Translating Structural Process Properties to Petri Net Markings

Sven Linker
Department of Computing Science
Carl von Ossietzky Universität
Oldenburg, Germany
Email: sven.linker@informatik.uni-oldenburg.de

***Abstract*—We introduce a spatio-temporal logic PSTL defined on Pi-Calculus processes. This logic is especially suited to formulate properties in relation to the structural semantics of the Pi-Calculus due to Meyer, a representation of processes as Petri nets. To allow for the use of well-researched verification techniques, we present a translation of a subset of PSTL to LTL on Petri nets. We further prove soundness of our translation.**

***Keywords*-process algebra; Pi-Calculus; spatial logic; temporal logic; petri nets; verification**

I. Introduction

The structure of computer systems has drastically changed from single computing devices to small, communicating components in the last years. Hence verification of structural properties such as stability of connections between processes is of growing interest, e.g. to ensure reliability in cases of emergency.

Meyer has shown that a large class of concurrent systems can be translated into place/transition Petri nets [1], [2]. Using this translation, standard Petri net verification techniques may be employed. At the moment, verification properties have to be directly formulated as LTL formulae on Petri nets, i.e. after translating a process into its Petri net semantics, the user has to manually inspect the translation. Since this approach is error-prone as well as unintuitive, we propose the use of a structural logic on processes inspired by the approach of Caires and Cardelli [3] to formulate properties.

We present a structural-temporal logic on π-Calculus processes called PSTL (π-Calculus *s*tructural *t*emporal *l*ogic), where temporal modalities may not occur within the scope of structural modalities. I.e., we use the structural connectives to describe processes and the temporal operators to relate these descriptions along the evolution of processes. This approach is an application of Finger and Gabbay's technique of temporalising logics [4].

For this logic we present a sound translation to LTL on Petri nets with respect to a process P. That is, we translate a formula φ on processes into a formula φ' describing the evolution of markings in the structural semantics corresponding to the processes reachable from P satisfying φ. The translation is possible for the rather large set of *structurally stationary* processes. This set contains, e.g., *recursion-free*, *bounded* [5] and *restriction-free* [6] processes as shown by Meyer [1], [2].

The organisation of the paper is as follows. After establishing the necessary preliminaries in the next section, we give the translation of the structural part of PSTL to LTL in Sect. III. In Sect. IV we define the full logic PSTL and lift the translation accordingly. Section V concludes the paper. For reviewing purposes, we add an appendix.

II. Preliminaries

Definition 1 (Place/Transition Petri Net)**.** *A* place/transition Petri net *is a tuple $N = (S, T, W, M_0)$, where S and T are disjoint sets of* places*, respectively* transitions*. The* weight function *$W\colon (S \times T) \cup (T \times S) \to \mathbb{N}$ defines the number of arcs between places and transitions. The* initial marking *$M_0\colon S \to \mathbb{N}$ gives the number of tokens on each place. A Petri net is* finite *if both S and T are finite.*

A transition t is enabled *under a marking $M\colon S \to \mathbb{N}$, if $M(s) \geq W(s,t)$ for all $s \in S$. The* transition relation *$\rightarrowtriangle$ is defined by $M \rightarrowtriangle M'$ iff there is a transition t, such that t is enabled under M and $M'(s) = M(s) - W(s,t) + W(t,s)$ for all $s \in S$.*

Let $\omega = M_0 M_1 M_2 \ldots$ be an infinite sequence of markings, where M_0 is the initial marking of a Petri net N and $M_i \rightarrowtriangle M_{i+1}$ for each $M_i \neq M_{i+1}$. If there is some k such that M_k has no successor, we define $M_i = M_k$ for all $i > k$. We call ω an occurrence sequence *of N. The notation ω^i refers to the subsequence of ω starting at the i-th marking. The initial marking of ω is denoted by M_ω. The* support *of a marking M is the set $supp(M) = \{p \mid M(p) > 0\}$.*

The π-Calculus is a process algebra capable of describing dynamically reconfigurable systems [7], [8]. Its main concept is sharing and hiding of *names* between processes. We assume a countably infinite set $\mathcal{A}$ of names and will abbreviate a sequence of names $m_0 m_1 m_2 \ldots m_n$ by $\tilde{m}$. Names can be used as channels by which the processes communicate, but also as the content of a message. By this mechanism, previously unconnected

processes may establish new communication channels. *Prefixes* π define possible communications of processes. A prefix is of one of the following forms: $\overline{m}\langle n\rangle$ expresses the ability to *send* the name n over the channel m, $m(x)$ gives the process the possibility to *receive* a name over channel m and replacing x with the received name. The prefix τ represents an internal action, while the *matching* prefix $[x = y]$ only allows the process to proceed, if x and y are equal. Matching prefixes must be followed by another prefix, i.e. a process with a matching prefix is of the form $[x = y].\pi.P$.

Definition 2 (Processes)**.** *Processes P are constructed according to the following EBNF, where m is a name and $\tilde{m}$ a name sequence.*

$$M := \mathbf{0} \mid \pi.P \mid M_1 + M_2$$
$$P := M \mid K\lfloor\tilde{m}\rfloor \mid P_1 \,|\, P_2 \mid \nu m.P_1$$

The process $\mathbf{0}$ denotes the process which possesses no behaviour. The operators are *choice* $(+)$ and *parallel composition* $(\,|\,)$. That a certain name m is only known to a certain set of processes is expressed by *restriction* (ν) on a process. A *process call* is denoted by $K\lfloor\tilde{m}\rfloor$, if there is a *process equation* $K(\tilde{x}) := P_K$. In the following, we will use the notation $\prod_{i\in I} P_i$ for the process $P_1 \mid \ldots \mid P_n$, where $I = \{1, \ldots, n\}$. If $I = \emptyset$, we set $\prod_{i\in I} P_i = \mathbf{0}$. If $P \equiv P_1 \equiv P_2 \equiv \ldots \equiv P_n$, we also write $\prod^n P$.

The set of *free names* of a process P, denoted $fn(P)$, is the collection of all names occurring in P, which are not bound by a restriction or an input action. Similarly, the set of *bound names* of P is the collection of names occurring in P under a restriction or bound by an input action. Definitions of the π-Calculus normally require that $fn(P_K) \subseteq \tilde{x}$ for process equations. For brevity, we will violate this condition in the examples, and silently assume all parameter lists to be completed accordingly.

Definition 3 (Structural Congruence)**.** *The* structural congruence $\equiv$ *is the smallest congruence relation on processes allowing for alpha-conversion of bound names, where $+$ and $\mid$ are commutative and associative operators with $\mathbf{0}$ as the neutral element. The quantifier ν is commutative and absorbed by $\mathbf{0}$. The matching prefix may be absorbed, if the compared names are equal. In addition $\equiv$ has to fulfil the following law, called* scope extrusion.

$$\nu a.(P|Q) \equiv P|\nu a.Q \qquad \text{where } a \notin fn(P)$$

We will denote the equivalence class of a process P with respect to structural congruence by $[P]$ and the set of equivalence classes of a set $\mathcal{S}$ by $\mathcal{S}_{/\equiv}$.

The *substitution* of a name a by another name m is denoted by $P\{m/a\}$. Furthermore we denote the direct substitution of name sequences by $P\{\tilde{m}/\tilde{a}\}$. *Reactions* define the behaviour of processes.

Definition 4 (Reaction)**.** *The* reaction relation $\rightarrow$ *is defined as follows.*

$$\textit{(Tau)} \qquad \tau.P + M \rightarrow P$$
$$\textit{(React)} \qquad (x(y).P + M) \,|\, (\overline{x}\langle z\rangle.Q + N) \rightarrow P\{z/y\} \,|\, Q$$
$$\textit{(Const)} \qquad K\lfloor\tilde{a}\rfloor \rightarrow P\{\tilde{a}/\tilde{x}\}, \text{ if } K(\tilde{x}) := P$$

$$\textit{(Par)} \ \frac{P \rightarrow P'}{P \,|\, Q \rightarrow P' \,|\, Q} \qquad \textit{(Res)} \ \frac{P \rightarrow P'}{\nu a.P \rightarrow \nu a.P'}$$

$$\textit{(Struct)} \ \frac{Q \equiv P \qquad P \rightarrow P' \qquad P' \equiv Q'}{Q \rightarrow Q'}$$

We will denote the set of all reachable processes from P by $Reach(P)$.

Example 1. *As a running example, we use a part of the specification taken from Orava and Parrow [9], which formally defines a handover procedure of the GSM protocol. The full specification is, e.g., available as an example provided by the tool* Petruchio *[10].*

The system consists of four agents. Two base stations, a passive one BS_p and an active one BS_a, a mobile station MS and a mobile switching centre MSC. The mobile station is connected to the active base station by a private link m. The mobile switching centre serves two purposes. It connects the base stations over the channels f_a and f_p and coordinates the handover of the mobile station from the active base station to the passive station. The mobile station is able to either receive data or to act according to the handover procedure. To distinguish both behaviours, the base station sends the one of the names data and ho_cmd on the shared channel m.

$$\begin{aligned} MS(m) := m(x).\ ([x = data].m(v).\overline{out}\langle v\rangle.MS\lfloor m\rfloor \\ + [x = ho\_cmd].m(m_{new}). \\ (\overline{m_{new}}\langle ho\_acc\rangle.MS\lfloor m_{new}\rfloor \\ + \overline{m}\langle ho\_fail\rangle.MS\lfloor m\rfloor)) \end{aligned}$$

If the received name is data, the mobile station may receive the value on the shared channel and then outputs it on the channel out. If otherwise the handover procedure is initiated, the station receives a new channel m_{new}. This channel shall be used to communicate with the new base station. Now the station may either complete the handover by sending ho_acc on this new channel or it may cancel the handover and proceed to communicate on the channel m. For the definitions of BS_p, BS_a and MSC, we refer to the paper of Orava and Parrow [9] and the appendix. To allow for the sending of data values to the mobile station, we also employ the processes $S := \nu v.\overline{in}\langle v\rangle.S$ and $R := out(d).R$.

The full system is defined by the following process

definitions.

$$P(f_a, f_p) := \nu m.(MSC\lfloor f_a, f_p, m\rfloor \mid BS_p\lfloor f_p, m\rfloor)$$
$$Q(f_a) := \nu m.(BS_a\lfloor f_a, m\rfloor \mid MS\lfloor m\rfloor)$$
$$System := \nu f_a, f_p.(P\lfloor f_a, f_p\rfloor \mid Q\lfloor f_a\rfloor \mid S \mid R)$$

The process P models the connection of the passive base station to the mobile switching centre, while Q defines that the mobile station is only connected to the active base station. Note that both processes define a restriction on a name m. These occurrences of m denote distinct *names, since they are included in the scopes of different restrictions.*

The structural semantics of Meyer [1] is a mapping from π-Calculus processes to place/transition Petri nets respecting the internal structure of the process, i.e. structural congruence. Its main features are *full retrievability* and *full abstraction*. Full retrievability means that the transition systems of a process P and its structural semantics are isomorphic, as well as the fact that every reachable process from P can be retrieved from a reachable marking of its structural semantics. Full abstraction denotes the property that for every two processes P and Q the structural semantics yields structurally similar Petri nets, if and only if P and Q are structurally congruent.

