$. Again $f'' \equiv_s g''$.

The proof of Theorem 2 proceeds by induction on the sum of the length of f and g. The basis is obvious as there is only one word of length 0. The inductive step requires some lemmas.

Lemma 4. *If two partial schedulers f, g are strongly Mazurkiewicz equivalent then for every fragment scheduler $\tilde{f}$ extending f there exists a fragment scheduler $\tilde{g}$ extending g such that $f; \tilde{f} \equiv_s g; \tilde{g}$. If $f; \tilde{f}$ and g are reduced, the $\tilde{g}$ can be chosen such that $g; \tilde{g}$ is also reduced.*

Proof. Clearly it is enough to prove the statement for fragment schedulers of length 1. Let n be the length of f, g. Let W_f, W_g be the supports of (the words of) f, g. Let $\phi : W_f \to W_g$ be a witness for $f \equiv_s g$. Let $\tilde{f} : E_\partial^n \to P^+(E_\partial)$ be a fragment scheduler extending f. We want to build a fragment scheduler $\tilde{g}$. For $\tau \notin W_g$ we set $\tilde{g}(\tau) = \{\partial\}$. Otherwise we set $\tilde{g}(\tau) = \tilde{f}(\phi^{-1}(\tau))$. In order to check that $\tilde{g}$ extends g we have just to check that $\tilde{g}(\tau)$ is an action at τ. Since ϕ is a witness for $f \equiv_s g$, then $\tau \equiv \phi^{-1}(\tau)$. Therefore the two strings represent the same case. Since $\tilde{f}$ extends f, then $\tilde{f}(\phi^{-1}(\tau))$ is an action at $\phi^{-1}(\tau)$ which is the same case as τ.

To prove the equivalence, define $\phi' : W_{f;\tilde{f}} \to W_{g;\tilde{g}}$ as follows:

$$\phi'(\sigma; e) = \phi(\sigma); e \ .$$

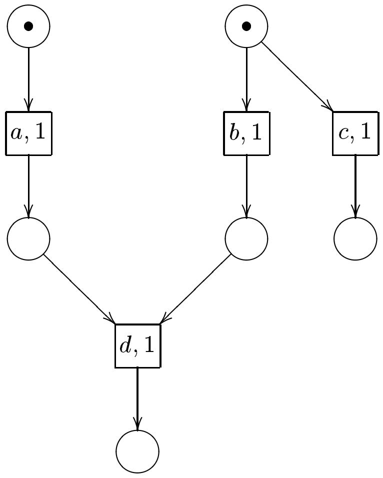

Fig. 2.

The function ϕ' is injective, as ϕ is. The corresponding strings are equivalent, as $\equiv$ is a congruence. As for the probability:

$$\nu_{f;\tilde{f}}(\sigma;e) = \nu_f(\sigma)\tilde{f}(\sigma)(e) = \nu_f(\sigma)\tilde{g}(\phi(\sigma))(e)$$
$$= \nu_g(\phi(\sigma))\tilde{g}(\phi(\sigma))(e) = \nu_{g;\tilde{g}}(\phi(\sigma);e) \ .$$

Lemma 1 gives us the result. □

To prove the theorem let f, g be schedulers with $n := |f|$ and $m := |g|$. Let χ be a fragment scheduler extending f, with $|\chi| = 1$.

By induction hypothesis there exist two fragment schedulers $\tilde{f}, \tilde{g}$ such that $f;\tilde{f} \equiv_s g;\tilde{g}$. Let k be the length of $f;\tilde{f}$.

We are going to build two fragment schedulers s, t such that $f;\chi;s \equiv_s f;\tilde{f};t$. Then, by Lemma 4 we get t' such that $f;\tilde{f};t \equiv_s g;\tilde{g};t'$, and we are done.

First we build a fragment scheduler t of length 1 that extends $f;\tilde{f}$. We then build s by induction. We build s_{n+2} so that it extends $f;\chi$, then for $2 \leq i \leq k-n$ we build s_{n+i+1} so that it extends $f;\chi;s_{n+2};\ldots;s_{n+i}$. The idea is to parallel the choice of $\tilde{f}$ until $\tilde{f}$ behaves like χ. If $\tilde{f}$ never behaves like χ then we add one more step to ensure that this choice is eventually made.

Together with the definition of s_{n+i} we define an auxiliary function sfp_{n+i} (for shiftpoint), which keeps track of the point where $\tilde{f}$ behaves like χ.

