Safe Reasoning with Logic LTS[☆]

Gerald Lüttgen[a,*], Walter Vogler[b]

[a]*Faculty of Information Systems and Applied Computer Sciences, University of Bamberg, 96045 Bamberg, Germany*
[b]*Institute for Computer Science, University of Augsburg, 86135 Augsburg, Germany*

Abstract

Previous work has introduced the setting of Logic Labelled Transition Systems, called Logic LTS or LLTS for short, together with a variant of ready simulation as fully-abstract refinement preorder, which allows one to compose operational specifications using a CSP-style parallel operator and the propositional connectives conjunction and disjunction.

In this article, we show how a temporal logic for specifying safety properties may be embedded into LLTS so that (a) the temporal operators are compositional for ready simulation; (b) ready simulation, when restricted to pairs of processes and formulas, coincides with the logic's satisfaction relation; (c) ready simulation, when restricted to formulas, is entailment.

The utility of this setting as a semantic foundation for mixed operational and temporal-logic specification languages is demonstrated by means of a simple example. We also adopt the concept of may- and must-transitions from modal transition systems for notational convenience, and investigate the relation between modal refinement on modal transition systems and ready simulation on LLTS.

Keywords: labelled transition systems, ready simulation, temporal logic, safety properties, heterogeneous specification, modal refinement

2000 MSC: 68Q55, 68Q60, 68Q85

1. Introduction

Recently, the setting of *Logic Labelled Transition Systems*, also referred to as Logic LTS or LLTS for short, has been introduced [1, 2], which combines operational and logic styles of specification within a unified framework. It includes operational (i.e., process-algebraic) operators [3], such as parallel composition and hiding, and the propositional-logic operators conjunction and disjunction. LLTS extends labelled transition systems by an *inconsistency* predicate on states, where an inconsistent state, or process, denotes empty behaviour that cannot be implemented. Inconsistencies may arise when conjunctively composing processes with different *ready sets*,

[☆]This article is based on an extended abstract that appeared in M. Nielsen, A. Kucera, P. Bro Miltersen, C. Palamidessi, P. Tuma and F. Valencia, eds., 35th Intl. Conf. on *Current Trends in Theory and Practice of Computer Science* (SOFSEM 2009), vol. 5404 of Lecture Notes in Computer Science, pp. 367–387, Springer, 2009.

[*]Corresponding author. Voice: +49 951 863-3850. Fax: +49 951 863-3855.

Email addresses: `gerald.luettgen@swt-bamberg.de` (Gerald Lüttgen), `walter.vogler@informatik.uni-augsburg.de` (Walter Vogler) i.e., initial action sets [1]. The refinement preorder $\sqsubseteq_{\mathrm{RS}}$ adapted for LLTS is a variant of *ready simulation* [4, 5, 6]. It is fully abstract with respect to a reference preorder that relates consistent implementations only to consistent specifications [2], i.e., it is the coarsest compositional preorder with respect to parallel composition and conjunction when taking consistency into account. Most notably, the setting justifies a *simulation-type* preorder when starting from the binary basic observable 'consistency'. The preorder $\sqsubseteq_{\mathrm{RS}}$ is also compositional regarding other operators, namely prefixing, hiding, and external and internal choice, where internal choice coincides with disjunction.

This article extends LLTS by temporal-logic operators, thereby fulfilling our ultimate goal of combining process-algebraic and temporal-logic operators in a uniform compositional refinement setting, in which logical satisfaction and process refinement can be used interchangeably. The temporal logic of interest is a branching-time logic, allowing one to specify the most important class of temporal properties in practice, viz. *safety properties*, over atomic propositions that refer to the enabledness of actions; in particular, we consider the standard temporal operators *always* and *unless* (*weak until*). These operators will be embedded into LLTS such that the logic satisfaction relation $\models$ is *compatible* with $\sqsubseteq_{\mathrm{RS}}$. This means that, firstly, $p \models \phi$ if and only if $p \sqsubseteq_{\mathrm{RS}} \phi$, for any process p and temporal-logic formula ϕ; secondly, ready simulation is *compositional* for the temporal operators. Moreover, when restricted to formulas, $\sqsubseteq_{\mathrm{RS}}$ coincides with *entailment*. The resulting mixed setting satisfies standard process-algebraic, propositional-logic and temporal-logic laws, as one would expect, plus new laws that refer to both logic and temporal operators. For employing our LLTS setting in practice, this article also shows how the idea of may- and must-transitions in Larsen's *modal transition systems* [7] may be adopted to LLTS. This allows one to specify ranges of ready sets compactly, thus achieving representation economy when specifying systems using LLTS; it also permits the formal investigation as to how *modal refinement* [7] on modal transition systems and ready simulation on LLTS relate.

Our LLTS setting is unique in the literature in that it allows one to *freely* mix operational operators, propositional-logic operators and temporal-logic operators, while still permitting *compositional* reasoning, as discussed in the related work section. Our work is strongly inspired by current research into novel notations and methodologies for developing software, where requirements and designs of behaviourally complex systems are regularly specified using a mixture of declarative and operational languages, allowing for the traceable transitioning from software requirements to designs. At the requirements level, popular languages include restricted forms of English or simple spreadsheets (*declarative, also temporal*) and block diagrams or state machines (*operational*). At design level, UML class diagrams combined with the Object Constraint Language [8] (*declarative, partly temporal*) and Statecharts [9] (*operational*) are frequently used [10]. The setting presented in this article serves as the semantic backbone for a related, industry-supported research project ("*Refinement Patterns for Contractual Statecharts*"; EPSRC grant EP/E034853/1) which extends Statecharts with temporal-logic-style contracts and employs ready simulation $\sqsubseteq_{\mathrm{RS}}$ for compositional model checking. Indeed, our main theorem proving the compatibility of $\models$ with $\sqsubseteq_{\mathrm{RS}}$ (Thm. 15) provides a formal basis for compositional verification.

Organisation. The remainder of this article is organised as follows. Sec. 2 revisits the setting of LLTS introduced in [1, 2], including: the notion of inconsistency; parallel, conjunction and disjunction operators on LLTS; ready simulation on LLTS; and full abstraction [2]. Sec. 3 presents our integration of temporal-logic operators in LLTS and proves several key results, foremost compositionality, compatibility, entailment and various laws. The utility of this extension for developing reactive systems is then illustrated in Sec. 4 by means of a small example. Sec. 5

adopts the concept of may- and must-transitions from modal transition systems as a convenient shorthand notation for LLTS, formally relates modal refinement to our ready simulation, and discusses conjunction in modal transition systems in the light of our work. Further related work is discussed in Sec. 6, while Sec. 7 presents our conclusions and directions for future work; in particular, Sec. 7 highlights the challenges of extending our setting so as to be able to also express liveness. Finally, the proofs of some lemmas employed in Secs. 3 and 5 are contained in the appendix, so as not to unnecessarily disrupt the flow of reading.

2. The Setting of Logic LTS

We begin with briefly recalling the setting of LLTS, together with several results and notations that are relevant to this article.

2.1. Inconsistency

LLTS considers *inconsistencies* that may arise under conjunctive composition as first-class observables. A conjunctively composed state between two processes is marked as inconsistent if one offers an action that the other cannot perform, i.e., if the processes have different *ready sets*. Consider the processes p, q and r in Fig. 1(a). Process p and q specify that exactly action a and resp. b is offered initially, i.e., their ready sets are $\{a\}$ and resp. $\{b\}$. Similarly, r specifies that a and b are offered initially and thus has ready set $\{a, b\}$. Hence, $p \wedge q$ and $p \wedge r$ are *inconsistent* (or *false*), and should be tagged as such. Formally, our variant of LTS will be augmented by an *inconsistency predicate* F, so that $p \wedge q \in F$ and $p \wedge r \in F$ in our example. Observe also that, e.g., according to failures semantics [11], p and q (resp. p and r) do not have a common implementation.

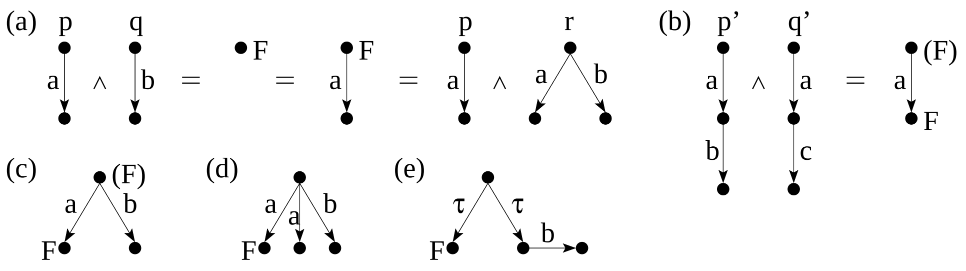

Figure 1: (a)–(b): Conjunctive composition; (c)–(e): Backward propagation.