The mapping heavily relies on a syntactic normal form of processes, the *restricted form* of a process. In the restricted form of a process, the scope of restrictions ν is as small as possible, building *fragments*, where every process under a restriction on a name m contains m as a free name.

Definition 5 (Fragments and Restricted Form)**.** *A* fragment *is a process constructed according to the following definition.*

$$F := M_1 + M_2 \mid K\lfloor\tilde{a}\rfloor \mid \nu m.(F_1 \mid \ldots \mid F_n),$$

where $m \in fn(F_i)$ for all $i \in \{1, \ldots, n\}$ and $M_1 + M_2$ is a summation not structurally equivalent to $\mathbf{0}$. A process P^{rf} in restricted form *consists of a parallel composition of fragments, i.e. $P^{rf} = \prod_{i\in I} F_i$. We denote the set of all fragments by $\mathcal{P}^{\mathcal{F}}$ and the set of fragments of P^{rf} by $fg(P^{rf})$.*

Meyer characterised the function rf, computing the restricted form of a process, syntactically [1]. Intuitively, the application of rf shrinks the scope of all restrictions in P as much as possible and removes processes congruent to $\mathbf{0}$. For the definition of the structural semantics of a process P, we need a function to count the occurrences of fragments in its restricted form.

Definition 6 (Decomposition)**.** *Let $P^{rf} = rf(P)$ be a process in restricted form and $I_F = \{\, G \in fg(P^{rf}) \mid G \equiv F \,\}$. We associate a function $dec(P^{rf})\colon \mathcal{P}^{\mathcal{F}}/_{\equiv} \to \mathbb{N}$ with P^{rf} such that $dec(P^{rf})([F]) = |I_F|$.*

Lemma 1. *Consider two processes P and Q. Then the following properties hold.*

1) *$dec(rf(P \mid Q)) = dec(rf(P)) + dec(rf(Q))$*
2) *$P \equiv Q$ if and only if $dec(rf(P)) = dec(rf(Q))$*

With these preliminaries established, we are able to define the structural semantics $\mathcal{N}[\![P]\!]$ of a process P. The places of $\mathcal{N}[\![P]\!]$ are exactly the congruence classes of reachable fragments of P. A transition models reactions between two fragments F_1 and F_2 to a new process Q, which may again consist of several fragments, or a reaction within one fragment F to a process Q.

Definition 7 (Structural Semantics)**.** *The* structural semantics *maps a process P to a Petri net $\mathcal{N}[\![P]\!] = (S, T, W, M_0)$, where S is the set of places, T the set of transitions and M_0 the initial marking given by the following definitions.*

$$\begin{aligned} S &= fg(rf(Reach(P)))_{/\equiv} \\ T &= \{\, ([F], [Q]) \subseteq S \times \mathcal{P}_{/\equiv} \mid F \to Q \,\} \\ &\quad \cup \{\, ([F_1 \mid F_2], [Q]) \subseteq \mathcal{P}_{/\equiv} \times \mathcal{P}_{/\equiv} \\ &\qquad \mid [F_1], [F_2] \in S \text{ and } F_1 \mid F_2 \to Q \,\} \\ M_0 &= dec(rf(P)) \end{aligned}$$

For $[G] \in S$ and transitions $([F], [Q]), ([F_1 \mid F_2], [Q]) \in T$, the weight function is defined by

$$\begin{aligned} W([G], ([F], [Q])) &= dec(F)([G]) \\ W([G], ([F_1 \mid F_2], [Q])) &= dec(F_1 \mid F_2)([G]) \\ W(([F], [Q]), [G]) &= dec(rf(Q))([G]) \\ W(([F_1 \mid F_2], [Q]), [G]) &= dec(rf(Q))([G]) \end{aligned}$$

Remark 1. *As shown by Meyer, the transition systems of a process and its structural semantics are isomorphic. That is, the evolution of markings of the structural semantics of P exactly reflects the reduction sequences of P.*

Example 2. *Consider Example 1. If we shrink the scopes of the restrictions in the definition of System, we get*

$$System := \nu f_a.(\nu f_p.(P\lfloor f_a, f_p\rfloor) \mid Q\lfloor f_a\rfloor) \mid S \mid R,$$

which is a process in restricted form consisting of the three fragments S, R and $\nu f_a.(\nu f_p.(P\lfloor f_a, f_p\rfloor) \mid Q\lfloor f_a\rfloor)$. On the whole, 223 structurally incongruent fragments are reachable from System. Two of these fragments are

$$\begin{aligned} F \equiv \nu m, f_p.\Big(& BS_p\lfloor f_p, m\rfloor \\ & \mid \nu l.\big(HC\lfloor l, m\rfloor \mid \nu f_a.(CC\lfloor f_a, f_p, l\rfloor \\ & \mid \nu m.(BS_a\lfloor f_a, m\rfloor \mid \nu d.\overline{out}\langle d\rangle.MS\lfloor m\rfloor))\big)\Big) \end{aligned}$$

and $G \equiv out(v).R$. The calls to the process definitions CC and HC are parts the mobile switching centre MSC. The set of transitions of the structural semantics contains $([F \mid G], [F'|R])$, where F' is similar to the fragment F, in all but the process $\nu d.\overline{out}\langle d\rangle.MS\lfloor m \rfloor$, which is replaced by $MS\lfloor m \rfloor$. The weight function is defined by

$$W([F], ([F \mid G], [F'|R])) = 1$$
$$W([G], ([F \mid G], [F'|R])) = 1$$
$$W(([F \mid G], [F'|R]), [F']) = 1$$
$$W(([F \mid G], [F'|R]), [R]) = 1,$$

and

$$W([H], ([F \mid G], [F'|R])) = W(([F \mid G], [F'|R]), [H]) = 0$$

for all other fragments H. I.e., the transition may only fire if the places $[F]$ and $[G]$ are marked with a token and puts a token on the places $[F']$ and $[R]$. With this definition, it models the reduction of the process $F \mid G$ to $F' \mid R$ via the communication over the channel out. The second type of transitions defines reaction within a fragment, e.g. the expansion of a process definition. For example,

$$W([R], ([R], [out(d).R])) = 1$$
$$W(([R], [out(d).R]), [out(d).R]) = 1.$$

If the set of reachable fragments of a process P is finite, we call P *structurally stationary*. By Definition 7, the structural semantics of P is finite if and only if P is structurally stationary. Decidable properties of place/transition Petri nets directly carry over to structurally stationary processes, due to the isomorphic transition systems of processes and their structural semantics [1]. Such properties include, e.g., reachability [11] and coverability [12] of markings.

We briefly recall LTL with the *eventually* modality on Petri nets.

Definition 8 (LTL)**.** *The syntax is given by the following EBNF*

$$\varphi := p \geq c \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \Diamond\varphi,$$

where p is the place of a Petri net and $c \in \mathbb{N}$. The semantics of LTL on Petri Nets is defined inductively, where ω is an occurrence sequence.

$\omega \models p \geq c$	*iff*	$M_\omega(p) \geq c$
$\omega \models \neg\varphi$	*iff*	*not* $\omega \models \varphi$
$\omega \models \varphi_1 \wedge \varphi_2$	*iff*	$\omega \models \varphi_1$ *and* $\omega \models \varphi_2$
$\omega \models \Diamond\varphi$	*iff*	$\exists y\colon \omega^y \models \varphi$

If $\omega \models \varphi$ we say that ω satisfies *φ. A Petri net N* satisfies *a formula φ, written $N \models \varphi$, if all occurrence sequences of N satisfy φ.*

We also use $p = c$, which abbreviates $p \geq c \wedge \neg(p \geq c + 1)$, as an atomic formula.

III. Translating Structural Formulae

In the following we define the syntax and semantics of the structural part of the logic. We refer to a subset of processes without parallel composition or restriction as the topmost operator. Furthermore, the process $\mathbf{0}$ is not part of this subset. We call this set *sequential processes*.

In addition to atoms $\top$ and $\bot$ denoting *true* and *false* respectively, we use the notions $free(b)$ and $\neg free(b)$ ($b \in \mathcal{A}$) to express that b is either free (not free, respectively) in the process under consideration. Finally, every sequential process P^{seq} is allowed as a *sequential atom*. Such a formula is satisfied by a process P iff P is structurally congruent to P^{seq}.

In contrast to the work of Caires and Cardelli [3], we consider the *restriction quantifier* res to be a primitive notion and do not split it into a freshness quantifier and an operation to reveal hidden names. A process P satisfying res $b\ \varphi$ can be decomposed into a restriction on a process P', i.e., $P \equiv \nu m.P'$ and P' satisfies φ, where b was substituted by a fresh name m in φ. To mimic the parallel composition of processes we use the binary parallelism modality $\parallel$. A process P satisfying $\varphi \parallel \psi$ can be split into processes Q and R, which satisfy φ and ψ respectively. Note that we do not allow for Boolean operators on the structural level. Conjunction and negation will only arise at the level of temporal formulae. Thus, we avoid ambiguity.

Definition 9 (Syntax of Structural Formulae)**.** *The syntax of* structural formulae *is defined as follows, where P^{seq} is a sequential process and $b \in \mathcal{A}$.*

$$\beta ::= \bot \mid \top \mid free(b) \mid \neg free(b) \mid P^{\text{seq}} \mid \beta_1 \parallel \beta_2 \mid \text{res } b\ \beta$$

We use $\coprod_{i \in I} \beta_i$ to denote the formula $\beta_1 \parallel \ldots \parallel \beta_n$, where $I = \{1, \ldots, n\}$. If $I = \emptyset$, we set $\coprod_{i \in I} \beta_i = \bot$. The definitions of the sets of free and bound names of formulae are as usual, i.e. a name is free if it is not bound by ν, res or an input action. We denote the set of subformulae of β by $sub(\beta)$. For $0 \leq i \leq n$, we denote by $\{a_0, \ldots, a_n \leftarrow m_0, \ldots, m_n\}$ or $\{\tilde{a} \leftarrow \tilde{m}\}$ the *capture-avoiding substitution* σ with $\sigma(a_i) = m_i$ and $\sigma(x) = x$ for all $x \notin \{a_0, \ldots, a_n\}$. In the rest of the paper, we assume that the sets of the bound names of formulae and processes are disjoint. This is not a loss of generality, since we always may employ alpha-conversion to achieve this setting. The semantics captures the intuitive meanings of the operators given above.