This function takes three kinds of value: integers, $\perp, \infty$. Intuitively:

- $\mathrm{sfp}_{n+i}(\tau := \sigma;\bar{e}e_{n+2}\ldots e_{n+i-1}) = h \in \mathbb{N}$ means that τ is in the support of $f;\chi;s_{n+2};\ldots;s_{n+i-1}$ and that position h is the first time where $\tilde{f}_h(\sigma;e_{n+2}\ldots e_h) = \chi(\sigma)$;
- $\mathrm{sfp}_{n+i}(\tau := \sigma;\bar{e}e_{n+2}\ldots e_{n+i-1}) = \perp$ means that τ is in the support of $f;\chi;s_{n+2};\ldots;s_{n+i-1}$ and that $\tilde{f}_h(\sigma;e_{n+2}\ldots e_h) \neq \chi(\sigma)$, for every $n+2 \leq h \leq n+i-1$;

– $\mathrm{sfp}_{n+i}(\tau) = \infty$ means that τ is not in the support of $f; \chi; s_{n+2}; \ldots; s_{n+i-1}$ (and so no extension of τ is in the support of any extension of $f; \chi; s_{n+2}; \ldots; s_{n+i-1}$).

We now give the formal definitions. We will need the following function on strings: for every $1 \leq i \leq j \leq n$

$$\mathrm{shift}_{i,j}(e_1 \ldots e_i \ldots e_j \ldots e_n) = e_1 \ldots e_{i-1} e_{i+1} \ldots e_j e_i e_{j+1} \ldots e_n \ .$$

Notice that in a context of a concurrent alphabet, if e_i is independent of all symbols $e_{i+1}, \ldots, e_j$ then $\mathrm{shift}_{i,j}(e_1 \ldots e_n) \equiv e_1 \ldots e_n$.

Also let $\tilde{f}_{n+i}$ be $\tilde{f}$ restricted to strings of length $< n+i$.

First we define t:

$$t(\sigma; e'_{n+1} e'_{n+2} \ldots e'_k)$$

$$= \begin{cases} \text{any fixed enabled action if } \begin{cases} \sigma; e'_{n+1} \ldots e'_k \in f; \tilde{f} \\ \exists l.\ e'_l \in \chi(\sigma) \end{cases} \\ \\ \chi(\sigma) \qquad\qquad\qquad\qquad\quad \text{if } \begin{cases} \sigma; e'_{n+1} \ldots e'_k \in f; \tilde{f} \\ \nexists l.\ e'_l \in \chi(\sigma) \end{cases} \\ \\ \{\partial\} \qquad\qquad\qquad\qquad\qquad \text{if } \sigma; e'_{n+1} e'_{n+2} \ldots e'_k \notin f; \tilde{f} \end{cases}$$

The idea is that if $\exists l.\ e'_l \in \chi(\sigma)$ then for the smallest such l, $\tilde{f}(\sigma; e'_{n+1} e'_{n+2} \ldots e'_{l-1}) = \chi(\sigma)$. If this is not the case, then we force t to make that choice. The fact that choice is still available is a consequence of Lemma 3.

Then we define s together with the auxiliary function sfp. We do this inductively:

$$\mathrm{sfp}_{n+2}(\sigma; \bar{e}) = \begin{cases} n+1 & \text{if } \sigma \in f \ \&\ \bar{e} \in \chi(\sigma) \ \&\ \chi(\sigma) = \tilde{f}_{n+1}(\sigma) \\ \bot & \text{if } \sigma \in f \ \&\ \bar{e} \in \chi(\sigma) \ \&\ \chi(\sigma) \neq \tilde{f}_{n+1}(\sigma) \\ \infty & \text{if } \sigma \notin f \text{ or } \bar{e} \notin \chi(\sigma) \end{cases}$$

$$s_{n+2}(\sigma; \bar{e}) = \begin{cases} \tilde{f}_{n+2}(\sigma; \bar{e}) & \text{if } \ \mathrm{sfp}_{n+2}(\sigma; \bar{e}) = n+1 \\ \tilde{f}_{n+1}(\sigma) & \text{if } \ \mathrm{sfp}_{n+2}(\sigma; \bar{e}) = \bot \\ \{\partial\} & \text{if } \ \mathrm{sfp}_{n+2}(\sigma; \bar{e}) = \infty \end{cases}$$

For $2 \leq i \leq k - n$ we put

$$\mathrm{sfp}_{n+i+1}(\sigma;\bar{e}e_{n+2}\dots e_{n+i}) = \begin{cases} h & \text{if } \mathrm{sfp}_{n+i}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) = h \in \mathbb{N} \\ \infty & \text{if } \mathrm{sfp}_{n+i}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) = \infty \\ n+i & \text{if } \begin{cases} \mathrm{sfp}_{n+i}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) = \bot \\ e_{n+i} \in s_{n+i}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) \\ \chi(\sigma) = \tilde{f}_{n+i}(\sigma;e_{n+2}\dots e_{n+i}) \end{cases} \\ \bot & \text{if } \begin{cases} \mathrm{sfp}_{n+i}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) = \bot \\ e_{n+i} \in s_{n+i}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) \\ \chi(\sigma) \neq \tilde{f}_{n+i}(\sigma;e_{n+2}\dots e_{n+i}) \end{cases} \\ \infty & \text{if } \begin{cases} \mathrm{sfp}_{n+i}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) = \bot \\ e_{n+i} \notin s_{n+i}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) \end{cases} \end{cases}$$