Most notably, inconsistencies may propagate backwards along transitions. For example, in the conjunction $p' \wedge q'$ shown in Fig. 1(b), both conjuncts require action a to be performed, whence $p' \wedge q'$ should have an a-transition. But this transition leads to an inconsistent state and, in the absence of any alternative a-transition leading to a consistent state, $p' \wedge q'$ must itself be considered inconsistent. In this spirit, inconsistency propagates backwards for the process in Fig. 1(c), whereas it does not for the processes in Figs. 1(d) and 1(e). Note that, in Fig. 1(e), actions τ are used to specify a disjunction between alternatives; hence, our treatment of F corresponds to the law that *false* is the neutral element with respect to disjunction.

2.2. Formal Definitions

Let $\mathcal{A}$ be a non-empty alphabet of visible actions with representatives a and b. With τ being a distinguished, internal action, let $\mathcal{A}_\tau$ denote $\mathcal{A} \cup \{\tau\}$ with representatives α and β. A *labelled transition system*, or LTS, is a triple $\langle P, \longrightarrow, F\rangle$, where P is the set of *processes* (states), $\longrightarrow \subseteq P \times \mathcal{A}_\tau \times P$ is the *transition relation*, and $F \subseteq P$ is the *inconsistency predicate*. We write $p \xrightarrow{\alpha} p'$ instead of $\langle p, \alpha, p'\rangle \in \longrightarrow$ and $p \xrightarrow{\alpha}$ instead of $\exists p'.\, p \xrightarrow{\alpha} p'$, and denote a transition $p \xrightarrow{\alpha} p'$ with $p, p' \notin F$ by $p \xrightarrow{\alpha}_{\mathrm{F}} p'$. Further, we let $\mathcal{I}(p)$ stand for the *ready set* $\{\alpha \in \mathcal{A}_\tau \mid p \xrightarrow{\alpha}\}$ of p. A process p that cannot engage in a τ-transition, i.e., $p \not\xrightarrow{\tau}$, is called *stable*.

We introduce weak transitions by writing (i) $p \overset{\epsilon}{\Longrightarrow} p'$ if $p \xrightarrow{\tau}{}^{*} p'$; and (ii) $p \overset{a}{\Longrightarrow} p'$ if $\exists \overline{p}, \overline{p}'.\, p \overset{\epsilon}{\Longrightarrow} \overline{p} \xrightarrow{a} \overline{p}' \overset{\epsilon}{\Longrightarrow} p'$. If all processes along a computation $p \overset{\epsilon}{\Longrightarrow} p'$ or $p \overset{a}{\Longrightarrow} p'$, including p and p', are consistent, we write $p \overset{\epsilon}{\Longrightarrow}_{\mathrm{F}} p'$ and resp. $p \overset{a}{\Longrightarrow}_{\mathrm{F}} p'$; if in addition p' is stable, we write $p \overset{\epsilon}{\Longrightarrow}\!| \, p'$ and resp. $p \overset{a}{\Longrightarrow}\!| \, p'$. We also introduce a notion to deal with *divergence*, i.e., infinite sequences of τ-transitions, where divergence is viewed as catastrophic if a process cannot stabilise; here, process *p cannot stabilise* if $\nexists p'.\, p \overset{\epsilon}{\Longrightarrow}\!| \, p'$.

Moreover, we require an LTS to satisfy the following *τ-purity* condition: $p \xrightarrow{\tau}$ implies $\nexists a \in \mathcal{A}.\, p \xrightarrow{a}$, for all $p \in P$. Hence, each process represents either an external or internal (disjunctive) choice between its outgoing transitions. This restriction reflects the fact that ready sets can only be observed at stable states, and is justified in [1]. LLTSs must satisfy two further properties, of which the first one formalises our backward propagation of inconsistencies:

Definition 1 (Logic LTS [1]). *An LTS $\langle P, \longrightarrow, F\rangle$ is a* Logic LTS, *or LLTS for short, if*

(LTS1) $p \in F$ *if* $\exists \alpha \in \mathcal{I}(p)\, \forall p' \in P.\ p \xrightarrow{\alpha} p' \implies p' \in F$*;*

(LTS2) *p cannot stabilise* $\implies p \in F$.

2.3. Operators on Logic LTS

LLTSs are equipped with various propositional-logic and process-algebraic operators which were introduced in [1, 2]. The *parallel operator* $\|_A$ on LLTS, for a synchronisation alphabet $A \subseteq \mathcal{A}$, is essentially the one of CSP [11], but it favours τ-transitions over visible transitions so as to preserve τ-purity. Naturally, $p \|_A q$ is inconsistent if p or q is inconsistent. Formally:

Definition 2 (Parallel operator [2]). *The parallel composition of the two LLTSs $\langle P, \longrightarrow_P, F_P\rangle$ and $\langle Q, \longrightarrow_Q, F_Q\rangle$ for the synchronisation alphabet $A \subseteq \mathcal{A}$ is the LLTS $\langle P \|_A Q, \longrightarrow_{P\|_A Q}, F_{P\|_A Q}\rangle$:*

- $P \|_A Q =_{df} \{p \|_A q \mid p \in P,\ q \in Q\}$
- $\longrightarrow_{P\|_A Q}$ *is determined by the following operational rules:*

$$\begin{array}{rcl}
p \xrightarrow{\alpha}_P p',\ \alpha \notin A,\ (\alpha = \tau \text{ or } q \not\xrightarrow{\tau}_Q) & \Longrightarrow & p \|_A q \xrightarrow{\alpha}_{P\|_A Q} p' \|_A q \\
q \xrightarrow{\alpha}_Q q',\ \alpha \notin A,\ (\alpha = \tau \text{ or } p \not\xrightarrow{\tau}_P) & \Longrightarrow & p \|_A q \xrightarrow{\alpha}_{P\|_A Q} p \|_A q' \\
p \xrightarrow{a}_P p',\ q \xrightarrow{a}_Q q',\ a \in A & \Longrightarrow & p \|_A q \xrightarrow{a}_{P\|_A Q} p' \|_A q'
\end{array}$$

- $p \|_A q \in F_{P\|_A Q}$ *if* $p \in F_P$ *or* $q \in F_Q$.

The *conjunction operator* $\wedge$ on LLTS is a synchronous product (or parallel composition) for visible transitions and an asynchronous product for τ-transitions, and also favours τ-transitions. Process $p \wedge q$ is inconsistent if p or q is inconsistent; or if p and q are stable but have different ready sets; or if it becomes inconsistent by backward propagation. Formally:

Definition 3 (Conjunction operator [1]). *The conjunction of the two LLTSs $\langle P, \longrightarrow_P, F_P\rangle$ and $\langle Q, \longrightarrow_Q, F_Q\rangle$ is the LLTS $\langle P \wedge Q, \longrightarrow_{P\wedge Q}, F_{P\wedge Q}\rangle$:*

- *$P \wedge Q =_{df} \{p \wedge q \mid p \in P,\ q \in Q\}$*
- *$\longrightarrow_{P\wedge Q}$ is determined by the following operational rules:*

$$\begin{array}{rcl} p \xrightarrow{\tau}_P p' & \Longrightarrow & p \wedge q \xrightarrow{\tau}_{P\wedge Q} p' \wedge q \\ q \xrightarrow{\tau}_Q q' & \Longrightarrow & p \wedge q \xrightarrow{\tau}_{P\wedge Q} p \wedge q' \\ p \xrightarrow{a}_P p',\ q \xrightarrow{a}_Q q' & \Longrightarrow & p \wedge q \xrightarrow{a}_{P\wedge Q} p' \wedge q' \end{array}$$

- *$F_{P\wedge Q}$ is the least set containing each $p \wedge q$ that satisfies at least one of the following conditions:*

  (C1) *$p \in F_P$ or $q \in F_Q$;*

  (C2) *$p \wedge q \not\xrightarrow{\tau}_{P\wedge Q}$ and $\mathcal{I}(p) \neq \mathcal{I}(q)$;*

  (C3) *$\exists \alpha \in \mathcal{I}(p \wedge q)\, \forall p' \wedge q'.\ p \wedge q \xrightarrow{\alpha}_{P\wedge Q} p' \wedge q' \implies p' \wedge q' \in F_{P\wedge Q}$;*

  (C4) *$p \wedge q$ cannot stabilise.*

Here, a conjunction is inconsistent if a conjunct is inconsistent (cf. Cond. (C1)), and Conds. (C2) and (C3) reflect our intuition of inconsistency and backward propagation. Cond. (C4) is added to ensure (LTS2); note that this condition is not automatically enforced since it is *not* true that $p \wedge q$ can stabilise if both p and q can stabilise.