Definition 10 (Semantics of Structural Formulae)**.** *Let P be a process and $b \in \mathcal{A}$. The* satisfaction relation $\models$ *is given according to the following definition.*

$P \models \bot$ *for no* P
$P \models \top$ *for all* P
$P \models \mathit{free}(b)$ *iff* $b \in \mathit{fn}(P)$
$P \models \neg\mathit{free}(b)$ *iff* $b \notin \mathit{fn}(P)$
$P \models P^{\mathrm{seq}}$ *iff* $P \equiv P^{\mathrm{seq}}$
$P \models \beta_1 \parallel \beta_2$ *iff* $\exists Q, R\colon P \equiv Q \mid R$ *such that* $Q \models \beta_1$ *and* $R \models \beta_2$
$P \models \mathrm{res}\ b\ \beta$ *iff* $\exists m, Q\colon P \equiv \nu m.Q$ *such that* $m \notin (\mathit{fn}(P) \cup \mathit{fn}(\beta))$ *and* $Q \models \beta\{b \leftarrow m\}$

Two formulae β_1 *and* β_2 *are* equivalent*, denoted by* $\beta_1 \mathrel{\widehat{\equiv}} \beta_2$*, if and only if for all processes* P*,* $P \models \beta_1$ *iff* $P \models \beta_2$*.*

Example 3. *This logic is capable of describing structural properties of processes. Consider Example 1. The formula* $\beta = \mathrm{res}\ y\ (MS\lfloor y \rfloor \parallel \top)$ *denotes the existence of the process call to the mobile station. Note that a process satisfying* β *has to contain a process structurally congruent to* $MS\lfloor m \rfloor$*, e.g., the process* $\nu\, m, v.\overline{out}\langle v\rangle.MS\lfloor m \rfloor$ *does not satisfy* β*, but* $\nu m.MS\lfloor m \rfloor$ *does.*

The parallelism modality is commutative and associative. For the translation defined in Section III-A, we need the following additional equivalences. The proofs are easy and therefore omitted.

Proposition 1. *Let* β *be a structural formula and* $b \in \mathcal{A}$*. Then the following equivalences hold.*

$$\top \parallel \top \mathrel{\widehat{\equiv}} \top \tag{1}$$
$$\mathit{free}(b) \parallel \top \mathrel{\widehat{\equiv}} \mathit{free}(b) \tag{2}$$
$$\neg\mathit{free}(b) \parallel \top \mathrel{\widehat{\equiv}} \top \tag{3}$$
$$\mathrm{res}\ b\ (\beta \parallel \top) \parallel \top \mathrel{\widehat{\equiv}} \mathrm{res}\ b\ (\beta \parallel \top) \tag{4}$$

A. Translatable Subset

In this section we define the translation of a subset of structural formulae to propositional formulae on Petri nets. To examine the truth value of $P \models \beta$, we construct a formula $\theta(\beta, P)$ which is satisfied by the structural semantics of P if and only if P satisfies β. Since P determines the names of places occurring in its structural semantics, the translation to LTL can only be formulated with respect to P, i.e., with respect to the reachable fragments of P.

We will call a translatable formula a *restriction formula* (cf. Definition 12) since the structure of these formulae is very similar to the *restricted form* of processes as defined by Meyer [1]. A restriction formula consists of a parallel composition of one or more *fragment formulae*. Like fragments of processes, such a formula consists of a restriction on a name b around a parallel composition of other fragment formulae. In contrast to process fragments, we do not require that all composed fragment formulae contain b as a free name, but allow for $\top$ to avoid overspecifications.

The idea of the translation is as follows. For a process P and a formula $\beta = \coprod_{i\in I} \beta_i$ we first identify the set of fragments of P satisfying each β_i. Then from these sets we construct the minimal processes needed for the satisfaction of β. The minimal processes can be decomposed according to the structural semantics into their fragments, so that we can identify the number of tokens on the corresponding places in the structural semantics. The formula associated with a minimal process is then a conjunction of atoms. Finally, the disjunction of all such conjunctions is the translated formula φ.

For the definition of fragment formulae, observe that the existence of a free name b in a formula β does not imply that all processes satisfying β also have b in the set of their free names. Consider for example the formula $\beta = \neg\mathit{free}(b)$. In the standard definition of free names on formulae, β uses b as a free name, but for every process P with $P \models \beta$, $b \notin \mathit{fn}(P)$. If we consider the set of free names of a formula to be a commitment of its satisfying processes, it would be more sensible to say that $\neg\mathit{free}(b)$ does not contain *any* free names. To take these considerations into account, we define a set of *ensured free names* of a formula.

Definition 11 (Ensured Free Names)**.** *In the following, let* P^{seq} *be a sequential process,* $b \in \mathcal{A}$ *and* α, β *be structural formulae. The* ensured free names *of a formula are defined as follows.*

$\mathit{efn}(\top) = \mathit{efn}(\bot) = \emptyset$ $\quad \mathit{efn}(P^{\mathrm{seq}}) = \mathit{fn}(P^{\mathrm{seq}})$
$\mathit{efn}(\mathit{free}(b)) = \{b\}$ $\quad \mathit{efn}(\mathrm{res}\ b\ \beta) = \mathit{efn}(\beta) \setminus \{b\}$
$\mathit{efn}(\neg\mathit{free}(b)) = \emptyset$ $\quad \mathit{efn}(\alpha \parallel \beta) = \mathit{efn}(\alpha) \cup \mathit{efn}(\beta)$

The following lemma shows that *ensured free names* is indeed a reasonable name for this set, since a process satisfying a formula φ uses at least all ensured free names of φ as free names.

Lemma 2. *Let* P *be a process,* β *be a formula and* $b \in \mathcal{A}$*. Then* $P \models \beta$ *and* $b \in \mathit{efn}(\beta)$ *implies* $b \in \mathit{fn}(P)$*.*

Proof: By induction on the structure of formulae. ■

Now we can turn to the definition of translatable formulae, i.e. fragment and restriction formulae.

Definition 12 (Fragment and Restriction Formulae)**.** *Let* $b \in \mathcal{A}$*.* Fragment formulae ϕ *are defined by*

$$\phi ::= \top \mid \mathit{free}(b) \mid \neg\mathit{free}(b) \mid P^{\mathrm{seq}} \mid \mathrm{res}\ b\ (\phi_1 \parallel \ldots \parallel \phi_n),$$

where $\phi_i = \top$ *or* $b \in \mathit{efn}(\phi_i)$ *for each* $i \in \{1, \ldots, n\}$ *and there is at least one* $i \in \{1, \ldots, n\}$ *s.t.* $\phi_i \neq \top$*.* Restriction formulae *are defined as* $\rho ::= \phi_1 \parallel \ldots \parallel \phi_m$*.*

Observe that due to this definition, formulae of the form $\neg\mathit{free}(b)$ may not appear under any restriction. Furthermore, the only atoms satisfied by $\mathbf{0}$ are $\top$ and

$\neg free(b)$. Henceforth, we will call these atoms *segregative formulae*, as justified by Lemma 4. Proposition 2 shows the reason for calling the formulae ϕ fragment formulae. If a process P satisfies a fragment formula ϕ, the restricted form of P either contains a fragment F which already satisfies ϕ or ϕ is segregative. The second part of the proposition states that if P consists of more fragments than F, ϕ has to contain certain subformulae, which are satisfied by (almost) arbitrary processes.

Proposition 2. *Let ϕ be a fragment formula and P a process with $P \models \phi$. Then*

1) *there exists an F s.t. $P \equiv F \mid Q$ and $F \models \phi$ where $F \in \mathcal{P}^{\mathcal{F}}$ or $F \equiv \mathbf{0}$, and*
2) *if $Q \not\equiv \mathbf{0}$ then $\top \in sub(\phi)$, $free(b) \in sub(\phi)$ or $\neg free(b) \in sub(\phi)$.*

Proof: By induction on the structure of formulae. The base cases are immediate. In the induction step, the process has to be divided into fragments F_i satisfying the subformulae of ϕ and the remaining parts Q_i of the subprocesses. Let m be the name in the process corresponding to the restricted name b in ϕ. Then the scope of the restriction of m is shrunken to exclude all fragments F_i and processes Q_i not containing m free. The process under this shrink scope is a fragment and is satisfying ϕ. The second part of the proposition holds, since the only formulae allowing certain indeterminate processes for their satisfaction are $\top$, $free(b)$ and $\neg free(b)$. ∎

We use a function to compute a normal form of the translatable formulae. This function propagates $\top$ atoms beneath restrictions to the uppermost parallel composition, removes redundant $\top$ atoms, and, if both atoms of the form $\neg free(b)$ and $\top$ occur in the formula, it removes all occurrences of the former. This normal form allows for the following. Let $P \equiv \prod_{i\in I} F_i$ be a process in restricted form, i.e. all F_i are fragments, satisfying a normalised formula $\rho = \coprod_{j\in J} \phi_j$. By Proposition 2, we can find a mapping f from I to J, such that for each i, $F_i \models \phi_{f(j)}$. The parallel composition $R \equiv \prod_{i\in \bar{I}} F_i$, where $\bar{I} \subseteq I$ and each F_i is needed to satisfy a non-segregative formula ϕ_j, is a *minimal satisfying process* of ρ with respect to P (cf. Definition 14). Proposition 3 shows that each P with $P \models \rho$ consists of a minimal satisfying process in parallel to some process S. If $S \not\equiv \mathbf{0}$, Proposition 2 yields the existence of a subformula $\top$, $free(b)$ or $\neg free(b)$. The normalisation of ρ results in the existence of either $\top$ or $\neg free(b)$, i.e., segregative subformulae. Then all the fragments of S satisfy segregative subformulae of ρ. Hence, the occurrences of fragments in R define a lower bound on their occurrences in all P' satisfying ρ, where $P \to^* P'$.

Definition 13 (Normalisation Function)**.** *The* normalisation function *nf is defined inductively. Considering a formula $\beta = \coprod_{i\in I} \phi_i$, we use $\coprod_{j\in J} \psi_j$ to denote $\coprod_{i\in I} nf(\phi_i)$. The sets I and J may be different, due to the creation of new $\top$ atoms by the normalisation. Then $J_\top$ is defined by $J_\top = \{j \mid \psi_j = \top\}$ and $J_\neg = \{j \mid \psi_j = \neg free(b)$ for some name $b\}$.*

$$nf(\top) = \top \qquad nf(\neg free(b)) = \neg free(b)$$
$$nf(P^{\text{seq}}) = P^{\text{seq}} \qquad nf(free(b)) = free(b) \parallel \top$$

$$nf(\text{res } b\ (\coprod_{i\in I} \phi_i)) = \begin{cases} \top, \text{ if} \\ nf(\coprod_{i\in I} \phi_i) = \top \\ \top \parallel \text{res } b\ (nf(\coprod_{i\in I} \phi_i)), \text{ if} \\ nf(\coprod_{i\in I} \phi_i) \text{ contains } \top \\ \text{res } b\ (nf(\coprod_{i\in I} \phi_i)), \text{ otherwise} \end{cases}$$

$$nf(\coprod_{i\in I} \phi_i) = \begin{cases} \top, \text{ if} \\ J_\top \neq \emptyset \text{ and } J = J_\top \cup J_\neg \\ \top \parallel \coprod_{j\in J\setminus(J_\top \cup J_\neg)} \psi_j, \text{ if} \\ J_\top \neq \emptyset \text{ and } J \supset J_\top \cup J_\neg \\ \coprod_{j\in J} \psi_j, \text{ otherwise} \end{cases}$$

The normalisation of a formula does not change its truth value.