Then for $2 \leq i < k-n$:
$s_{n+i+1}(\sigma;\bar{e}e_{n+2}\dots e_{n+i})$

$$= \begin{cases} \tilde{f}_{n+i+1}(\mathrm{shift}_{n+1,h}(\sigma;\bar{e}e_{n+2}\dots e_{n+i})) & \text{if } \mathrm{sfp}_{n+i+1}(\sigma;\bar{e}e_{n+2}\dots e_{n+i}) = h \in \mathbb{N} \\ \tilde{f}_{n+i}(\sigma;e_{n+2}\dots e_{n+i}) & \text{if } \mathrm{sfp}_{n+i+1}(\sigma;\bar{e}e_{n+2}\dots e_{n+i}) = \bot \\ \{\partial\} & \text{if } \mathrm{sfp}_{n+i+1}(\sigma;\bar{e}e_{n+2}\dots e_{n+i}) = \infty \end{cases}$$

and when $i = k-n$:

$s_{k+1}(\sigma;\bar{e}e_{n+2}\dots e_k)$

$$= \begin{cases} t_{k+1}(\mathrm{shift}_{n+1,h}(\sigma;\bar{e}e_{n+2}\dots e_k)) & \text{if } \mathrm{sfp}_{k+1}(\sigma;\bar{e}e_{n+2}\dots e_k) = h \in \mathbb{N} \\ \tilde{f}_k(\sigma;e_{n+2}\dots e_k) & \text{if } \mathrm{sfp}_{k+1}(\sigma;\bar{e}e_{n+2}\dots e_k) = \bot \\ \{\partial\} & \text{if } \mathrm{sfp}_{k+1}(\sigma;\bar{e}e_{n+2}\dots e_k) = \infty \end{cases}$$

Intuitively, sfp returns the first step where $\tilde{f}$ has chosen $\chi(\sigma)$. If this hasn't happened yet, sfp returns $\bot$ which means "not yet". If the word is outside the support of $f;\tilde{f}$, then sfp returns ∞ which means "I don't care". To define s we check what sfp tells us. If $\tilde{f}$ has already behaved like χ then we parallel its choices (taking care of shifting up the event in $\chi(\sigma)$). Otherwise we parallel the choices of $\tilde{f}$, keeping one step ahead.

Now define

$$s := \bigcup_{2\leq i\leq k-n+1} s_{n+i}$$

Then the main lemma is the following

Lemma 5. *The function t is a fragment scheduler extending $f;\tilde{f}$. The function s is a fragment scheduler extending $f;\chi$.*

For every $e_{k+1} \in s(\sigma;\bar{e}e_{n+2}\dots e_k)$:
if $\mathrm{sfp}_{k+1}(\sigma;\bar{e}e_{n+2}\dots e_k) = h \in \mathbb{N}$, then

– $\sigma;\bar{e}e_{n+2}\dots e_{k+1} \equiv \mathrm{shift}_{n+1,h}(\sigma;\bar{e}e_{n+2}\dots e_{k+1})$;
– $\nu_{f;\chi;s}(\sigma;\bar{e}e_{n+2}\dots e_{k+1}) = \nu_{f;\tilde{f}_k;t_{k+1}}(\mathrm{shift}_{n+1,h}(\sigma;\bar{e}e_{n+2}\dots e_{k+1}))$.

If $\mathrm{sfp}_{k+1}(\sigma;\bar{e}e_{n+2}\dots e_k) = \bot$, *then*

– $\sigma;\bar{e}e_{n+2}\dots e_{k+1} \equiv \mathrm{shift}_{n+1,k+1}(\sigma;\bar{e}e_{n+2}\dots e_{k+1})$;
– $\nu_{f;\chi;s}(\sigma;\bar{e}e_{n+2}\dots e_{k+1}) = \nu_{f;\tilde{f}_k;t_{k+1}}(\mathrm{shift}_{n+1,k+1}(\sigma;\bar{e}e_{n+2}\dots e_{k+1}))$.

If $\mathrm{sfp}_{k+1}(\sigma;\bar{e}e_{n+2}\dots e_k) = \infty$, *then*

– $\nu_{f;\chi;s}(\sigma;\bar{e}e_{n+2}\dots e_{k+1}) = 0$.

The proof of this lemma can be found in the Appendix.