The *disjunction operator* $\vee$ is an internal choice operator, where $p \vee q$ is inconsistent if both p and q are. Fig. 1(e) depicts a disjunction of an inconsistent process with a consistent process that can engage in action b; hence, the disjunctive process is consistent. Thus, $p \vee q$ essentially is a process with two τ-transitions to p and resp. q; correspondingly, τ is not so much seen as an internal action in our setting but primarily indicates a logical disjunct.

2.4. Refinement on Logic LTS

Our refinement preorder is a variant of ready simulation [4, 5, 6] and thus ensures refinement via successively resolving choices (nondeterminism):

Definition 4 (Ready simulation on LLTS [2]). *Let $\langle P, \longrightarrow_P, F_P\rangle$, $\langle Q, \longrightarrow_Q, F_Q\rangle$ be two LLTSs. Relation $\mathcal{R} \subseteq P \times Q$ is a* stable ready simulation relation*, or stable rs-relation for short, if the following conditions hold, for any $\langle p, q\rangle \in \mathcal{R}$ and $a \in \mathcal{A}$:*

(RS1) *p, q stable*

(RS2) *$p \notin F_P \implies q \notin F_Q$*

(RS3) *$p \overset{a}{\Longrightarrow}\!\!\mid p' \implies \exists q'.\ q \overset{a}{\Longrightarrow}\!\!\mid q'$ and $\langle p', q'\rangle \in \mathcal{R}$*

(RS4) *$p \notin F_P \implies \mathcal{I}(p) = \mathcal{I}(q)$*

We write $p \underset{\sim}{\sqsubset}_{RS} q$ if there exists a stable rs-relation $\mathcal{R}$ such that $\langle p, q\rangle \in \mathcal{R}$. Further, p is ready simulated *by q, in symbols $p \sqsubseteq_{RS} q$, if $\forall p'.\ p \overset{\epsilon}{\Longrightarrow}\!\!\mid p' \implies \exists q'.\ q \overset{\epsilon}{\Longrightarrow}\!\!\mid q'$ and $p' \underset{\sim}{\sqsubset}_{RS} q'$. Finally, we let $=_{RS}$ stand for the kernel of $\sqsubseteq_{RS}$.*

While we allow transitions leaving inconsistent states, they are ignored in the above definition. Thus, one may remove such transitions without changing the relevant behaviour of processes; for technical convenience, we do not include this additional normalisation when defining our operators. The above operators satisfy the following properties with respect to $\sqsubseteq_{\mathrm{RS}}$:

Proposition 5 ([2]). *Let p, q, r be processes, $p' \sqsubseteq_{RS} q'$ and $A \subseteq \mathcal{A}$.*
(1) Compositionality: $p' \wedge r \sqsubseteq_{RS} q' \wedge r,\ p' \vee r \sqsubseteq_{RS} q' \vee r,\ p' \|_A r \sqsubseteq_{RS} q' \|_A r;$
(2) $\wedge$ is conjunction: $r \sqsubseteq_{RS} p \wedge q \iff r \sqsubseteq_{RS} p$ *and* $r \sqsubseteq_{RS} q;$
(3) $\vee$ is disjunction: $p \vee q \sqsubseteq_{RS} r \iff p \sqsubseteq_{RS} r$ *and* $q \sqsubseteq_{RS} r.$

The second item above demonstrates that $\wedge$ is indeed conjunction: clearly, a process should implement a conjunction if and only if it implements both conjuncts.

In addition, we have shown in [2] that relation $\sqsubseteq_{\mathrm{RS}}$ is fully abstract for the preorder $\sqsubseteq_F$, which is defined by $p \sqsubseteq_F q$ if and only if $q \in F_Q \implies p \in F_P$ (i.e., an inconsistent specification q cannot have a consistent implementation p as refinement). Formally:

Theorem 6 (Full abstraction [2]). *The largest precongruence within $\sqsubseteq_F$, with respect to parallel composition and conjunction, equals $\sqsubseteq_{RS}$.*

This means that our simulation-type preorder is justified simply by starting from a binary basic observable, namely consistency; moreover, the preorder is compositional for parallel composition and conjunction, which is also true for other operators, e.g., disjunction, prefixing, external choice and hiding [2].

3. Temporal Logic & Logic LTS

The temporal properties we embed in LLTS are essentially the safety properties of the universal fragment of the temporal logic *action-based CTL* [12], adapted to our setting. This is the largest fragment we can hope for since, firstly, LLTS is based on standard LTS, without Büchi annotations or similar acceptance conditions; hence, finite-state LLTS is not expressive enough for encoding liveness (or fairness) properties. Secondly, we wish for the logic satisfaction relation $\models$ to be compatible with $\sqsubseteq_{\mathrm{RS}}$, i.e., $p \models \phi \iff p \sqsubseteq_{\mathrm{RS}} \phi$, for any process p and formula ϕ (cf. Thm. 15 below). Hence, by transitivity of $\sqsubseteq_{\mathrm{RS}}$, we have that $p \sqsubseteq_{\mathrm{RS}} q$ and $q \models \phi$ implies $p \models \phi$, i.e., the implementation p with the 'smaller' behaviour has to satisfy more formulas than the specification q. This justifies our focus on the *universal* fragment.

3.1. Syntax, Satisfaction & Characterisation

We consider the following set $\mathcal{F}$ of temporal-logic formulas ϕ:

$$\phi \quad ::= \quad tt \mid f\!f \mid en(a) \mid dis(a) \mid \phi \vee \phi \mid \phi \wedge \phi \mid [a]\phi \mid \Box\phi \mid \phi \,\mathsf{W}\, \phi$$

Here, the atomic propositions $en(a)$ and $dis(a)$ denote the enabledness and resp. disabledness of action a, and $[a]$, $\Box$ and W are the usual *next*, *always* (*generally*) and *unless* (*weak until*) operators. The latter can be seen as a weak version of the until in [12]. In addition, formula tt (resp. $f\!f$) may be derived as $en(a) \vee dis(a)$ (resp. $en(a) \wedge dis(a)$); moreover, $\Box\phi$ is equivalent to $\phi \,\mathsf{W}\, f\!f$ (cf. Sec. 3.3). Note that having $en(a)$ and $dis(a)$ in the logic is similar to positive normal forms in state-based logics.

The meaning of formulas is defined via a satisfaction relation $\models$. Recall that, in our setting, action τ is not so much seen as an internal action, but an instable process p is a ‘disjunction’; hence, $p \models \phi$ should mean that $p_0 \models \phi$ for all ‘disjuncts’ p_0 of p, i.e., for each p_0 with $p \overset{\epsilon}{\Longrightarrow|} p_0$. Thus, we define $\models$ as follows, where $\overset{\mathcal{A}}{\Longrightarrow|}$ stands for $\bigcup_{a \in \mathcal{A}} \overset{a}{\Longrightarrow|}$:

Definition 7 (Satisfaction relation). *Given an LLTS with state set P, the satisfaction relation $\models \subseteq P \times \mathcal{F}$ is defined by the following rules:*

$$\begin{array}{lll}
p \models tt & & always \\
p \models f\!f & if & p \in F \\
p \models en(a) & if & \forall p_0.\, p \overset{\epsilon}{\Longrightarrow|} p_0 \implies p_0 \overset{a}{\longrightarrow} \\
p \models dis(a) & if & \forall p_0.\, p \overset{\epsilon}{\Longrightarrow|} p_0 \implies p_0 \overset{a}{\not\longrightarrow} \\
 & & \\
p \models \phi \vee \psi & if & \forall p_0.\, p \overset{\epsilon}{\Longrightarrow|} p_0 \implies (p_0 \models \phi \ or \ p_0 \models \psi) \\
p \models \phi \wedge \psi & if & \forall p_0.\, p \overset{\epsilon}{\Longrightarrow|} p_0 \implies (p_0 \models \phi \ and \ p_0 \models \psi) \\
 & & \\
p \models [a]\phi & if & \forall p_0, p_1.\ p \overset{\epsilon}{\Longrightarrow|} p_0 \overset{a}{\Longrightarrow|} p_1 \implies p_1 \models \phi \\
p \models \Box\phi & if & \forall p_0, p_1, \ldots, p_n.\, (p \overset{\epsilon}{\Longrightarrow|} p_0 \overset{\mathcal{A}}{\Longrightarrow|} p_1 \ldots \overset{\mathcal{A}}{\Longrightarrow|} p_n \implies p_n \models \phi) \\
p \models \phi \mathsf{W} \psi & if & \forall p_0, p_1, \ldots, p_n.\, (p \overset{\epsilon}{\Longrightarrow|} p_0 \overset{\mathcal{A}}{\Longrightarrow|} p_1 \ldots \overset{\mathcal{A}}{\Longrightarrow|} p_n \implies (p_n \models \phi \ or \ \exists i \leq n.\, p_i \models \psi))
\end{array}$$

This definition coincides for τ-less p with the standard one but, in contrast to processes within LTS, *ff* is satisfiable, namely by inconsistent processes.