Lemma 3. *Let φ be a fragment or restriction formula. Then $\varphi \mathrel{\widehat{\equiv}} nf(\varphi)$.*

Proof: The lemma follows by induction on the structure of fragment and restriction formulae including several applications of the equivalences denoted in Proposition 1. The first equivalence allows for collapsing several parallel $\top$ atoms to one. The second justifies the base case of $free(b)$, while the equivalences (3) and (4) ensure the equivalence, when atoms of both forms $\top$ and $\neg free(b)$ are present. ∎

Example 4. *Consider the fragment formula $\beta =$ res y $(MS\lfloor y\rfloor \parallel \top)$. The normalisation of the parallel composition in the scope of the restriction yields $\top \parallel MS\lfloor y\rfloor$. Since this formula contains $\top$, the normalisation of β is $nf(\beta) = \top \parallel$ res y $(\top \parallel MS\lfloor y\rfloor)$, which is equivalent to β by Proposition 1 (4).*

Lemma 4. *Let φ be a normalised fragment or restriction formula. Then exactly one of the following properties holds:*

- *$\neg free(b) \notin sub(\varphi)$ for all b, or*
- *$\top \notin sub(\varphi)$ and $\neg free(b) \in sub(\varphi)$ for some b.*

Proof: The normalisation function never creates new formulae of the form $\neg free(b)$ for any b, hence if φ does not contain a formulae $\neg free(b)$, neither does its normalisation. On the other hand, if φ contains a subformula $\neg free(b)$, then it does not occur under any restriction. Thus, if φ contains a subformula $\top$, this will be propagated to the outermost level. Since the normalisation function removes occurrences of $\neg free(b)$ in parallel to $\top$, the normalised formula does not contain

formulae of type $\neg \mathit{free}(b)$. So a normalised formula may only contain $\neg \mathit{free}(b)$, if it does not contain $\top$. ∎

We now turn to the definition of minimal processes satisfying a restriction formula ρ with respect to a process P, denoted by $\mathcal{R}[\![\rho]\!]_P$. These processes shall be minimal in the following sense: A minimal process Q consists only of fragments, which are absolutely needed for satisfaction, i.e., segregative subformulae of ρ formulae are not considered. For the definition of this set, we need the satisfying fragments of a fragment formula ϕ, denoted by $\mathcal{F}[\![\phi]\!]_P$.

Definition 14 (Minimal Satisfying Processes)**.** *Let P be a process, ϕ and $\rho = \coprod_{i\in I} \phi_i$ be normalised fragment resp. restriction formulae. Furthermore, let $I_0 = \{i \mid \mathbf{0} \models \phi_i\}$ and $\bar{I} = I \setminus I_0$. The set of* satisfying fragments *with respect to P is given by the following definition.*

$$\mathcal{F}[\![\phi]\!]_P = \{\, [F] \mid [F] \in fg(rf(Reach(P)))_{/\equiv} \wedge F \models \phi \,\}$$

The set of minimal satisfying processes *is either $\mathcal{R}[\![\rho]\!]_P = \{\bot\}$, if there is an $i \in \bar{I}$ such that $\mathcal{F}[\![\phi_i]\!]_P = \emptyset$ or otherwise*

$$\mathcal{R}[\![\rho]\!]_P = \left\{ \left[\prod\nolimits_{i\in\bar{I}} F_i\right] \mid \forall i \in \bar{I}\colon [F_i] \in \mathcal{F}[\![\phi_i]\!]_P \right\}.$$

Observe that if $\bar{I} = \emptyset$, i.e., all subformulae of ρ are segregative, $\mathcal{R}[\![\rho]\!]_P = \{[\mathbf{0}]\}$. The following proposition is crucial for the proof of the well-definition of the structural translation. Every process satisfying a normalised restriction formula ρ consists of two parts: A process contained in the set of minimal satisfying process of ρ and a process whose fragments all satisfy segregative formulae, i.e. formulae of the form $\top$ or $\neg \mathit{free}(b)$.

Proposition 3. *Let P be a process, $Q \in Reach(P)$ and $\rho = \phi_1 \parallel \ldots \parallel \phi_n$ a normalised restriction formula. Then $Q \models \rho$ if and only if there are processes in restricted form R and S such that $rf(Q) \equiv R \,|\, S$ and the following holds:*

1) *$[R] \in \mathcal{R}[\![\phi_1 \parallel \ldots \parallel \phi_n]\!]_P$.*
2) *for all $F \in fg(S)$, there is a $k \in \{1, \ldots, n\}$ with $[F] \in \mathcal{F}[\![\phi_k]\!]_P$ and $\mathbf{0} \models \phi_k$.*

Proof: If $Q \models \rho$, we reorder $rf(Q)$ according to Proposition 2. The fragments F used for the satisfaction of formulae ϕ_i with $\mathbf{0} \not\models \phi_i$ are collected in the process R , while we accumulate the rest of $rf(Q)$ in S. Then $R \models \rho$ and hence $[R] \in \mathcal{R}[\![\rho]\!]_P$. By the second part of Proposition 2, we know that if $S \not\equiv \mathbf{0}$, there are subformulae of the form $\top$, $\mathit{free}(b)$ or $\neg \mathit{free}(b)$, and since ρ is normalised, the formulae $\top$ and $\neg \mathit{free}(b)$ occur on the outermost level of ρ. Hence, for all fragments F_S of S, we can find a suitable ϕ_k, such that $F_S \models \phi_k$, i.e. $[F_S] \in \mathcal{F}[\![\phi_k]\!]_P$.

The converse holds by similar arguments. ∎

For the translation function θ, we need a distinction between fragments that may occur unboundedly often for the satisfaction of a formula ρ and fragments, whose occurrences are exactly determined by ρ.

Definition 15 (Fixed and Flexible Fragments)**.** *Let $\rho = \phi_1 \parallel \ldots \parallel \phi_n$ be a normalised restriction formula and $I = \{\, i \mid \mathbf{0} \models \phi_i \,\}$. Then the set of* flexible fragments *is defined by $\mathrm{flex}(\rho, P) = \bigcup_{i\in I} \mathcal{F}[\![\phi_i]\!]_P$. Furthermore, we call $\mathrm{fix}(\rho, P) = \big(fg(rf(Reach(P)))_{/\equiv}\big) \setminus \mathrm{flex}(\rho, P)$ the set of* fixed fragments*.*

Example 5. *Let $\rho = nf(\beta) = \top \parallel \mathrm{res}\ y\ (\top \parallel MS\lfloor y \rfloor)$. Furthermore let $P \equiv System$ as in Example 1. The set of minimal satisfying fragments of $\top$ contains all reachable fragments of P. Since $\mathbf{0} \models \top$, all these fragments are flexible.*

If $\rho = \neg \mathit{free}(out)$, all fragments not using out as a free name are contained in $\mathcal{F}[\![\neg \mathit{free}(out)]\!]_P$. These fragments are flexible, while all fragments with a free occurrence of out are fixed. E.g, $[F']$ of Example 2 is flexible, while $[F]$ is fixed.

We are now in the position to define the translation function θ. The idea of translating a restriction formula ρ to LTL with respect to a process P is as follows. For each minimal process Q satisfying ρ, we create a conjunction of atomic LTL formulae α_i ranging over the reachable fragments F_i of P. If F is fixed, α_i expresses that F has to occur exactly as often as in Q, while α_i defines a lower bound on the occurrences of flexible fragments.

Definition 16 (Structural Translation)**.** *Let ρ be a normalised restriction formula and P be a process. The* structural translation of ρ with respect to P *is $[F] < 0$ for an arbitrary fragment of P, if $\mathcal{R}[\![\rho]\!]_P = \{\bot\}$, otherwise it is*

$$\theta(\rho, P) = \bigvee_{[Q]\in\mathcal{R}[\![\rho]\!]_P} \Big(\bigwedge_{[F]\in \mathrm{fix}(\rho,P)} [F] = dec(rf(Q))([F]) \wedge \bigwedge_{[F]\in \mathrm{flex}(\rho,P)} [F] \geq dec(rf(Q))([F]) \Big).$$

Hence, a structural formula is translated into a LTL formula which describes its satisfying markings. In particular, the LTL formula does not contain any temporal modalities.

Now we can give the main result of this section, the soundness of the translation. That is, a process P satisfies a restriction formula ρ iff the structural semantics of P satisfies the structural translation of ρ with respect to P.

Theorem 1. *Let P be a process and ρ a normalised restriction formula. Furthermore, let ω be an occurrence of $\mathcal{N}[\![P]\!]$. Then*

$$P \models \rho \text{ iff } \omega \models \theta(\rho, P).$$

Proof: We sketch the idea of the proof. If $P \models \rho$, then we know by Proposition 3, that we can decompose P into two processes in restricted form R and S, such that R is equivalent to a minimal satisfying process of ρ with respect to P, i.e. $[R] \in \mathcal{R}[\![\rho]\!]_P$. Furthermore, all fragments of S satisfy segregative subformulae of ρ, i.e. they are flexible fragments. Recall that the initial marking of the structural semantics is $dec(rf(P))$ and by Lemma 1 we know that $dec(rf(P)) = dec(R) + dec(S)$. By the definition of θ, there is a disjunct in the translation, where all atomic formulae are either of the form $[F] = dec(R)([F])$ for fixed or $[F] \geq dec(R)([F])$ for flexible fragments. Hence, these atoms are satisfied by $dec(rf(P))$.