Now to prove the theorem we have to prove that $f;\chi;s \equiv_s f;\tilde{f};t$. We claim that a witness for this is the function ϕ defined as follows:

$$\phi(\sigma;\bar{e}e_{n+2}\dots e_k e_{k+1}) = \begin{cases} \mathrm{shift}_{n+1,h}(\sigma;\bar{e}e_{n+2}\dots e_{k+1}) & \text{if } \mathrm{sfp}_k(\sigma;\bar{e}e_{n+2}\dots e_k) = h \in \mathbb{N} \\ \mathrm{shift}_{n+1,k+1}(\sigma;\bar{e}e_{n+2}\dots e_{k+1}) & \text{if } \mathrm{sfp}_k(\sigma;\bar{e}e_{n+2}\dots e_k) = \bot \end{cases}$$

We have just to argue that ϕ is injective, and then Lemma 5 tells us exactly that the function ϕ is well defined and a witness. Consider two strings $\tau := \sigma;\bar{e}e_{n+2}\dots e_{k+1}$ and $\tau' := \sigma';\bar{e}'e'_{n+2}\dots e'_{k+1}$ in (the support of) f'_{k+1}, and suppose that $\phi(\tau) = \phi(\tau')$. Let's say that $\phi(\tau) = \mathrm{shift}_{n+1,h}(\tau)$ and $\phi(\tau') = \mathrm{shift}_{n+1,h'}(\tau')$. First notice that it must be $\sigma = \sigma'$. Moreover we claim that $h = h'$. Suppose they are different, say $h < h'$. Then $\sigma;e_{n+2}\dots e_h\bar{e}e_{h+1}\dots e_{k+1} = \sigma;e'_{n+2}\dots e'_h e'_{h+1}\dots e_{h'}\bar{e}'e_{h'+1}\dots e'_{k+1}$. This means that $\bar{e} = e'_{h+1}$. By definition of ϕ is $\mathrm{sfp}_k(\sigma;\bar{e}'e'_{n+2}\dots e'_k) = h', \bot$. Either case implies that $\mathrm{sfp}_{h+1}(\sigma;\bar{e}'e'_{n+2}\dots e'_{h+1}) = \bot$ which, as shown by Lemma 6 in the Appendix means that e'_{h+1} is independent of $\bar{e}'$, and that $\bar{e}' \in \chi(\sigma)$. Again, by definition of ϕ is $\mathrm{sfp}_k(\sigma;\bar{e}e_{n+2}\dots e_k) = h$ which implies that $\bar{e} \in \chi(\sigma)$. Therefore $\bar{e}'$ is not independent of $\bar{e} = e'_{h+1}$. Contradiction. Since $h = h'$ we have that $\sigma;e_{n+2}\dots e_h\bar{e}e_{h+1}\dots e_{k+1} = \sigma;e'_{n+2}\dots e'_h\bar{e}'e_{h'+1}\dots e'_{k+1}$. This means that $\bar{e} = \bar{e}'$ and $e_{n+i} = e'_{n+i}$ for every $2 \le i \le k-n+1$. That is $\tau = \tau'$. □

As a corollary we get a similar statement involving the weaker, and more standard, notion of probabilistic Mazurkiewicz equivalence.

Corollary 1. *Let L be the language of a confusion-free weighted net system. Then for all partial schedulers $f, g \in L$ there exist two schedulers $f', g' \in L$ such that $f \sqsubseteq f'$, $g \sqsubseteq g'$ and $f' \equiv_p g'$.*

Using Proposition 2, we derive the following well known theorem.

Corollary 2. *Let $\mathcal{N}$ be a deterministic net system. If σ, σ' are two runs of the system, then there exist two runs τ, τ such that $\sigma \sqsubseteq \tau$, $\sigma' \sqsubseteq \tau'$ and $\tau \equiv \tau'$.*

Proof. Since in a deterministic system the equivalence classes of conflict are singletons, the corresponding probability distributions assign 1 to the only element of the class (regardless of the weights). Probabilistic words reduce then to strings, and probabilistic Mazurkiewicz equivalence reduces to standard Mazurkiewicz equivalence. □

5 Future Work

In this paper we have introduced a new notion of probabilistic net systems, we have extended the notion of Mazurkiewicz equivalence and we have proved a theorem showing the soundness of our definitions. We are now working, together with Glynn Winskel, to extend these idea to the denotational universe. In our forthcoming work we aim to present a notion of probabilistic event structures, and its relation with the probabilistic powerdomain.

Acknowledgement

The first author wishes to thank Jiří Srba for useful discussions, and for providing the macros for the figures.

References

[Bal00] Gianfranco Balbo. Introduction to stochastic petri nets. In *Euro Summer School on Trends in Computer Science*, volume 2090 of *LNCS*, pages 84–155, 2000.