To motivate the quantification “$\forall p_0.\, p \overset{\epsilon}{\Longrightarrow|} p_0$” for the $\vee$-case further, consider that $\models$ must be defined such that the process p that has one initial a-transition followed by a b-transition, satisfies formula $[a]en(b)$. Similarly, the process q that has one initial a-transition followed by a c-transition, should satisfy $[a]en(c)$. Since we aim for a setting in which $\models$ may be freely replaced by $\sqsubseteq_{\mathrm{RS}}$ and since $\sqsubseteq_{\mathrm{RS}}$ is a precongruence, we must have $p \vee q \models [a]en(b) \vee [a]en(c)$. In a classic definition of satisfiability, this would mean $p \vee q \models [a]en(b)$ or $p \vee q \models [a]en(c)$, which are both clearly false. In addition and as claimed above, each process p indeed satisfies $en(a) \vee dis(a)$ since each ‘disjunct’ p_0 of p is stable and hence either can engage in a (i.e., satisfies $en(a)$) or cannot (i.e., satisfies $dis(a)$).

As an aside and provided that p and q belong to LLTSs that are *finitely branching*, we get a Hennessy-Milner-style characterisation of $\sqsubseteq_{\mathrm{RS}}$; here, $\mathcal{F}_{\mathrm{RS}}$ are the *essential formulas*, namely the formulas in $\mathcal{F}$ that do neither contain operators $\wedge$, $\Box$ and W, nor sub-formulas *tt* and $dis(a)$, i.e.,

$$\phi_{\mathrm{RS}} \quad ::= \quad f\!f \mid en(a) \mid \phi_{\mathrm{RS}} \vee \phi_{\mathrm{RS}} \mid [a]\phi_{\mathrm{RS}}\,.$$

Theorem 8 (Characterisation). $p \sqsubseteq_{RS} q \iff \forall \phi \in \mathcal{F}_{RS}.\, q \models \phi \implies p \models \phi.$

This characterisation is pretty much a corollary to an analogous result of Bloom [4], and is thus not proved here. It should be noted that, in his thesis, Bloom considered a characterisation based on the opposite implication than the one we require. Correspondingly, he used the dual fragment of formulas, employing $\langle a \rangle$-modalities instead of $[a]$-modalities.

3.2. Embedding in Logic LTS

We embed our temporal formulas into LLTS and present the desired compatibility result between $\models$ and $\sqsubseteq_{\mathrm{RS}}$. The embedding is conducted along the structure of formulas. Formula *tt*

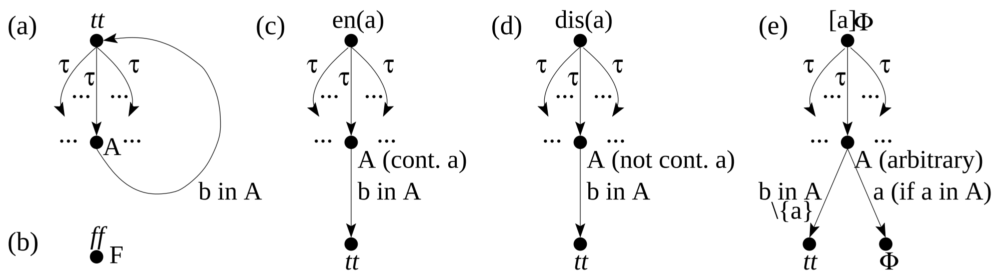

Figure 2: Embedding of temporal-logic formulas into LLTS.

corresponds to the initial state of the LLTS sketched in Fig. 2(a), which can nondeterministically select an arbitrary ready set $A \subseteq \mathcal{A}$ via a τ-transition to process A. From there, it can engage in any transition labelled with an action $b \in A$ and return to *tt*. Hence, *tt* is a process that can simulate any other process, and is thus indeed the desired 'universal' process. Formula *ff* is trivially mapped to the inconsistent process depicted in Fig. 2(b), which can only ready simulate an inconsistent process. Formula $en(a)$ corresponds to the initial state of the LLTS in Fig. 2(c). This can select any ready set A containing a by silently moving to process A, from where it can engage in a b-transition, for any $b \in A$, to *tt*. We embed formula $dis(a)$ analogously, where we require $a \notin A$ instead of $a \in A$; see Fig. 2(d).

Formula $\phi \wedge \psi$ (resp. $\phi \vee \psi$) is embedded by conjunctively (resp. disjunctively) composing the LLTSs of the embeddings of ϕ and ψ, using operator $\wedge$ (resp. $\vee$) on LLTS. The embedding of a formula $[a]\phi$ is sketched in Fig. 2(e). Again, the initial process may choose an arbitrary ready set A. The corresponding process A can engage in a b-step, for any $b \in A \setminus \{a\}$, to *tt*. In addition, if $a \in A$, there is an a-step to the initial state of ϕ's embedding. Hence, any a-derivative of $[a]\phi$ behaves as ϕ, whereas arbitrary behaviour is permitted for differently labelled derivatives.

We now define $\Box$- and W-operators on LLTS, which facilitate the straightforward embedding of formulas $\Box\phi$ and $\phi \,\mathsf{W}\, \psi$:

Definition 9 ($\Box$-operator, "always"). *Let $\langle P, \longrightarrow_P, F_P\rangle$ be an LLTS. Then, $\Box p$, for $p \in P$, is process (p) in LLTS $\langle \Box P, \longrightarrow_{\Box P}, F_{\Box P}\rangle$, where:*

- $\Box P =_{df} \{\vec{p} = (p_1, p_2, \dots, p_n) \,|\, n{\geq}1,\ \forall 1{\leq}i{\leq}n.\ p_i{\in}P\}$ *is the set of finite vectors over P.*
- $\longrightarrow_{\Box P}$ *is defined by the following operational rules:*

$$\begin{array}{lcl} p_i \xrightarrow{\tau}_P p_i' & \Longrightarrow & (p_1, \dots, p_i, \dots, p_n) \xrightarrow{\tau}_{\Box P} (p_1, \dots, p_i', \dots, p_n) \\ \forall i.\ p_i \xrightarrow{a}_P p_i' & \Longrightarrow & (p_1, \dots, p_n) \xrightarrow{a}_{\Box P} (p_1', \dots, p_n', p)\,. \end{array}$$

- *$F_{\Box P}$ is the least set of finite vectors such that $\vec{p} = (p_1, \dots, p_n) \in F_{\Box P}$ if any one of the following conditions holds:*

  (BF1) $\exists i.\ p_i \in F_P$;

  (BF2) *$\vec{p}$ stable but $\exists i, j.\ \mathcal{I}(p_i) \neq \mathcal{I}(p_j)$;*

  (BF3) $\exists \alpha \in \mathcal{I}(\vec{p})\, \forall \vec{p}\,'.\ \vec{p} \xrightarrow{\alpha}_{\Box P} \vec{p}\,' \implies \vec{p}\,' \in F_{\Box P}$;

  (BF4) *$\vec{p}$ cannot stabilise outside $F_{\Box P}$, i.e., via a sequence of transitions over states that are* not *in $F_{\Box P}$.*

In the sequel, we use the convention that $\vec{p} \in \Box P$ has components $p_1, p_2, \ldots, p_n$. Observe that $\langle \Box P, \longrightarrow_{\Box P}, F_{\Box P}\rangle$ is indeed an LLTS and that $\vec{p}$ behaves as the conjunction $\bigwedge_i p_i$. Intuitively, the above construction adds p to the process vector after every visible step. To illustrate the above construction, we sketch part of the LLTS of $\Box en(a)$ in Fig. 3.

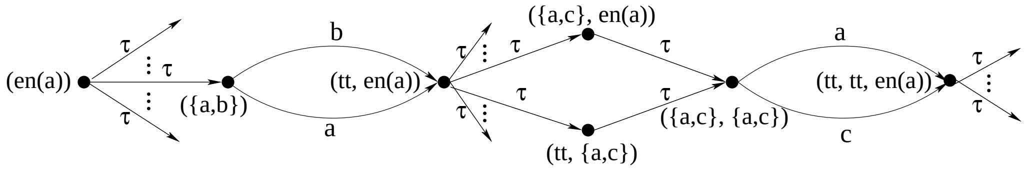

Figure 3: Sketch of the LLTS of $\Box en(a)$.