For the converse direction, assume that the initial marking satisfies a disjunct of $\theta(\rho, P)$. Then we can reconstruct a corresponding process R due to full retrievability of the structural semantics by creating c_F-many fragments F for each atom $[F] \sim c_F$, where $\sim$ is $=$ or $\geq$. By the construction of $\theta(\rho, P)$, this is a minimal satisfying process of ρ with respect to P. Let $dec(rf(P))([F]) = d_F$. Then consider all fragments $[F]$, such that $d_F - c_F > 0$. These are all flexible fragments with which we can create the process S such that $R \,|\, S \equiv P$. Hence we apply Proposition 3 and get that $P \models \rho$. ∎

Example 6. *Let $P \equiv$ System as in Example 1 and $\rho = \top \parallel \text{res}\ y\ (\top \parallel MS\lfloor y \rfloor)$. As described in Example 5, all fragments are flexible with respect to ρ. The set of minimal satisfying processes contains 77 elements, all of which are single fragments. Hence the translation of ρ with respect to P consists of a disjunction of conjunctions γ of the following type: Let $[G]$ be the minimal satisfying process under consideration. Then $\gamma = [G] \geq 1$, where we omit trivially satisfied atomic formulae $[\bar{G}] \geq 0$. E.g., the fragment $[F']$ of Example 2 satisfies ρ, since one of its subprocesses is $\nu m.(BS_a\lfloor f_a, m \rfloor \mid MS\lfloor m \rfloor)$. Thus, the corresponding conjunct is $[F'] \geq 1$. On the whole, the relatively simple formula* $\text{res}\ y\ (MS\lfloor y \rfloor \parallel \top)$ *corresponds to a disjunction of 77 atomic formulae on the structural semantics with respect to the GSM handover protocol.*

IV. Translating Temporal Formulae

In this section we define the *π-Calculus structural-temporal logic (PSTL)*, which is in essence LTL where the atomic propositions are formulae of the translatable subset defined in the last section. That is, we create PSTL by adding a temporal dimension to translatable formulae along the lines of Finger and Gabbay [4].

Definition 17 (Syntax)**.** *The syntax of structural-temporal formulae is given by the following EBNF.*

$$\varphi ::= \rho \mid \neg\varphi_1 \mid \varphi_1 \wedge \varphi_2 \mid \Diamond\varphi_1,$$

where ρ is a restriction formula according to Definition 12. We denote the set of structural-temporal formulae by $\mathcal{PSTL}$.

The formulae are interpreted on sequences of process reactions.

Definition 18 (Process Sequence)**.** *Let $\pi = P_0 P_1 \ldots$ be an infinite sequence of processes such that $P_i \to P_{i+1}$. If there is some k such that P_k can not react anymore, we set $P_j = P_k$ for all $j \geq k$. Then π is called a* process sequence of P_0. *We denote the initial process P_0 of π by P_π. Similar to occurrence sequences, the subsequence of π starting at k is denoted by π^k.*

Remark 2. *Note that due to the isomorphic transition systems (see Remark 1), a process sequence of a process P corresponds to exactly one occurrence sequence of the structural semantics of P, i.e. we can lift the isomorphism of processes and markings to sequences.*

Definition 19 (Semantics)**.** *Let π be a process sequence. The semantics of PSTL formulae is defined inductively as follows.*

$\pi \models \rho$	*iff*	$P_\pi \models \rho$
$\pi \models \neg\varphi$	*iff*	*not* $\pi \models \varphi$
$\pi \models \varphi_1 \wedge \varphi_2$	*iff*	$\pi \models \varphi_1$ *and* $\pi \models \varphi_2$
$\pi \models \Diamond\varphi$	*iff*	$\exists y\colon \pi^y \models \varphi$

If $\pi \models \varphi$ we say that π satisfies *φ. A formula φ is* valid *for a process P, written $P \models \varphi$, if and only if all process sequences of P satisfy φ.*

The set of minimal satisfying processes of a formula ρ is decreasing monotonically with respect to the reachability of processes. I.e., if the process P' is reachable from P, the minimal satisfying processes with respect to P' are a subset of the minimal satisfying processes with respect to P.

Lemma 5. *Let P and P' be processes such that $P \to^* P'$ and ρ be a normalised restriction formula. Then $\mathcal{R}[\![\rho]\!]_{P'} \subseteq \mathcal{R}[\![\rho]\!]_P$ or $\mathcal{R}[\![\rho]\!]_{P'} = \{\bot\}$.*

Proof: By induction on the structure of restriction formulae. ∎

Definition 20. *The translation of $\Theta(\varphi, P)$ a PSTL formula φ with respect to a process P is defined inductively on the structure of φ as follows.*

$$\begin{aligned}
\Theta(\rho, P) &= \theta(\rho, P) \\
\Theta(\neg\varphi, P) &= \neg\Theta(\varphi, P) \\
\Theta(\varphi_1 \wedge \varphi_2, P) &= \Theta(\varphi_1, P) \wedge \Theta(\varphi_2, P) \\
\Theta(\Diamond\varphi, P) &= \Diamond\Theta(\varphi, P)
\end{aligned}$$

Consider processes P, P' such that $P \to^* P'$. The truth of the translation of a formula ρ with respect to the

structural semantics of P' is not affected, if we construct the translation with respect to P. This is due to the fact, that all minimal satisfying processes with respect to P' are contained in the set of minimal satisfying processes with respect to P, i.e. Lemma 5.

Lemma 6. *Let P and P' be processes such that $P \to^* P'$ and φ be a formula of PSTL. Furthermore let ω be an occurrence sequence of $\mathcal{N}[\![P']\!]$. Then*

$$\omega \models \Theta(\varphi, P') \text{ iff } \omega \models \Theta(\varphi, P).$$

Proof: By induction on the structure of formulae. The "if" direction of the base case holds due to Lemma 5, while the "only if" direction holds, since the initial marking of ω can only reference places of the structural semantics of P'. The induction steps are simple applications of the induction hypothesis. ∎

The next lemma states that satisfaction of a formula with respect to a state in a process sequence and satisfaction of the translated formula with respect to the marking in the associated occurrence sequence is equivalent.

Lemma 7. *Let P be a process, π be a process sequence of P and g an isomorphism from process sequences to occurrence sequences (see Remark 2). Then*

$$\pi \models \varphi \text{ iff } g(\pi) \models \Theta(\varphi, P)$$

Proof: Let $\pi \models \varphi$. We proceed by induction on the structure of φ. The base case is immediate by an application of Theorem 1. Most of the induction steps are immediate consequences of the semantics of LTL and PSTL as well as the induction hypothesis. The only interesting case involves the temporal modality. Assume the lemma holds for φ. Then

$$\begin{aligned}
\pi \models \Diamond\varphi &\Leftrightarrow \exists y\colon \pi^y \models \varphi \\
\{\text{Let } \pi' = \pi^y\} &\Leftrightarrow \pi' \models \varphi \\
\{\text{IH}\} &\Leftrightarrow g(\pi') \models \theta(\varphi, P_{\pi'}) \\
\{\text{Lemma 6}\} &\Leftrightarrow g(\pi') \models \theta(\varphi, P_\pi) \\
\{\pi' = \pi^y \text{ and } P_\pi = P\} &\Leftrightarrow \exists y\colon g(\pi^y) \models \Theta(\varphi, P) \\
\{\text{Semantics and Definition 20}\} &\Leftrightarrow g(\pi) \models \Theta(\Diamond\varphi, P)
\end{aligned}$$

∎

The main theorem of this section is a direct consequence of Lemma 7.

Theorem 2. *Let $P \in \mathcal{P}$ and $\varphi \in \mathcal{PSTL}$. Then*

$$P \models \varphi \text{ iff } \mathcal{N}[\![P]\!] \models \Theta(\varphi, P).$$

Example 7. *Consider β and P as in Example 6. Now let $\varphi = \Box\Diamond\beta$, where $\Box$ is the dual of $\Diamond$. The translation of φ with respect to P is $\Theta(\Box\Diamond\beta, P) = \Box\Diamond\theta(\beta, P)$. All in all, φ expresses that the mobile station is able to return to its initial state after all possible reductions of the system. I.e., the system is deadlock-free.*

V. Conclusion

We have shown how structural formulae defined on π-Calculus processes can be translated into equivalent propositional formulae on Petri nets with respect to the structural semantics as defined by Meyer. This gives us the ability to verify systems and properties defined as processes, resp. formulae on these processes with the help of standard Petri net techniques. The application to the GSM handover protocol exemplifies that simple structural properties of processes may correspond to rather large LTL formulae on the structural semantics. Hence, using PSTL formulae and the presented translation is much more convenient than manually identifying all places of interest. The size of the formulae increases exponentially in the size of the process under consideration, since the structural semantics already grows exponentially. The translation is sensible for *structurally stationary* processes, i.e. processes where the set of reachable fragments and hence the set of places in the structural semantics is finite, otherwise the LTL formulae would consist of infinitely many conjunctions.

The translation still relies on directly checking the satisfaction of formulae by fragments in the computation of minimal satisfying processes. But this check is much simpler than checking satisfaction of arbitrary formulae and processes, since only the structure of processes has to be taken into account. The current approach has been prototypically implemented in the Petruchio tool [10]. If a formula $\Diamond\rho$ contains $\top$, the minimal satisfying processes describe an upward closed set, i.e. deciding whether $\Diamond\rho$ holds is an instance of the covering problem for Petri nets [12]. A suitable backward coverability checker is implemented in Petruchio [13]. In general, processes can be effectively checked with respect to PSTL properties.

In the following, we will use the terms structural and spatial synonymically. Different spatio-temporal logics for process calculi have been proposed. The initial approach of Caires and Cardelli [3] has been proven decidable for bounded processes [5]. This set is a subset of structurally stationary processes [2].

Pattinson and Reus presented a proof system for linear spatial temporal logic $\mathcal{LSTL}$ [14], a combination of hybrid logic and LTL. They also follow Finger and Gabbay [4], but add constraints to keep track of names while processes react. Using hybrid logic enables for referring to different reachable processes directly.

Conforti et al. have defined the spatial logic BiLog on bigraphs [15]. BiLog is more expressive than PSTL, since bigraphs are a general concept for describing structure and mobility, subsuming e.g. π-Calculus [7], [8], Ambient Calculus [16] and CSP [17]. Similar to our approach, BiLog contains constants for bigraphs. In special cases,

satisfaction of these constants coincides with structural congruence. However, we are not aware of work on automated verification for bigraphs.

Partial order reductions for spatial properties have been investigated by Affeldt and Kobayashi [18]. They define three subsets of spatial logic with increasing expressiveness, all of which allow for the application of partial order techniques. However, the subsets lack a treatment of restrictions on names.

Acciai and Boreale have investigated decidability of spatial properties using a typing system [19]. In their work, CCS types abstract π-Calculus processes. Since CCS processes possess well-structured transition systems, safety is decidable.