[CW01] Federico Crazzolara and Glynn Winskel. Events in security protocols. In *Proc. of the 8th ACM Conference on Computer and Communication Security, Philadelphia*, November 2001.

[dAHJ01] Luca de Alfaro, Thomas A. Henzinger, and Ranjit Jhala. Compositional methods for probabilistic systems. In *Proc. 12th CONCUR*, 2001.

[JLY01] Bengt Jonsson, Kim Larsen, and Wang Yi. Probabilistic extensions of process algebras. In *Handbook of Process Algebras*. Elsevier, 2001.

[Maz86] Antoni Mazurkiewicz. Trace theory. In *Petri Nets: Applications and Relationships to Other Models of Concurrency*, volume 255 of *LNCS*, pages 279–324. Springer, 1986.

[Put94] Martin L. Puterman. *Markov decision processes : discrete stochastic dynamic programming*. Wiley, New York, 1994.

[RT86] Grzegorz Rozenberg and P.S. Thiagarajan. Petri nets: Basic notions, structure, behaviour. In *Current Trends in Concurrency*, volume 224 of *LNCS*, pages 585–668. Springer, 1986.

[SL95] Roberto Segala and Nancy Lynch. Probabilistic simulations for probabilistic processes. *Nordic Journal of Computing*, 2(2):250–273, 1995. An extended abstract appears in *Proc. 5th CONCUR*, Uppsala, Sweden, LNCS 836, pages 481–496, August 1994.

[Thi86] P.S. Thiagarajan. Elementary net systems. In *Advances in Petri Nets*, volume 254 of *LNCS*, pages 26–59. Springer, 1986.

A Proof of Lemma 5

To prove the lemma we need a complicated induction. First define

$$f'_{n+1} = f;\chi$$
$$f'_{n+i+1} = f'_{n+i} \uplus s_{n+i+1}$$

Lemma 6. *For every $2 \leq i \leq k-n+1$, f'_{n+i-1} is a scheduler and s_{n+i} extends it. Moreover:*
If $\mathrm{sfp}_{n+i}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) = \perp$, *then*

1. $\sigma;\bar{e}e_{n+2}\dots e_{n+i-1} \in f'_{n+i-1}$;
2. $e_{n+2},\dots,e_{n+i-1}$ *are independent of* $\bar{e}$;
3. $\chi(\sigma)$ *is an action at* $\sigma;e_{n+2}\dots e_{n+i-1}$;
4. $\bar{e} \notin \tilde{f}_{n+i-1}(\sigma;e_{n+2}\dots e_{n+i-1})$;
5. $\nu_{f'_{n+i-1}}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) = \nu_{f;\tilde{f}_{n+i-1}}(\sigma;e_{n+2}\dots e_{n+i-1})\cdot\chi(\sigma)(\bar{e})$.

If $\mathrm{sfp}_{n+i}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) = h \in \mathbb{N}$, *then*

1. $\sigma;\bar{e}e_{n+2}\dots e_{n+i-1} \equiv \mathrm{shift}_{n+1,h}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1})$;
2. $\nu_{f'_{n+i-1}}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) = \nu_{f;\tilde{f}_{n+i-1}}(\mathrm{shift}_{n+1,h}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}))$.

If $\mathrm{sfp}_{n+i}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) = \infty$, *then*

– $\nu_{f'_{n+i-1}}(\sigma;\bar{e}e_{n+2}\dots e_{n+i-1}) = 0$;

Proof of Lemma 6: by induction on i.

We first prove the basis, that is $i = 2$. If $\mathrm{sfp}_{n+2}(\sigma;\bar{e}) = \perp$, there are five clauses to prove.

1. by definition of sfp_{n+2}, $\sigma \in f$ and $\bar{e} \in \chi(\sigma)$;
2. this is vacuously true;
3. since χ is a fragment scheduler, then $\chi(\sigma)$ is an action at σ;
4. since $\tilde{f}_{n+1}(\sigma) \neq \chi(\sigma)$ and $\bar{e} \in \chi(\sigma)$ then $\bar{e} \notin \tilde{f}_{n+1}(\sigma)$;
5. $\nu_{f;\chi}(\sigma;\bar{e}) = \nu_f(\sigma)\cdot\chi(\sigma)(\bar{e})$, by definition of the word of a scheduler.