Although the employed vector notation is convenient for proving compositionality, its use immediately leads to an infinite state space. However, we could have used process *sets* instead of process vectors, which would result in an $=_{\mathrm{RS}}$-equivalent definition. This would make the process sets of $\Box P$ finite if P is finite, and permit an implementation of the $\Box$-operator.

Definition 10 (W -operator, "unless"). *Let $\langle P, \longrightarrow_P, F_P\rangle$ and $\langle Q, \longrightarrow_Q, F_Q\rangle$ be LLTSs. Then, $p\,W q$, for $p \in P$ and $q \in Q$, is a process within the LLTS $\langle P\,WQ, \longrightarrow_{P\,WQ}, F_{P\,WQ}\rangle$, where:*

- *$P\,WQ =_{df} \{p\,Wq\} \cup \Box P \cup (\Box P \times Q)$ with $\Box P = \{\vec{p} \,|\, n{\geq}1,\ \forall 1{\leq}i{\leq}n.\ p_i \in P\}$.*
- *$\longrightarrow_{P\,WQ}$ is defined by the following operational rules:*

$$\begin{array}{rcl}
\textit{always} & & p\,Wq \xrightarrow{\tau}_{P\,WQ} \langle (), q\rangle \\
\textit{always} & & p\,Wq \xrightarrow{\tau}_{P\,WQ} (p) \\
p_i \xrightarrow{\tau}_P p_i' & \Longrightarrow & (p_1, \ldots, p_i, \ldots, p_n) \xrightarrow{\tau}_{P\,WQ} (p_1, \ldots, p_i', \ldots, p_n) \\
\forall i.\ p_i \xrightarrow{a}_P p_i' & \Longrightarrow & (p_1, \ldots, p_n) \xrightarrow{a}_{P\,WQ} \langle (p_1', \ldots, p_n'), q\rangle \\
\forall i.\ p_i \xrightarrow{a}_P p_i' & \Longrightarrow & (p_1, \ldots, p_n) \xrightarrow{a}_{P\,WQ} (p_1', \ldots, p_n', p) \\
q' \xrightarrow{\tau}_Q q'' & \Longrightarrow & \langle (p_1, \ldots, p_n), q'\rangle \xrightarrow{\tau}_{P\,WQ} \langle (p_1, \ldots, p_n), q''\rangle \\
p_i \xrightarrow{\tau}_P p_i' & \Longrightarrow & \langle (p_1, \ldots, p_n), q'\rangle \xrightarrow{\tau}_{P\,WQ} \langle (p_1, \ldots, p_i', \ldots, p_n), q'\rangle \\
q' \xrightarrow{a}_Q q'' \text{ and } \forall i.\ p_i \xrightarrow{a}_P p_i' & \Longrightarrow & \langle (p_1, \ldots, p_n), q'\rangle \xrightarrow{a}_{P\,WQ} \langle (p_1', \ldots, p_n'), q''\rangle\,.
\end{array}$$

- *$F_{P\,WQ}$ is the least set such that $r \in F_{P\,WQ}$ if any one of these conditions holds:*

  (RF1) *$r = \vec{p}$ or $r = \langle \vec{p}, q'\rangle$ so that $\exists i.\ p_i \in F_P$, or $r = \langle \vec{p}, q'\rangle$ and $q' \in F_Q$;*
  (RF2) *r is stable, equals $\vec{p}$ or $\langle \vec{p}, q'\rangle$ and $\exists i, j.\ \mathcal{I}(p_i) \neq \mathcal{I}(p_j)$,*
  or $r = \langle \vec{p}, q'\rangle$ stable and $\exists i.\ \mathcal{I}(p_i) \neq \mathcal{I}(q')$;
  (RF3) *$\exists \alpha \in \mathcal{I}(r)\, \forall r'.\ r \xrightarrow{\alpha}_{P\,WQ} r' \implies r' \in F_{P\,WQ}$;*
  (RF4) *r cannot stabilise outside $F_{P\,WQ}$.*

This LLTS is well-defined. Processes $\langle \vec{p}, q\rangle$ should be thought of as $\bigwedge_i p_i \wedge q$. Intuitively, $p\,\mathrm{W}\,q$ behaves similarly to $\Box p$; however, initially and at any stable state along a computation, it may decide to withdraw from conjoining p in favour of a one-off conjunction with q.

Theorem 11 (Compositionality). *Let $p \sqsubseteq_{RS} q$, $r \sqsubseteq_{RS} s$ and $a \in \mathcal{A}$. Then, $[a]p \sqsubseteq_{RS} [a]q$, $\Box p \sqsubseteq_{RS} \Box q$ and $p \,\mathsf{W}\, r \sqsubseteq_{RS} q \,\mathsf{W}\, s$.*

An essential point when proving this theorem is the reasoning about inconsistencies; e.g., for a $\Box P$ LLTS, we adapt the concept of witness of [1]:

Definition 12 ($\Box$-witness). *A* $\Box$-witness *for $\Box P$ is a set $W \subseteq \Box P$ such that, for all $\vec{p} \in W$, the following conditions hold:*

- ***(W1)*** $\forall i.\, p_i \notin F_P$;
- ***(W2)*** $\vec{p}$ *stable* $\implies \forall i, j.\, \mathcal{I}(p_i) = \mathcal{I}(p_j)$;
- ***(W3)*** $\forall \alpha \in \mathcal{I}(\vec{p}) \exists \vec{p}\,'.\ \vec{p} \xrightarrow{\alpha}_{\Box P} \vec{p}\,'$ *and* $\vec{p}\,' \in W$;
- ***(W4)*** $\vec{p}$ *can stabilise in W, i.e.,*
  $\exists \vec{p}_1, \dots \vec{p}_m.\ \vec{p} \xrightarrow{\tau}_{\Box P} \vec{p}_1 \xrightarrow{\tau}_{\Box P} \dots \xrightarrow{\tau}_{\Box P} \vec{p}_m \not\xrightarrow{\tau}_{\Box P}$ *and* $\forall i.\, \vec{p}_i \in W$.

The following straightforward property of $\Box$-witnesses gives us a useful tool for proving that *always* processes are consistent:

Proposition 13. *$\vec{p} \notin F_{\Box P}$ if and only if $\exists\, \Box$-witness $W.\ \vec{p} \in W$.*

Proof. Direction "$\Longrightarrow$" follows from the fact that $\overline{F_{\Box P}}$, the complement of $F_{\Box P}$, is an $\Box$-witness. For direction "$\Longleftarrow$" we note that $\overline{W}$ satisfies the conditions of $F_{\Box P}$, whence $F_{\Box P} \subseteq \overline{W}$. $\Box$

The concrete witness needed in the $\Box$-compositionality proof is the following:

Lemma 14 (Concrete witness). *Given stable $p \notin F_P$ and $q \in Q$ with $p \underset{\sim}{\sqsubset}_{RS} q$, the set $W =_{df} W_1 \cup W_2 \subseteq \Box Q$ is a $\Box$-witness, where*

$$W_1 \;=_{df}\; \{\vec{q} = (q_1, \dots, q_n) \,|\, \exists \vec{p} = (p_1, \dots, p_n).\ \vec{p} \notin F_{\Box P} \text{ and } \forall i.\, p_i \underset{\sim}{\sqsubset}_{RS} q_i\}\,;$$
$$W_2 \;=_{df}\; \{\vec{q} = (q_1, \dots, q_n) \,|\, \exists \vec{q}\,' = (q_1', \dots, q_n').\ \vec{q}\,' \in W_1 \text{ and } \forall i.\, q_i \overset{\epsilon}{\Longrightarrow}\!\!| \, q_i'\}\,.$$

The proof of this lemma is straightforward and contained in the appendix. A similar witness concept and construction is needed for proving the W-operator compositional. We are now in a position to prove Thm. 11:

Proof. *[of Thm. 11]* Note that the compositionality results for parallel composition, conjunction and disjunction were stated and proved in [2].