Future Work: The translatable subset of formulae can be extended in various ways. The atom $\neg \mathit{free}(b)$ may be extended to sequences of names $\neg \mathit{free}(\tilde{b})$, without affecting any proofs given in this paper. Such an easy extension is not possible for the dual atom $\mathit{free}(b)$. For example, $\alpha = \mathit{free}(b, c)$ does no longer ensure that a single fragment satisfies α, contradicting Proposition 2. Consider $P \equiv \overline{b}\langle x\rangle \mid \overline{c}\langle y\rangle$. Both fragments are needed for the satisfaction of α. Interpreting α as a disjunction, i.e. $\mathit{free}(b) \vee \mathit{free}(c)$ is not an option either, since we can no longer ensure that b and c occur free in all processes satisfying α, as stated in Lemma 2. Furthermore, disjunction distributes always over the parallel modality, while conjunction does not. Hence, allowing for arbitrary conjunctions, disjunctions or negations in the translatable structural set would spawn a more complicated normal form, if there is any.

Enhancing the structural semantics with the possibility to track bound names creates the need to reflect such properties in the structural subset. That is, bound names in different structural formulae have to be associated. Techniques developed by Pattinson and Reus [14] may be useful for further investigation.

The selection of a particular subset of LTL for the temporal aspect is not of importance for the described translation. The extension to LTL with the next- or until-modality is straightforward.

Acknowledgment

The author would like to thank Sören Jeserich for implementing this approach as a part of his minor thesis.

References

[1] R. Meyer, "A theory of structural stationarity in the π-calculus," *Acta Inf.*, vol. 46, no. 2, 2009.

[2] ——, "Structural stationarity in the π-calculus," Ph.D. dissertation, Department of Computing Science, University of Oldenburg, 2009, http://concurrency.cs.uni-kl.de/documents/phd_final_2009.pdf.

[3] L. Caires and L. Cardelli, "A spatial logic for concurrency (part I)," in *TACS 2001*. Springer, Heidelberg, 2001.

[4] M. Finger and D. M. Gabbay, "Adding a temporal dimension to a logic system," in *Temporal logic (vol. 1): mathematical foundations and computational aspects*, D. M. Gabbay, I. Hodkinson, and M. Reynolds, Eds. Oxford University Press, 1994.

[5] L. Caires, "Behavioral and spatial observations in a logic for the π-calculus," in *FoSSaCS 2004*. Springer, Heidelberg, 2004.

[6] R. M. Amadio and C. Meyssonnier, "On decidability of the control reachability problem in the asynchronous π-calculus," *Nordic J. of Computing*, vol. 9, pp. 70–101, 2002.

[7] R. Milner, *Communicating and mobile systems: the π-calculus*. Cambridge University Press, 1999.

[8] D. Sangiorgi and D. Walker, *The π-Calculus: A Theory of Mobile Processes*. New York, NY, USA: Cambridge University Press, 2001.

[9] F. Orava and J. Parrow, "An algebraic verification of a mobile network," *FAC*, vol. 4, pp. 497–543, 1992.

[10] R. Meyer and T. Strazny, "Petruchio: From dynamic networks to nets," in *CAV 2010*, T. Touili, B. Cook, and P. Jackson, Eds. Springer, Heidelberg, 2010.

[11] S. R. Kosaraju, "Decidability of reachability in vector addition systems," in *STOC 1982*. New York, NY, USA: ACM, 1982, pp. 267–281.

[12] A. Finkel and P. Schnoebelen, "Well-structured transition systems everywhere!" *TCS*, vol. 256, no. 1-2, 2001.

[13] T. Strazny, "Accelerating backward reachability analysis," in *NWPT 2011*, P. Pettersson and C. Seceleanu, Eds. Mälardalen University Sweden, October 2011, pp. 2–4.

[14] D. Pattinson and B. Reus, "A complete temporal and spatial logic for distributed systems," in *FroCoS 2005*. Springer, Heidelberg, 2005.

[15] G. Conforti, D. Macedonio, and V. Sassone, "Spatial logics for bigraphs," in *ICALP 2005*. Springer, Heidelberg, 2005.

[16] L. Cardelli and A. Gordon, "Mobile ambients," in *FoSSaCS 1998*. Springer, Heidelberg, 1998.

[17] C. A. R. Hoare, "Communicating sequential processes," *Commun. ACM*, vol. 21, August 1978.

[18] R. Affeldt and N. Kobayashi, "Partial order reduction for verification of spatial properties of pi-calculus processes." *Electr. Notes Theor. Comput. Sci.*, 2005.

[19] L. Acciai and M. Boreale, "Deciding safety properties in infinite-state pi-calculus via behavioural types," in *ICALP 2009*. Springer, Heidelberg, 2009.

Appendix

We present both the formal specification of Example 1 as well as full proofs for most results.

A. Full Specification of Example 1

The system consists of four main agents. The active and passive base stations BS_a and BS_p, the mobile station MS and the mobile switching center MSC. The latter consists of two controllers, the handover controller HC and the communication controller CC. For a complete description see Orava and Parrow [9].

$$
\begin{aligned}
CC(f_a, f_p, l) &:= in(v).\overline{f_a}\langle data\rangle.\overline{f_a}\langle v\rangle.CC\lfloor f_a, f_p, l\rfloor \\
&\quad + l(m_{new}).\overline{f_a}\langle ho\_cmd\rangle.\overline{f_a}\langle m_{new}\rangle. \\
&\quad \Big(f_p(x).[x = ho\_cmd]\overline{f_a}\langle ch\_rel\rangle. \\
&\quad\quad f_a(m_{old}).\bar{l}\langle m_{old}\rangle.CC\lfloor f_p, f_a, l\rfloor \\
&\quad + f_a(x).[x = ho\_fail]. \\
&\quad\quad \bar{l}\langle m_{new}\rangle.CC\lfloor f_a, f_p, l\rfloor\Big) \\
HC(l, m) &:= \bar{l}\langle m\rangle.l(m).HC\lfloor l, m\rfloor \\
MSC(f_a, f_p, m) &:= \nu l.(HC\lfloor l, m\rfloor \mid CC\lfloor f_a, f_p, l\rfloor) \\
BS_a(f, m) &:= m(x). \\
&\quad ([x = data].f(v).\overline{m}\langle data\rangle.\overline{m}\langle v\rangle.BS_a\lfloor f, m\rfloor \\
&\quad + [x = ho\_cmd].f(v).\overline{m}\langle ho\_cmd\rangle.\overline{m}\langle v\rangle. \\
&\quad \Big(f(w).[w = ch\_rel].\overline{f}\langle m\rangle.BS_p\lfloor f, m\rfloor \\
&\quad + m(w).[w = ho\_fail]. \\
&\quad\quad \overline{f}\langle ho\_fail\rangle.BS_a\lfloor f, m\rfloor\Big) \\
BS_p(f, m) &:= m(x).[x = ho\_acc]. \\
&\quad \overline{f}\langle ho\_com\rangle.BS_a\lfloor f, m\rfloor \\
MS(m) &:= m(x).\ ([x = data].m(v).\overline{out}\langle v\rangle.MS\lfloor m\rfloor \\
&\quad + [x = ho\_cmd].m(m_{new}). \\
&\quad (\overline{m_{new}}\langle ho\_acc\rangle.MS\lfloor m_{new}\rfloor \\
&\quad + \overline{m}\langle ho\_fail\rangle.MS\lfloor m\rfloor)) \\
P(f_a, f_p) &:= \nu m.(MSC\lfloor f_a, f_p, m\rfloor \mid BS_p\lfloor f_p, m\rfloor) \\
Q(f_a) &:= \nu m.(BS_a\lfloor f_a, m\rfloor \mid MS\lfloor m\rfloor) \\
S &:= \nu v.\overline{in}\langle v\rangle.S \\
R &:= out(d).R \\
System &:= \nu f_a, f_p.(P\lfloor f_a, f_p\rfloor \mid Q\lfloor f_a\rfloor \mid S \mid R)
\end{aligned}
$$

B. Proposition 2

Let ϕ be a fragment formula and P a process with $P \models \phi$. Then

1) there exists an F s.t. $P \equiv F \mid Q$ and $F \models \phi$ where $F \in \mathcal{P}^{\mathcal{F}}$ or $F \equiv \mathbf{0}$, and
2) if $Q \not\equiv \mathbf{0}$ then $\top \in sub(\phi) \vee free(b) \in sub(\phi) \vee \neg free(b) \in sub(\phi)$.

Proof: By induction on the structure of formulae. If $\phi = \top$, the proposition holds trivially. If $\phi = free(b)$, let $rf(P) = \prod_{i\in I} F_i$. Since $b \in fn(P)$, there is a $j \in I$ such that $b \in fn(F_j)$. Hence $P \equiv F_j \mid \prod_{i\in I\setminus\{j\}} F_i$ and $F_j \in \mathcal{P}^{\mathcal{F}}$ and $F_j \models \phi$. Now let $\phi = \neg free(b)$ and $rf(P) = \prod_{i\in I} F_i$. Since $b \notin fn(P)$, there is no $j \in I$ such that $b \in fn(F_j)$. Choose an arbitrary $j \in I$, then $P \equiv F_j \mid \prod_{i\in I\setminus\{j\}} F_i$ and $F_j \in \mathcal{P}^{\mathcal{F}}$ and $F_j \models \phi$. If $P \equiv \mathbf{0}$, then $\mathbf{0} \models \neg free(b)$ and $P \equiv \mathbf{0} \mid \mathbf{0}$. In all these three cases, the second part of the proposition holds. If $\phi = P^{\text{seq}}$, then $P \equiv P^{\text{seq}}$, hence $P \in \mathcal{P}^{\mathcal{F}}$. Since $P \equiv P \mid \mathbf{0}$, the second part holds as well.

For the induction step, let $\phi = \text{res } b\ (\phi_1 \parallel \ldots \parallel \phi_n)$. Then there exist m and P' such that $P \equiv \nu m.P'$ with $m \notin fn(P)$ and $m \notin (fn(\phi_1) \cup \cdots \cup fn(\phi_n))$ and $P' \models (\phi_1 \parallel \ldots \parallel \phi_n)\{b \leftarrow m\}$.

Hence there are P_1 to P_n such that $P' \equiv P_1 \mid \ldots \mid P_n$ and $P_i \models \phi_i\{b \leftarrow m\}$. By the induction hypothesis, $P_i \equiv F_i \mid Q_i$ where $F_i \in \mathcal{P}^{\mathcal{F}} \vee F_i \equiv \mathbf{0}$ and $F_i \models \phi_i\{b \leftarrow m\}$. Now let $rf(Q_i) = \prod_{j\in J_i} R_{i,j}$ and $\{1, \ldots, n\} = I$, $I^m = \{i \mid m \in fn(F_i)\}$ and $J_i^m = \{j \mid m \in fn(R_{i,j})\}$.