If $\mathrm{sfp}_{n+2}(\sigma;\bar{e}) = h \in \mathbb{N}$, then $h = n+1$, and $\chi(\sigma) = \tilde{f}_{n+1}(\sigma)$. There are two clauses to prove:

1. $\sigma;\bar{e} = \mathrm{shift}_{n+1,n+1}(\sigma;\bar{e})$;
2. by definition $\nu_{f;\chi}(\sigma;\bar{e}) = \nu_f(\sigma)\cdot\chi(\sigma)(\bar{e})$. Since $\chi(\sigma) = \tilde{f}_{n+1}(\sigma)$, then $\nu_f(\sigma)\cdot\chi(\sigma)(\bar{e}) = \nu_f(\sigma)\cdot\tilde{f}_{n+1}(\sigma)(\bar{e})$. And by definition again $\nu_f(\sigma)\cdot\tilde{f}_{n+1}(\sigma)(\bar{e}) = \nu_{f;\tilde{f}_{n+1}}(\sigma;\bar{e})$.

If $\mathrm{sfp}_{n+2}(\sigma;\bar{e}) = \infty$, then either $\sigma \notin f$ or $\bar{e} \notin \chi(\sigma)$ in both cases $\nu_{f;\chi}(\sigma;\bar{e}) = 0$.

Finally we have that $f'_{n+1} = f;\chi$ is a scheduler by assumption. To prove that s_{n+2} extends f'_{n+1} we have to show that it is indeed a fragment scheduler. If $s_{n+2}(\sigma;\bar{e}) \neq \{\partial\}$ there are two possibilities:

– $s_{n+2}(\sigma;\bar{e}) = \tilde{f}_{n+2}(\sigma;\bar{e})$: if this is the case, since $\tilde{f}_{n+2}$ is a fragment scheduler, then $s_{n+2}(\sigma;\bar{e})$ is an action at $\sigma;\bar{e}$;
– $s_{n+2}(\sigma;\bar{e}) = \tilde{f}_{n+1}(\sigma)$: if this is the case, then $\mathrm{sfp}_{n+2}(\sigma;\bar{e}) = \perp$, which implies that $\tilde{f}_{n+1}(\sigma) \neq \chi(\sigma)$ and that $\bar{e} \in \chi(\sigma)$. By Lemma 2, $\tilde{f}_{n+1}(\sigma)$ is still an action at $\sigma;\bar{e}$.

The basis is proved.

For the inductive step, assume the (long) statement true for i, we are going to prove it for $i+1$. First notice that, since f'_{n+i-1} is a scheduler, and s_{n+i} extends it, then f'_{n+i} is indeed a scheduler.

If $\mathrm{sfp}_{n+i+1}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i}) = \perp$, there are five clauses to prove. In order to do this, we first notice that by definition of sfp_{n+i+1}, it is the case that $\mathrm{sfp}_{n+i}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i-1}) = \perp$.

1. By induction hypothesis we have that $\sigma;\bar{e}e_{n+2}\ldots e_{n+i-1} \in f'_{n+i-1}$. Moreover $e_{n+i} \in s_{n+i}(\sigma;\bar{e}e_{n+2}\ldots e_{n+i-1})$. Thus $\sigma;\bar{e}e_{n+2}\ldots e_{n+i} \in f'_{n+i-1};s_{n+i} = f'_{n+i}$;
2. By induction hypothesis we have that $e_{n+2},\ldots,e_{n+i-1}$ are independent of $\bar{e}$. Moreover, by definition of sfp_{n+i+1}, $e_{n+i} \in s_{n+i}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i-1})$. By induction hypothesis, $\chi(\sigma)$ is an action at $\sigma;e_{n+2},\ldots,e_{n+i-1}$. Since $\sigma;\bar{e}e_{n+2}\ldots e_{n+i-1} \in f'_{n+i-1}$, in particular it is the case that $\bar{e} \in \chi(\sigma)$. Again by induction hypothesis, $\bar{e} \notin \tilde{f}_{n+i-1}(\sigma;e_{n+2}\ldots e_{n+i-1})$. Thus $\chi(\sigma)$ is different from $\tilde{f}_{n+i-1}(\sigma;e_{n+2},\ldots,e_{n+i-1})$. Since $\mathrm{sfp}_{n+i}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i-1}) = \perp$ we have that $\tilde{f}_{n+i-1}(\sigma;e_{n+2},\ldots,e_{n+i-1}) = s_{n+i}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i-1})$. So $\bar{e}$ is also independent of e_{n+i}.
3. By induction hypothesis, $\chi(\sigma)$ is an action at $\sigma;e_{n+2},\ldots,e_{n+i-1}$. We have shown above that e_{n+i} is independent of $\bar{e} \in \chi(\sigma)$. Therefore every transition in $\chi(\sigma)$ is independent of e_{n+i}. By Lemma 2, $\chi(\sigma)$ is still an action at $\sigma;e_{n+2},\ldots,e_{n+i}$.
4. As observed above, $\bar{e} \in \chi(\sigma)$. By definition of sfp_{n+i+1}, we have that $\chi(\sigma) \neq \tilde{f}_{n+i}(\sigma;e_{n+2},\ldots,e_{n+i})$. They are both actions at $\sigma;e_{n+2},\ldots,e_{n+i}$, therefore they are disjoint.
5. $\nu_{f'_{n+i}}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i})$

$$= \nu_{f'_{n+i-1}}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i-1}) \cdot s_{n+i}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i-1})(e_{n+i})$$
$$= \nu_{f;\tilde{f}_{n+i-1}}(\sigma;e_{n+2},\ldots,e_{n+i-1}) \cdot \chi(\sigma)(\bar{e}) \cdot s_{n+i}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i-1})(e_{n+i}) \ .$$