We start off with sketching the compositionality proof for $[a]$. Firstly, stable process A in the encoding $[a]P$ of $[a]p$ is matched by stable process A in the encoding $[a]Q$ of $[a]q$, showing Conds. (RS1) and (RS4). For Cond. (RS2), we observe: if $A \in F_{[a]Q}$, then we must have $a \in A$ and $q \in F_Q$, thus $p \in F_P$ and $A \in F_{[a]P}$. Now, we assume $A \notin F_{[a]P}$; if $A \xrightarrow{a}_{\mathrm{F}} p \overset{\epsilon}{\Longrightarrow}\!\!|\, p_0$ then, since $p \sqsubseteq_{\mathrm{RS}} q$ by assumption, there is some q_0 with $q \overset{\epsilon}{\Longrightarrow}\!\!|\, q_0$ and $p_0 \underset{\sim}{\sqsubset}_{\mathrm{RS}} q_0$; furthermore, $A \xrightarrow{a}_{\mathrm{F}} q \overset{\epsilon}{\Longrightarrow}\!\!|\, q_0$ in $[a]Q$. For $b \in \mathcal{A} \setminus \{a\}$, we have $A \xrightarrow{b}_{\mathrm{F}} tt$ in both $[a]P$ and $[a]Q$. Thus, Cond. (RS3) holds, too.

We now turn to proving compositionality regarding operator $\Box$. If $p \in F_P$, then $\Box p \sqsubseteq_{\mathrm{RS}} \Box q$ is trivial. Now consider $p \notin F_P$ (and hence $q \notin F_Q$). Since the processes on which $\Box p$ can stabilise are exactly those $(\hat{p})$ with $p \overset{\epsilon}{\Longrightarrow}\!\!|\, \hat{p}$ (and similarly for q), we only have to establish the following statement:

Let $p \sqsubseteq_{\mathrm{RS}} q$ be given, i.e., for all $\hat{p}$ with $p \overset{\epsilon}{\Longrightarrow\!|} \hat{p}$, there exists some $\hat{q}$ such that $q \overset{\epsilon}{\Longrightarrow\!|} \hat{q}$ and $\hat{p} \sqsubset\!\!\!\sim_{\mathrm{RS}} \hat{q}$. We show that $(\hat{p}) \sqsubset\!\!\!\sim_{\mathrm{RS}} (\hat{q})$ in $\Box P$ and resp. $\Box Q$. To do so, it is sufficient to prove that

$$\mathcal{R} =_{\mathrm{df}} \{\langle \vec{p}, \vec{q} \rangle \,|\, \vec{p} = (p_1, \ldots, p_n),\ \vec{q} = (q_1, \ldots, q_n),\ \forall 1{\le}i{\le}n.\ p_i \sqsubset\!\!\!\sim_{\mathrm{RS}} q_i\}$$

is a stable rs-relation. Obviously, $\langle (\hat{p}), (\hat{q}) \rangle \in \mathcal{R}$. We verify Conds. (RS1)–(RS4) of Def. 4, using the $\Box$-witness $W_1 \cup W_2$ of Lemma 14:

(RS1) Here, $\vec{p}$ and $\vec{q}$ are stable since all p_i and q_i are stable due to $p_i \sqsubset\!\!\!\sim_{\mathrm{RS}} q_i$.

(RS2) If $\vec{p} \notin F_{\Box P}$, then $\vec{q} \in W_1$ since $p_i \sqsubset\!\!\!\sim_{\mathrm{RS}} q_i$ for all i. Hence, $\vec{q} \notin F_{\Box Q}$ by Prop. 13.

(RS3) Let $\vec{p} \overset{a}{\Longrightarrow\!|} \vec{p}'$, i.e., $(p_1, \ldots, p_n) \xrightarrow{a}_{\mathrm{F}} (\overline{p}_1, \ldots, \overline{p}_n, p) \overset{\epsilon}{\Longrightarrow\!|} (p_1', \ldots, p_n', \hat{p}) = \vec{p}'$ for some suitably chosen $\overline{p}_i$. Hence, $\overline{p}_i \overset{\epsilon}{\Longrightarrow\!|} p_i'$ and resp. $p \overset{\epsilon}{\Longrightarrow\!|} \hat{p}$, as well as $p_i \xrightarrow{a}_{\mathrm{F}} \overline{p}_i$, for all $1 \le i \le n$. Therefore, by $p_i \sqsubset\!\!\!\sim_{\mathrm{RS}} q_i$ and Cond. (RS3), there exist $\overline{q}_i$ and q_i' such that $q_i \xrightarrow{a}_{\mathrm{F}} \overline{q}_i \overset{\epsilon}{\Longrightarrow\!|} q_i'$ and $p_i' \sqsubset\!\!\!\sim_{\mathrm{RS}} q_i'$, and also $\hat{p} \sqsubset\!\!\!\sim_{\mathrm{RS}} \hat{q}$ by assumption. Thus, $\vec{q} \xrightarrow{a} (\overline{q}_1, \ldots, \overline{q}_n, q) \overset{\epsilon}{\Longrightarrow} \vec{q}' =_{\mathrm{df}} (q_1', \ldots, q_n', \hat{q}) \overset{\mathcal{I}}{\not\longrightarrow}$. Since $\vec{p}' \notin F_{\Box P}$, we have $\vec{q}' \in W_1$, whence all processes along the computation $(\overline{q}_1, \ldots, \overline{q}_n, q) \overset{\epsilon}{\Longrightarrow} \vec{q}'$ are in W_2. Finally, $\vec{q} \notin F_{\Box Q}$ by Cond. (RS2) above. Summarising and referring to Prop. 13, we have $\vec{q} \overset{a}{\Longrightarrow\!|} \vec{q}'$ and, obviously, $\langle \vec{p}', \vec{q}' \rangle \in \mathcal{R}$.

(RS4) The premise $\vec{p} \notin F_{\Box P}$ and the stability of $\vec{p}$ by Cond. (RS1) imply $\mathcal{I}(\vec{p}) = \mathcal{I}(p_1) = \ldots = \mathcal{I}(p_n)$. Thus, by $p_i \sqsubset\!\!\!\sim_{\mathrm{RS}} q_i$ according to the definition of $\mathcal{R}$, we have $\mathcal{I}(p_i) = \mathcal{I}(q_i)$ for all i. Therefore, $\mathcal{I}(\vec{p}) = \mathcal{I}(q_1) = \ldots = \mathcal{I}(q_n) = \mathcal{I}(\vec{q})$ by our operational rules.

This completes the compositionality proof with respect to the $\Box$-operator. The proof for the W-operator follows along similar lines; it is omitted here since it does not require any new concept but only additional notation and case distinctions. $\Box$

We now turn to stating and proving the most important result of this article:

Theorem 15 (Compatibility). *Let p be a process and ϕ a temporal-logic formula in $\mathcal{F}$. Then, $p \models \phi \iff p \sqsubseteq_{RS} \phi$.*

The proof of this theorem uses the following lemma for dealing with process vectors in the case that $\phi = \Box\psi$; its proof can be found in the appendix.

Lemma 16. *Let $\vec{q} = (q_1, \ldots, q_n) \in \Box\Psi$ and $p \sqsubseteq_{RS} \vec{q}$. Then, $p \sqsubseteq_{RS} q_i$ for all $1 \le i \le n$.*

Proof. *[of Thm. 15]* (*Hint:* This proof is reused in connection with Table 2 below; the reader might want to postpone reading this proof until the latter parts of Sec. 3.3.)

The proof is by induction on the structure of ϕ. Note that the cases $\phi = tt$ and $\phi = ff$ are trivial, the case $\phi = dis(a)$ is analogous to the one for $\phi = en(a)$, and the case for $\phi = \psi_1 \mathsf{W} \psi_2$ follows along similar lines to the one for $\phi = \Box\psi$. Therefore, we focus on the remaining cases:

- $\underline{\phi = en(a):}$ ("$\Longrightarrow$") Let $p \models en(a)$, i.e., $p \overset{\epsilon}{\Longrightarrow\!|} p_0$ implies $p_0 \xrightarrow{a}$, for any p_0. Then, $p \sqsubseteq_{\mathrm{RS}} en(a)$ since $p_0 \sqsubset\!\!\!\sim_{\mathrm{RS}} \mathcal{I}(p_0)$.

  ("$\Longleftarrow$") For all p_0 such that $p \overset{\epsilon}{\Longrightarrow\!|} p_0$ we must have some action set A containing a with $p_0 \sqsubset\!\!\!\sim_{\mathrm{RS}} A$. Since $p_0 \notin F$, this means by Cond. (RS4) that $a \in \mathcal{I}(p_0)$, and by (LTS1) that $p_0 \xrightarrow{a}_{\mathrm{F}}$. Hence, $p \models en(a)$.