Using this notation, we have

$$
\begin{aligned}
P_1 \mid \ldots \mid P_n &\equiv \prod_{i\in I} F_i \mid \prod_{i\in I}\prod_{j\in J_i} R_{i,j} \\
&\equiv \prod_{i\in I^m} F_i \mid \prod_{i\in I}\prod_{j\in J_i^m} R_{i,j} \mid \prod_{i\in I\setminus I^m} F_i \mid \prod_{i\in I}\prod_{j\in J_i\setminus J_i^m} R_{i,j}.
\end{aligned}
$$

Observe that the first part of this process contains m as a free name, while the latter does not. Furthermore, if $i \notin I^m$, we have that $m \notin fn(F_i)$, which by Lemma 2 implies $m \notin efn(\phi_i\{b \leftarrow m\})$, hence $\phi_i = \top$. In addition, we have by the second part of the induction hypothesis that if there are i and j such that $R_{i,j} \not\equiv \mathbf{0}$, then $\top \in sub(\phi_i)$ or $free(c) \in sub(\phi)$ (since $\neg free(c)$ is not allowed by the syntax). In this case, due to equivalences (2) and (4) of Proposition 1, we get that $\phi_1\{b \leftarrow m\} \parallel \ldots \parallel \phi_n\{b \leftarrow m\} \mathrel{\widehat{\equiv}} \phi_1\{b \leftarrow m\} \parallel \ldots \parallel \phi_n\{b \leftarrow m\} \parallel \top$ All in all $\prod_{i\in I^m} F_i \mid \prod_{i\in I}\prod_{i\in J_i^m} R_{i,j} \models \phi_1\{b \leftarrow m\} \parallel \ldots \parallel \phi_n\{b \leftarrow m\}$, hence also $F' \equiv \nu m.\left(\prod_{i\in I^m} F_i \mid \prod_{i\in I}\prod_{i\in J_i^m} R_{i,j}\right) \models \text{res } b\ (\phi_1 \parallel \ldots \parallel \phi_n)$, where F' is a fragment. By scope extrusion $P \equiv F' \mid \prod_{i\in I\setminus I^m} F_i \mid \prod_{i\in I}\prod_{j\in J_i\setminus J_i^m} R_{i,j}$. Observe that if there is an $i \in I \setminus I^m$ or there are i and j such that $j \in J_i \setminus J_i^m$, the second part of the proposition holds by the remarks above. ∎

C. Lemma 3

Let φ be a fragment or restriction formula. Then $\varphi \mathrel{\widehat{\equiv}} nf(\varphi)$.

Proof: By induction on the structure of formulae. First we consider fragment formulae, i.e. $\varphi = \phi$. If ϕ is either $\top$, $\neg free(b)$ or P^{seq} the statement follows

immediately. If $\phi = \mathit{free}(b)$, we have by Proposition 1 (2) that $\phi = \mathit{free}(b) \mathrel{\widehat{\equiv}} \mathit{free}(b) \parallel \top = \mathit{nf}(\phi)$.

For the induction step, assume that for $i \in I$ the statement $\phi_i \mathrel{\widehat{\equiv}} \mathit{nf}(\phi_i)$ holds. Let $\phi = \text{res } b\ (\coprod_{i\in I} \phi_i)$. By the induction hypothesis, $\coprod_{i\in I} \phi_i \mathrel{\widehat{\equiv}} \coprod_{i\in I} \mathit{nf}(\phi_i) = \coprod_{j\in J} \psi_j$.

1) If there is no $j \in J$ such that $\psi_j = \top$, then also $\mathit{nf}(\coprod_{i\in I} \phi_i) = \coprod_{j\in J} \psi_j$, i.e.,

$$\begin{aligned}\mathit{nf}(\text{res } b\ (\coprod_{i\in I} \phi_i)) &= \text{res } b\ (\mathit{nf}(\coprod_{i\in I} \phi_i)) \\ &= \text{res } b\ (\coprod_{j\in J} \psi_j) \mathrel{\widehat{\equiv}} \text{res } b\ (\coprod_{i\in I} \phi_i)\end{aligned}$$

2) If there is at least one $j \in J$ such that $\psi_j = \top$, then $\mathit{nf}(\coprod_{i\in I} \phi_i) = \top \parallel \coprod_{j\in J\setminus(J_\top \cup J_\neg)} \psi_j$. For each $j' \in J_\top \cup J_\neg$, we get by Proposition 1 (1) and (3) that $\psi_{j'} \parallel \top \mathrel{\widehat{\equiv}} \top$ and hence $\coprod_{j\in J} \psi_j \mathrel{\widehat{\equiv}} \top \parallel \coprod_{j\in J\setminus(J_\top \cup J_\neg)} \psi_j$. That is,

$$\begin{aligned}\mathit{nf}(\text{res } b\ (\coprod_{i\in I} \phi_i)) &= \text{res } b\ (\top \parallel \coprod_{j\in J\setminus(J_\top \cup J_\neg)} \psi_j) \parallel \top \\ &\mathrel{\widehat{\equiv}} \text{res } b\ (\top \parallel \coprod_{j\in J\setminus(J_\top \cup J_\neg)} \psi_j) \\ &\mathrel{\widehat{\equiv}} \text{res } b\ (\coprod_{j\in J} \psi_j) \mathrel{\widehat{\equiv}} \text{res } b\ (\coprod_{i\in I} \phi_i),\end{aligned}$$

where the first equivalence is justified by Proposition 1 (4).

The case $\mathit{nf}(\text{res } b\ (\coprod_{i\in I} \phi_i)) = \top$ is a special case of the above.

The case of restriction formulae, i.e. $\varphi = \coprod_{i\in I} \phi_i$, is treated similar to the previous case. ∎

D. Lemma 4

Let φ be a normalised fragment or restriction formula. Then exactly one of the following properties holds:

- $\neg\mathit{free}(b) \notin \mathit{sub}(\varphi)$ for all b, or
- $\top \notin \mathit{sub}(\varphi)$ and $\neg\mathit{free}(b) \in \mathit{sub}(\varphi)$ for some b.

Proof: If φ is an atomic fragment formula, the lemma follows immediatly. Let $\varphi = \text{res } c\ (\coprod_{i\in I} \phi_i)$.

1) For no $i \in I$ can ϕ_i be a formula of the form $\neg\mathit{free}(b)$, since this formula possesses an empty set of ensured free names and hence may not occur under an restriction. Due to the same argument, no ϕ_i *contains* such a formula.
2) Let φ contain $\top$ or $\mathit{free}(b)$. The application of nf does not create any formula of the type $\neg\mathit{free}(d)$. Hence $\mathit{nf}(\varphi)$ does not contain a subformula $\neg\mathit{free}(d)$.

Let $\varphi = \coprod_{i\in I} \phi_i$. If φ possesses $\neg\mathit{free}(b)$ as a subformula, there is an $k \in I$ such that $\phi_k = \neg\mathit{free}(b)$. Now let there be a $l \in I$ such that ϕ_l contains $\top$, i.e. there is a ψ, such that $\mathit{nf}(\phi_l) = \psi \parallel \top$. Consider $\coprod_{i\in I} \mathit{nf}(\phi_i) = \coprod_{j\in J} \psi_j$. The formula ϕ_k is still a subformula of this normalised formula, i.e. there is a k' such that $\phi_k = \psi_{k'}$ and $k' \in J_\neg$. Furthermore, there is a formula $\top$ in this normalised formula (the one created due to ϕ_l containing $\top$), hence $\mathit{nf}(\varphi) = \top \parallel \coprod_{j\in J\setminus(J_\top \cup J_\neg)} \psi_j$. Now all formulae of the form $\neg\mathit{free}(b)$ have been removed, hence the lemma holds. Again, $\mathit{nf}(\varphi) = \top$ is a special case. ∎

E. Proposition 3

Let P be a process, $Q \in \mathit{Reach}(P)$ and $\rho = \phi_1 \parallel \ldots \parallel \phi_n$ a normalised restriction formula. Then $Q \models \rho$ if and only if there are processes in restricted form R and S such that $\mathit{rf}(Q) \equiv R \mid S$ and the following holds:

1) $[R] \in \mathcal{R}\,[\![\phi_1 \parallel \ldots \parallel \phi_n]\!]_P$.
2) for all $F \in \mathit{fg}(S)$, there is a $k \in \{1, \ldots, n\}$ with $[F] \in \mathcal{F}\,[\![\phi_k]\!]_P$ and $\mathbf{0} \models \phi_k$.

Proof: Let $Q \models \phi_1 \parallel \ldots \parallel \phi_n$ and $I = \{1, \ldots, n\}$. Then $Q \equiv \prod_{i\in I} Q_i$ such that $Q_i \models \phi_i$ for all $i \in I$. By Proposition 2 1) we know that for every i there are processes F_i and P_i, such that $Q_i \equiv F_i \mid P_i$, where $F_i \models \phi_i$ and either F_i is a fragment or $F_i \equiv \mathbf{0}$. Now let $I_0 = \{i \mid F_i \equiv \mathbf{0}\}$. Then for all $i \in I \setminus I_0$, we have $[F_i] \in \mathcal{F}\,[\![\phi_i]\!]_P$ and hence $\left[\prod_{i\in I\setminus I_0} F_i\right] \in \mathcal{R}\left[\!\!\left[\coprod_{i\in I\setminus I_0} \phi_i\right]\!\!\right]_P$. Now let $[R] = \left[\prod_{i\in I\setminus I_0} F_i\right]$. Because each ϕ_i with $i \in I_0$ is satisfied by the deadlock process $\mathbf{0} \equiv F_i$, we even have $[R] = \left[\prod_{i\in I} F_i\right] \in \mathcal{R}\,[\![\coprod_{i\in I} \phi_i]\!]_P$.

Since parallel composition is commutative, we have $Q \equiv R \mid S'$ with $R \equiv \prod_{i\in I} F_i$ and $S' \equiv \prod_{i\in I} P_i$. Now consider one $P_i \equiv \mathit{rf}(P_i) = \prod_{j\in J} G_j \not\equiv \mathbf{0}$. Then we still have $F_i \mid \prod_{j\in J} G_j \models \phi_i$ since this process is structurally congruent to Q_i. Hence by Proposition 2 2), we have $\top \in \mathit{sub}(\phi_i)$, $\mathit{free}(b) \in \mathit{sub}(\phi_i)$ or $\neg\mathit{free}(b) \in \mathit{sub}(\phi_i)$ for some name b.

1) If $\top \in \mathit{sub}(\phi_i)$ or $\mathit{free}(b) \in \mathit{sub}(\phi_i)$, then due to the normalisation of $\coprod_{i\in I} \phi_i$, there is a $k \in I$ such that $\phi_k = \top$. Then for all $j \in J$, we have $[G_j] \in \mathcal{F}\,[\![\phi_k]\!]_P$.
2) Let $\neg\mathit{free}(b) \in \mathit{sub}(\phi_i)$. Due to the syntax of fragment formulae, $\neg\mathit{free}(b)$ may not appear under a restriction on a name, hence $\phi_i = \neg\mathit{free}(b)$. Since $Q_i \models \phi_i$ also $b \notin \mathit{fn}(\prod_{j\in J} G_j)$. Hence for each $j \in J$, $G_j \models \phi_i$ holds, i.e. $[G_j] \in \mathcal{F}\,[\![\phi_i]\!]_P$.