As noticed above $s_{n+i}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i-1}) = \tilde{f}_{n+i-1}(\sigma;e_{n+2},\ldots,e_{n+i-1})$. Thus

$$\nu_{f;\tilde{f}_{n+i-1}}(\sigma;e_{n+2},\ldots,e_{n+i-1}) \cdot \chi(\sigma)(\bar{e}) \cdot s_{n+i}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i-1})(e_{n+i})$$
$$= \nu_{f;\tilde{f}_{n+i-1}}(\sigma;e_{n+2},\ldots,e_{n+i-1}) \cdot \chi(\sigma)(\bar{e}) \cdot \tilde{f}_{n+i-1}(\sigma;e_{n+2},\ldots,e_{n+i-1})(e_{n+i})$$
$$= \nu_{f;\tilde{f}_{n+i-1}}(\sigma;e_{n+2},\ldots,e_{n+i-1}) \cdot \tilde{f}_{n+i-1}(\sigma;e_{n+2},\ldots,e_{n+i-1})(e_{n+i}) \cdot \chi(\sigma)(\bar{e})$$
$$= \nu_{f;\tilde{f}_{n+i-1}}(\sigma;e_{n+2},\ldots,e_{n+i}) \cdot \chi(\sigma)(\bar{e})$$

and the result is proven.

If $\mathrm{sfp}_{n+i+1}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i}) = h \in \mathbb{N}$, there are two possibilities. First $h < n+i$. This implies $\mathrm{sfp}_{n+i}(\sigma;\bar{e}e_{n+2},\ldots,e_{n+i-1}) = h$, already. We have to show two clauses:

1. by induction hypothesis $\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1} \equiv \text{shift}_{n+1,h}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1})$. Thus $\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1}; e_{n+i} \equiv \text{shift}_{n+1,h}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1}); e_{n+i} = \text{shift}_{n+1,h}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i})$.
2. $s_{n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1}) = \tilde{f}_{n+i}(\text{shift}_{n+1,h}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1}))$. Thus, using the induction hypothesis:

$$\begin{aligned}
&\nu_{f'n+i}(\sigma; e_{n+2}, \ldots, e_{n+i}) \\
&= \nu_{f'n+i-1}(\sigma; e_{n+2}, \ldots, e_{n+i-1}) \cdot s_{n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1})(e_{n+i}) \\
&= \nu_{f;\tilde{f}_{n+i-1}}(\text{shift}_{n+1,h}(\sigma; e_{n+2}, \ldots, e_{n+i-1})) \cdot s_{n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1})(e_{n+i}) \\
&= \nu_{f;\tilde{f}_{n+i-1}}(\text{shift}_{n+1,h}(\sigma; e_{n+2}, \ldots, e_{n+i-1})) \cdot \\
&\quad \cdot \tilde{f}_{n+i}(\text{shift}_{n+1,h}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1}))(e_{n+i}) \\
&= \nu_{f;\tilde{f}_{n+i}}(\text{shift}_{n+1,h}(\sigma; e_{n+2}, \ldots, e_{n+i}))
\end{aligned}$$

If instead $h = n + i$ then $\text{sfp}_{n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1}) = \perp$.

1. by induction hypothesis $e_{n+2}, \ldots, e_{n+i-1}$ are independent of $\bar{e}$. Moreover $\bar{e} \notin \tilde{f}_{n+i-1}(\sigma; e_{n+2}, \ldots, e_{n+i-1})$. But since $\text{sfp}_{n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1}) = \perp$ then it is the case that $\tilde{f}_{n+i-1}(\sigma; e_{n+2}, \ldots, e_{n+i-1}) = s_{n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1})$. And since $e_{n+i} \in s_{n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1})$, then $\bar{e}$ is also independent of e_{n+i}. Therefore
$\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i} \equiv \text{shift}_{n+1,n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1}) = \sigma; e_{n+2}, \ldots, e_{n+i}\bar{e}$.
2. similarly to point 5 above, we can show that $\nu_{f'_{n+i}}(\sigma; \bar{e}e_{n+2} \ldots e_{n+i}) = \nu_{f;\tilde{f}_{n+i-1}}(\sigma; e_{n+2} \ldots e_{n+i}) \cdot \chi(\sigma)(\bar{e})$. By definition of sfp_{n+i+1}, we have that $\chi(\sigma) = \tilde{f}_{n+i}(\sigma; e_{n+2} \ldots e_{n+i})$. Therefore