- $\underline{\phi = [a]\psi:}$ ("$\Longrightarrow$") Let $p \models [a]\psi$ and consider some process p_0 with $p \overset{\epsilon}{\Longrightarrow}\!\!| \, p_0$. By the definition of $\models$ we know that $p_1 \models \psi$ for all p_1 such that $p_0 \overset{a}{\Longrightarrow}\!\!| \, p_1$. Hence, $p_1 \sqsubseteq_{\mathrm{RS}} \psi$ by induction hypothesis, which implies $\psi \overset{\epsilon}{\Longrightarrow}\!\!| \, q_1$ for some q_1 with $p_1 \underset{\sim}{\sqsubset}_{\mathrm{RS}} q_1$. We argue $p_0 \underset{\sim}{\sqsubset}_{\mathrm{RS}} \mathcal{I}(p_0)$ by showing that $\{\langle p_0, \mathcal{I}(p_0)\rangle\} \cup \underset{\sim}{\sqsubset}_{\mathrm{RS}}$ is a stable rs-relation. Obviously, the pair $\langle p_0, \mathcal{I}(p_0)\rangle$ satisfies Conds. (RS1), (RS2) and (RS4) of Def. 4. Regarding Cond. (RS3), we have for all $p_0 \overset{b}{\Longrightarrow}\!\!| \, p_1$ with $b \neq a$ (and $b \in \mathcal{I}(p_0)$) that $\mathcal{I}(p_0) \overset{b}{\Longrightarrow}\!\!| \, tt$ and $p_1 \underset{\sim}{\sqsubset}_{\mathrm{RS}} tt$. Furthermore, for all $p_0 \overset{a}{\Longrightarrow}\!\!| \, p_1$, we have $\mathcal{I}(p_0) \overset{a}{\longrightarrow}_{\mathrm{F}} \psi \overset{\epsilon}{\Longrightarrow}\!\!| \, q_1$ with $p_1 \underset{\sim}{\sqsubset}_{\mathrm{RS}} q_1$, as noted above. Altogether, we thus obtain $p \sqsubseteq_{\mathrm{RS}} [a]\psi$.

  ("$\Longleftarrow$") Let $p \sqsubseteq_{\mathrm{RS}} [a]\psi$. Therefore, whenever $p \overset{\epsilon}{\Longrightarrow}\!\!| \, p_0$, we have $[a]\psi \overset{\epsilon}{\Longrightarrow}\!\!| \, A$ for some A with $p_0 \underset{\sim}{\sqsubset}_{\mathrm{RS}} A$. Obviously, $A = \mathcal{I}(p_0)$. By our LLTS encoding of $[a]\psi$ and Cond. (RS3), $p_0 \overset{a}{\Longrightarrow}\!\!| \, p_1$ for any such p_1 implies $\psi \overset{\epsilon}{\Longrightarrow}\!\!| \, q_1$ for some q_1 with $\mathcal{I}(p_0) \overset{a}{\Longrightarrow}\!\!| \, q_1$ and $p_1 \underset{\sim}{\sqsubset}_{\mathrm{RS}} q_1$. Hence, $p_1 \sqsubseteq_{\mathrm{RS}} \psi$ and, by induction hypothesis, $p_1 \models \psi$. Therefore, $p \models [a]\psi$.

- $\underline{\phi = \psi_1 \vee \psi_2:}$ ("$\Longrightarrow$") Let $p \models \psi_1 \vee \psi_2$. Whenever $p \overset{\epsilon}{\Longrightarrow}\!\!| \, p_0$, then $p_0 \models \psi_1$ or $p_0 \models \psi_2$, i.e., $p_0 \sqsubseteq_{\mathrm{RS}} \psi_1$ or $p_0 \sqsubseteq_{\mathrm{RS}} \psi_2$ by induction hypothesis. Assume w.l.o.g. that $p_0 \sqsubseteq_{\mathrm{RS}} \psi_1$, whence $\psi_1 \overset{\epsilon}{\Longrightarrow}\!\!| \, q_0$ for some q_0 with $p_0 \underset{\sim}{\sqsubset}_{\mathrm{RS}} q_0$. By $\psi_1 \vee \psi_2 \overset{\epsilon}{\Longrightarrow}\!\!| \, q_0$, we conclude $p \sqsubseteq_{\mathrm{RS}} \psi_1 \vee \psi_2$.

  ("$\Longleftarrow$") Let $p \sqsubseteq_{\mathrm{RS}} \psi_1 \vee \psi_2$ and $p \overset{\epsilon}{\Longrightarrow}\!\!| \, p_0$. Therefore, w.l.o.g., $\psi_1 \vee \psi_2 \overset{\epsilon}{\Longrightarrow}\!\!| \, q_0$ due to $\psi_1 \overset{\epsilon}{\Longrightarrow}\!\!| \, q_0$ with $p_0 \underset{\sim}{\sqsubset}_{\mathrm{RS}} q_0$. Hence, $p_0 \sqsubseteq_{\mathrm{RS}} \psi_1$ and, by induction hypothesis, $p_0 \models \psi_1$. This implies $p \models \psi_1 \vee \psi_2$.

- $\underline{\phi = \psi_1 \wedge \psi_2:}$ ("$\Longrightarrow$") Let $p \models \psi_1 \wedge \psi_2$. Whenever $p \overset{\epsilon}{\Longrightarrow}\!\!| \, p_0$, then $p_0 \models \psi_1$ and $p_0 \models \psi_2$, i.e., $p_0 \sqsubseteq_{\mathrm{RS}} \psi_1$ and $p_0 \sqsubseteq_{\mathrm{RS}} \psi_2$ by induction hypothesis. By Prop. 5(2), we get $p_0 \sqsubseteq_{\mathrm{RS}} \psi_1 \wedge \psi_2$. Hence, $p \sqsubseteq_{\mathrm{RS}} \psi_1 \wedge \psi_2$.

  ("$\Longleftarrow$") Let $p \sqsubseteq_{\mathrm{RS}} \psi_1 \wedge \psi_2$ and $p \overset{\epsilon}{\Longrightarrow}\!\!| \, p_0$. Thus, $p_0 \sqsubseteq_{\mathrm{RS}} \psi_1 \wedge \psi_2$ and, by Prop. 5(2), we can now conclude that $p_0 \sqsubseteq_{\mathrm{RS}} \psi_1$ and $p_0 \sqsubseteq_{\mathrm{RS}} \psi_2$. Hence, by induction hypothesis, $p_0 \models \psi_1$ and $p_0 \models \psi_2$ and thus $p \models \psi_1 \wedge \psi_2$.

- $\underline{\phi = \Box\psi:}$ Recall that $\overset{\mathcal{A}}{\Longrightarrow}\!\!|$ stands for $\bigcup_{a \in \mathcal{A}} \overset{a}{\Longrightarrow}\!\!|$. In this part of the proof, we write $p \overset{\mathcal{A}^*}{\Longrightarrow}\!\!| \, p'$ whenever $p \overset{\epsilon}{\Longrightarrow}\!\!| \, p_0 \overset{\mathcal{A}}{\Longrightarrow}\!\!| \, p_1 \ldots \overset{\mathcal{A}}{\Longrightarrow}\!\!| \, p_n = p'$ with $n \geq 0$.

  ("$\Longrightarrow$") We first prove that

  $$\mathcal{R} =_{\mathrm{df}} \{\langle p'', \vec{q}\rangle \mid p \overset{\mathcal{A}^*}{\Longrightarrow}\!\!| \, p'',\ \vec{q} \in \Box\Psi,\ \forall i.\, p'' \underset{\sim}{\sqsubset}_{\mathrm{RS}} q_i\}$$

  is a stable rs-relation. We verify Conds. (RS1)–(RS4) of Def. 4:

  (RS1) p'' and all q_i are stable, whence $\vec{q}$ is stable, too.

  (RS2) Here, it is sufficient to show that $W_1' \cup W_2'$ is a witness, where

  $$\begin{aligned} W_1' &=_{\mathrm{df}} \{\vec{q} \in \Box\Psi \mid \exists p''.\, p \overset{\mathcal{A}^*}{\Longrightarrow}\!\!| \, p'' \text{ and } \forall i.\, p'' \underset{\sim}{\sqsubset}_{\mathrm{RS}} q_i\} \\ W_2' &=_{\mathrm{df}} \{\vec{q} \in \Box\Psi \mid \exists \vec{q}' \in W_1'\, \forall i.\, q_i \overset{\epsilon}{\Longrightarrow}\!\!| \, q_i'\}\,. \end{aligned}$$

The proof is similar to the one of Lemma 14, except for the proof of (W3) in case $\alpha \neq \tau$. Here, $\vec{q} \xrightarrow{\alpha}$ means $\vec{q} \in W_1'$ and $q_i \xrightarrow{\alpha}$ for all i. Since $p'' \notin F$, we get $p'' \xrightarrow{\alpha}$, by Cond. (RS4), and $p'' \overset{\alpha}{\Longrightarrow\!|}\, p'''$. By Cond. (RS3), there exist q_i'' and q_i' such that $q_i \xrightarrow{\alpha}_{\mathrm{F}} q_i'' \overset{\epsilon}{\Longrightarrow\!|}\, q_i'$ and $p''' \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} q_i'$. Moreover, $p''' \models \psi$, whence $p''' \sqsubseteq_{\mathrm{RS}} \psi$ by induction hypothesis, and therefore $p''' \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} \psi_0$ for some $\psi \overset{\epsilon}{\Longrightarrow\!|}\, \psi_0$. Thus, $(q_1', \ldots, q_n', \psi_0) \in W_1'$ and $\vec{q} \xrightarrow{\alpha} (q_1'', \ldots, q_n'', \psi) \in W_2'$.