With $S = \mathit{rf}(S')$, we have shown that $[R]$ is a minimal satisfying process class of $\coprod_{i\in I} \phi_i$ and all fragments of S are used for the satisfaction of fragment formulae ϕ_i with $\mathbf{0} \models \phi_i$.

For the converse, assume that there are R and S such that both conditions of the proposition hold. By the definition of $\mathcal{R}\,[\![\cdot]\!]_P$ we get that $R \models \coprod_{i\in I} \phi_i$. Now we consider an arbitrary $F \in \mathit{fg}(S)$. Then there is a formula ϕ_k such that $F \models \phi_k$ and $\mathbf{0} \models \phi_k$, i.e. $\phi_k = \top$ or $\phi_k = \neg\mathit{free}(b)$ for some name a. Note that R has to be decomposable into $R_1, \ldots, R_n$ such that $R_i \models \phi_i$

($n = |I|$ and $1 \leq i \leq n$). Now if $\phi_k = \top$ obviously $R_k \,|\, F \models \phi_k$. The same holds for $\phi_k = \neg \mathit{free}(b)$. Hence

$$R_1 \,|\, \ldots \,|\, R_k \,|\, F \,|\, \ldots \,|\, R_n \models \coprod_{i \in I} \phi_i$$

This argument holds for all $F \in \mathit{fg}(S)$, i.e. all fragments of S can be composed in parallel to subprocesses of R to obtain $Q' \equiv R \,|\, S$ such that $Q' \models \coprod_{i \in I} \phi_i$. Since $Q' \equiv R \,|\, S \equiv \mathit{rf}(Q) \equiv Q$ we get $Q \models \coprod_{i \in I} \phi_i$. ∎

F. Theorem 1

Let P be a process and ρ a normalised restriction formula. Furthermore, let ω be an occurrence sequence of $\mathcal{N}[\![P]\!]$. Then

$$P \models \rho \text{ iff } \omega \models \theta(\rho, P).$$

Proof: Let $P \models \rho$. Then by Proposition 3, there are R and S in restriction form such that $\mathit{rf}(P) \equiv R \,|\, S$, $[R] \in \mathcal{R}[\![\rho]\!]_P$ and for all fragments F of S, there is a $k \in I$ such that $[F] \in \mathcal{F}[\![\phi_k]\!]_P$ and $\mathbf{0} \models \phi_k$. For simplicity, we distinguish the three cases for ρ as implied by Lemma 4.

1) If ρ does not contain $\top$ or $\neg \mathit{free}(b)$, then $S \equiv \mathbf{0}$, i.e. $P \equiv R$. Hence $[P] \in \mathcal{R}[\![\rho]\!]_P$. Since $\mathrm{flex}(\rho, P) = \emptyset$ in this case, $\theta(\rho, P)$ contains a disjunct ψ, such that $\psi = \bigwedge_{[F] \in \mathit{fg}(\mathit{rf}(\mathit{Reach}(P)))/\equiv} [F] = \mathit{dec}(\mathit{rf}(P))([F])$. Since the initial marking of $\mathcal{N}[\![P]\!]$ is defined as $M_0 = \mathit{dec}(\mathit{rf}(P))$, we have $\omega \models \psi$ and hence $\omega \models \theta(\rho, P)$.
2) Let ρ contain $\top$. First observe that all reachable fragments of P are flexible in this case. If $S \equiv \mathbf{0}$, the argument is the same as in case 1. Hence let $S \not\equiv \mathbf{0}$. Again there is a disjunct $\psi = \bigwedge_{[F] \in \mathrm{flex}(\rho, P)} [F] \geq \mathit{dec}(\mathit{rf}(R))([F])$. Since all fragments of S are also reachable fragments of $\mathit{rf}(P)$, we have by Lemma 1 $\mathit{dec}(\mathit{rf}(P))([F]) = \mathit{dec}(\mathit{rf}(R))([F]) + \mathit{dec}(\mathit{rf}(S))([F]) \geq \mathit{dec}(\mathit{rf}(R))([F])$ for all fragments F of P. Hence $\omega \models \psi$, and in particular $\omega \models \theta(\rho, P)$.
3) As the last case, let ρ contain at least one formula of type $\neg \mathit{free}(b)$. Then in general neither $\mathrm{flex}(\rho, P) = \emptyset$ nor $\mathrm{fix}(\rho, P) = \emptyset$. Furthermore let $S \not\equiv \mathbf{0}$ (otherwise proceed as in case 1). Again, there is a disjunct ψ such that

$$\psi = \bigwedge_{[F] \in \mathrm{fix}(\rho, P)} [F] = \mathit{dec}(\mathit{rf}(R))([F]) \wedge \bigwedge_{[F] \in \mathrm{flex}(\rho, P)} [F] \geq \mathit{dec}(\mathit{rf}(R))([F]).$$

Observe that ρ does not contain any $\top$, i.e., all fragments of S must satisfy a formula of the form $\neg \mathit{free}(b)$ for some name b (see Proposition 2. 2) and Lemma 4). Hence all equivalence classes of fragments of S are elements of $\mathrm{flex}(\rho, P)$. The initial marking of $\mathcal{N}[\![P]\!]$ is defined by $\mathit{dec}(\mathit{rf}(P)) = \mathit{dec}(R) + \mathit{dec}(S)$. That is, for all $[F] \notin \mathrm{flex}(\rho, P)$, $\mathit{dec}(\mathit{rf}(P))([F]) = \mathit{dec}(R)([F])$ holds and for all $[F] \in \mathrm{flex}(\rho, P)$, we have $\mathit{dec}(\mathit{rf}(P))([F]) = \mathit{dec}(R)([F]) + \mathit{dec}(S)([F]) \geq \mathit{dec}(R)([F])$. Altogether, $\omega \models \psi$, i.e., $\omega \models \theta(\rho, P)$.

Now assume that $\omega \models \theta(\rho, P)$. Since $\theta(\rho, P)$ is in disjunctive normal form, there is a disjunct ψ such that $\omega \models \psi$. In the following, note that M_ω is the initial marking of ω.

1) Let ρ be a formula without $\top$ and $\neg \mathit{free}(b)$. Then $\psi = \bigwedge_{[F] \in \mathrm{fix}(\rho, P)} [F] = c_F$. That is, $M_\omega([F]) = c_F$ for all reachable fragments of P. By construction of $\theta(\rho, P)$, there has to be a process Q such that $[Q] \in \mathcal{R}[\![\rho]\!]_P$ and $\mathit{dec}(\mathit{rf}(Q)) = M_\omega = \mathit{dec}(\mathit{rf}(P))$. By Lemma 1, $Q \equiv P$. Hence $[P] \in \mathcal{R}[\![\rho]\!]_P$ and by Proposition 3 we get $P \models \rho$.
2) Let ρ contain $\top$, i.e. for $\rho = \coprod_{i \in I} \phi_i$ there is a k such that $\phi_k = \top$, since ρ is normalised. Furthermore, ψ is of the form $\psi = \bigwedge_{[F] \in \mathrm{flex}(\rho, P)} [F] \geq c_F$. By the construction of $\theta(\rho, P)$, we get $\left[\prod_{[F] \in \mathit{supp}(M_\omega)} \prod^{c_F} F\right] \in \mathcal{R}[\![\rho]\!]_P$. Hence $P \equiv$

$$\prod_{[F] \in \mathit{supp}(M_\omega)} \prod^{c_F} F \,|\, \prod_{[F] \in \mathit{supp}(M_\omega)} \prod^{M_\omega([F]) - c_F} F$$

$= R \,|\, S$. For each fragment F of S, we can choose k as above, i.e. $[F] \in \mathcal{F}[\![\phi_k]\!]_P$. Hence by Proposition 3 we have $P \models \rho$.
3) Now let ρ contain $\neg \mathit{free}(b)$. Then there is a disjunct $\psi = \bigwedge_{[F] \in \mathrm{fix}(\rho, P)} [F] = c_F \wedge \bigwedge_{[F] \in \mathrm{flex}(\rho, P)} [F] \geq c_F$. We get by the construction of $\theta(\rho, P)$ that $[R] = \left[\prod_{[F] \in \mathit{supp}(M_\omega)} \prod^{c_F} F\right] \in \mathcal{R}[\![\rho]\!]_P$. Like above we can decompose P into R and S, where $S = \prod_{[F] \in \mathit{supp}(M_\omega)} \prod^{M_\omega([F]) - c_F} F$. All fragments of S are members of $\mathrm{flex}(\rho, P)$. Hence for all fragments F of S there is a subformula ϕ of ρ such that $[F] \in \mathcal{F}[\![\phi]\!]_P$. Proposition 3 yields $P \models \rho$.

∎

G. Lemma 6

Let P and P' be processes such that $P \to^* P'$ and φ be a formula of PSTL. Furthermore let ω be an occurrence sequence such that M_ω is the initial marking of $\mathcal{N}[\![P']\!]$. Then $\omega \models \Theta(\varphi, P')$ if and only if $\omega \models \Theta(\varphi, P)$.

Proof: By induction on the structure of PSTL formulae.

Let $\omega \models \Theta(\rho, P')$, i.e. $\omega \models \theta(\rho, P')$. By definition, this is equivalent to $M_\omega \models \theta(\rho, P')$. I.e., there is one disjunct ψ satisfied by M_ω, and ψ was constructed due to some process $[Q] \in \mathcal{R}[\![\rho]\!]_{P'}$. By Lemma 5, $[Q] \in \mathcal{R}[\![\rho]\!]_P$, i.e. ψ is also a disjunct of $\theta(\rho, P)$. Conversely, if $M_\omega \models \psi$ for some disjunct of $\theta(\rho, P)$, all fragments not reachable from P' are not mentioned in $\theta(\rho, P)$ and hence $M_\omega \models \theta(\rho, P')$.

The induction steps for negation and conjunction are immediate by application of the induction hypothesis. In the case of the temporal modality, the application of the hypothesis is more subtle. After expanding the semantics of $\Diamond$, we are left with $\exists y\colon \omega^y \models \Theta(\varphi, P')$. Now, since the transition systems of a process and its structural semantics are isomorphic, the process P'' corresponding to M_{ω^y} is reachable from P', i.e. also $P \to^* P' \to^* P''$. Hence the application of the induction hypothesis is possible, and we get $\omega^y \models \Theta(\varphi, P)$, which is by the definition of the semantics of LTL and of the translation equivalent to $\omega \models \Theta(\Diamond\varphi, P)$. ■