$$\begin{aligned}
&\nu_{f;\tilde{f}_{n+i-1}}(\sigma; e_{n+2} \ldots e_{n+i}) \cdot \chi(\sigma)(\bar{e}) = \\
&\quad \nu_{f;\tilde{f}_{n+i-1}}(\sigma; e_{n+2} \ldots e_{n+i}) \cdot \tilde{f}_{n+i}(\sigma; e_{n+2} \ldots e_{n+i-1})(\bar{e}) = \\
&\quad \nu_{f;\tilde{f}_{n+i}}(\sigma; e_{n+2} \ldots e_{n+i}; \bar{e}) = \\
&\quad \nu_{f;\tilde{f}_{n+i}}(\text{shift}_{n+1,n+i}(\sigma; \bar{e}e_{n+2} \ldots e_{n+i-1}))\ .
\end{aligned}$$

Finally, if $\text{sfp}_{n+i+1}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i}) = \infty$ then there are two possibilities: either $\text{sfp}_{n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1}) = \infty$, so that by induction hypothesis $\nu_{f'_{n+i-1}}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1}) = 0$, or $e_{n+i} \notin s_{n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1})$, which implies that $s_{n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1})(e_{n+i}) = 0$. Either case we have

$$\begin{aligned}
&\nu_{f'_{n+i}}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i}) = \\
&= \nu_{f'_{n+i-1}}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1}) \cdot s_{n+i}(\sigma; \bar{e}e_{n+2}, \ldots, e_{n+i-1})(e_{n+i}) = 0\ .
\end{aligned}$$

Finally, we show that s_{n+i+1} is a fragment scheduler and that it therefore extends f'_{n+i}. If $s_{n+i+1}(\sigma; \bar{e}e_{n+2} \ldots e_{n+i}) \neq \{\partial\}$ there are two possibilities.

- $s_{n+i+1}(\sigma; \bar{e}e_{n+2} \ldots e_{n+i}) = \tilde{f}_{n+i+1}(\text{shift}_{n+1,h}(\sigma; \bar{e}e_{n+2} \ldots e_{n+i}))$: if this is the case, then $\text{sfp}_{n+i+1}(\sigma; \bar{e}e_{n+2} \ldots e_{n+i}) = h$. Therefore we have that $\text{shift}_{n+1,h}(\sigma; \bar{e}e_{n+2} \ldots e_{n+i})) \equiv \sigma; \bar{e}e_{n+2} \ldots e_{n+i}$, and this implies that $\tilde{f}_{n+i+1}(\text{shift}_{n+1,h}(\sigma; \bar{e}e_{n+2} \ldots e_{n+i}))$ is an action at $\sigma; \bar{e}e_{n+2} \ldots e_{n+i}$.

– $s_{n+i+1}(\sigma;\bar{e}e_{n+2}\dots e_{n+i}) = \tilde{f}_{n+i}(\sigma;e_{n+2}\dots e_{n+i})$: if this is the case, then $\mathrm{sfp}_{n+i+1}(\sigma;\bar{e}e_{n+2}\dots e_{n+i}) = \perp$. Therefore $\chi\sigma \neq \tilde{f}_{n+i}(\sigma;e_{n+2}\dots e_{n+i})$, $\chi\sigma$ is an action at $\sigma;e_{n+2}\dots e_{n+i}$ and $e_{n+2}\dots e_{n+i}$ are independent of $\bar{e}$. This means that $\sigma;\bar{e}e_{n+2}\dots e_{n+i} \equiv \sigma;e_{n+2}\dots e_{n+i}\bar{e}$. Moreover, by Lemma 2, $\tilde{f}_{n+i}(\sigma;e_{n+2}\dots e_{n+i})$ is still an action at $\sigma;e_{n+2}\dots e_{n+i}\bar{e}$. Therefore it is also an action at $\sigma;\bar{e}e_{n+2}\dots e_{n+i}$.

$\square$

Now to prove Lemma 5 we first have to show that t is a fragment scheduler. We have only to show that when $\sigma;e'_{n+1}\dots e'_k \in f;\tilde{f}$ and $\not\exists l.\ e'_l \in \chi(\sigma)$, then $\chi(\sigma)$ is still an action at $\sigma;e'_{n+1}\dots e'_k$. This is exactly what Lemma 3 is saying. The remaining part of the statement follows easily from Lemma 6 and it is left to the reader. $\square$