(RS3) Let $p'' \overset{a}{\Longrightarrow\!|}\, p'''$. Then, for some q_i', $q_i \overset{a}{\Longrightarrow\!|}\, q_i'$ and $p''' \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} q_i'$ by Cond. (RS3) for $p'' \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} q_i$. Furthermore, $p \overset{\mathcal{A}^*}{\Longrightarrow\!|}\, p'''$ implies $p''' \models \psi$, i.e., by induction hypothesis, $p''' \sqsubseteq_{\mathrm{RS}} \psi$ and $p''' \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} \psi_0$ for some $\psi \overset{\epsilon}{\Longrightarrow\!|}\, \psi_0$. Thus, $\vec{q} \xrightarrow{a} (q_1'', \ldots, q_n'', \psi) \overset{\epsilon}{\Longrightarrow} (q_1', \ldots, q_n', \psi_0) \overset{\tau}{\not\longrightarrow}$, for suitably chosen $q_1'', \ldots, q_n''$, and $\langle p''', (q_1', \ldots, q_n', \psi_0)\rangle \in \mathcal{R}$. Therefore, we have $(q_1', \ldots, q_n', \psi_0) \in W_1'$, and all processes along the computation are in W_2'. By Prop. 13, this proves $\vec{q} \overset{a}{\Longrightarrow\!|}\, (q_1', \ldots, q_n', \psi_0)$.

(RS4) Let $p'' \notin F$. Then, Cond. (RS4) for $p'' \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} q_i$ yields $\mathcal{I}(p'') = \mathcal{I}(q_i)$ for all i, i.e., $\mathcal{I}(p'') = \mathcal{I}(\vec{q})$ by the definition of $\Box\Psi$.

Now, $p \sqsubseteq_{\mathrm{RS}} \Box\psi$ by the following. Firstly, $p \models \psi$ implies $p \sqsubseteq_{\mathrm{RS}} \psi$ by induction hypothesis. Together with $p \overset{\epsilon}{\Longrightarrow\!|}\, p_0$, this guarantees the existence of some ψ_0 such that $\psi \overset{\epsilon}{\Longrightarrow\!|}\, \psi_0$ and $p_0 \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} \psi_0$. Then, $(\psi) \overset{\epsilon}{\Longrightarrow\!|}\, (\psi_0)$ in $\Box\Psi$ and $\langle p_0, (\psi_0)\rangle \in \mathcal{R}$. Thus, $p \sqsubseteq_{\mathrm{RS}} (\psi) = \Box\psi$.

("$\Longleftarrow$") Let $p \overset{\mathcal{A}^*}{\Longrightarrow\!|}\, p'$. Then, by $p \sqsubseteq_{\mathrm{RS}} \Box\psi$, there exists some $\vec{\psi}'$ such that $(\psi) \overset{\mathcal{A}^*}{\Longrightarrow\!|}\, \vec{\psi}'$ (performing the same sequence of visible actions) and $p' \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} \vec{\psi}'$. By Lemma 16, we have $p' \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} \psi_i'$ for all i. By our operational rules, the last component ψ' of $\vec{\psi}'$ is such that $\psi \overset{\epsilon}{\Longrightarrow\!|}\, \psi'$. Hence, $p' \sqsubseteq_{\mathrm{RS}} \psi$ and, by induction hypothesis, $p' \models \psi$. Thus, $p \overset{\mathcal{A}^*}{\Longrightarrow\!|}\, p'$ implies $p' \models \psi$, i.e., $p \models \Box\psi$. □

The classic property of entailment is now a corollary to Thm. 15:

Corollary 17 (Entailment). $\phi \sqsubseteq_{RS} \psi \iff \forall p.\ p \models \phi \implies p \models \psi$.

Proof. Let $\phi \sqsubseteq_{\mathrm{RS}} \psi$ and $p \models \phi$. Then, $p \sqsubseteq_{\mathrm{RS}} \phi \sqsubseteq_{\mathrm{RS}} \psi$ by Thm. 15, and we are done by transitivity of $\sqsubseteq_{\mathrm{RS}}$. Conversely, if $p \models \phi$ implies $p \models \psi$, then $p \sqsubseteq_{\mathrm{RS}} \phi$ implies $p \sqsubseteq_{\mathrm{RS}} \psi$, again by Thm. 15, for all p. Hence, $\phi \sqsubseteq_{\mathrm{RS}} \psi$ when setting $p = \phi$. □

3.3. Laws for Logic LTS

Our setting of LLTS satisfies many desirable, and often expected, laws. Firstly, considering the "process-algebraic" fragment of LLTS, e.g., our CSP-style parallel composition operator $\|_A$ is commutative and associative for fixed action sets $A \subseteq \mathcal{A}$, as can easily be seen from its definition (cf. Def. 2).

Regarding the propositional-logic fragment, we first recall that $\vee$ is disjunction and $\wedge$ is conjunction (cf. Prop. 5). Furthermore, disjunction and conjunction are commutative and associative. Note that associativity of conjunction follows from Prop. 5(2): $r \sqsubseteq_{\mathrm{RS}} (p_1 \wedge p_2) \wedge p_3 \Leftrightarrow r \sqsubseteq_{\mathrm{RS}} p_1$ and $r \sqsubseteq_{\mathrm{RS}} p_2$ and $r \sqsubseteq_{\mathrm{RS}} p_3 \Leftrightarrow r \sqsubseteq_{\mathrm{RS}} p_1 \wedge (p_2 \wedge p_3)$. Applying this equivalence for

Table 1: LLTS laws of propositional logic

(1a,b)	$p \wedge p =_{\mathrm{RS}} p$		$p \vee p =_{\mathrm{RS}} p$		*(Idempotence)*
(2a,b)	$p \wedge (p \vee q) =_{\mathrm{RS}} p$		$p \vee (p \wedge q) =_{\mathrm{RS}} p$		*(Absorption)*
(3a,b)	$p \vee ff =_{\mathrm{RS}} p$		$p \wedge tt =_{\mathrm{RS}} p$		*(Neutral elements)*
(4a,b)	$p \wedge ff =_{\mathrm{RS}} ff$		$p \vee tt =_{\mathrm{RS}} tt$		*(Null elements)*
(5a,b)	$p \wedge q \sqsubseteq_{\mathrm{RS}} p$		$p \sqsubseteq_{\mathrm{RS}} p \vee q$		
(6)	$p \wedge q =_{\mathrm{RS}} p$	$\Leftrightarrow$	$p \vee q =_{\mathrm{RS}} q$	$\Leftrightarrow$	$p \sqsubseteq_{\mathrm{RS}} q$
(7a,b)	$ff \sqsubseteq_{\mathrm{RS}} p$		$p \sqsubseteq_{\mathrm{RS}} tt$		

Table 2: LLTS laws of temporal logic

(8)	$[a](p \wedge q) =_{\mathrm{RS}} [a]p \wedge [a]q$	(13)	$en(a) \vee dis(a) =_{\mathrm{RS}} tt$
(9)	$\Box(p \wedge q) =_{\mathrm{RS}} \Box p \wedge \Box q$	(14)	$en(a) \wedge dis(a) =_{\mathrm{RS}} ff$
(10)	$\Box p =_{\mathrm{RS}} p \wedge [\mathcal{A}](\Box p)$	(15)	$dis(a) \wedge [a]p =_{\mathrm{RS}} dis(a)$
(11)	$\Box p =_{\mathrm{RS}} p \,\mathsf{W}\, ff$		
(12)	$p \,\mathsf{W}\, q =_{\mathrm{RS}} q \vee (p \wedge [\mathcal{A}](p \,\mathsf{W}\, q))$		

$r = (p_1 \wedge p_2) \wedge p_3$ and $r = p_1 \wedge (p_2 \wedge p_3)$ shows the claim. In addition, disjunction and conjunction are distributive and satisfy the standard laws of propositional logic shown in Table 1. Distributivity and Laws (1), (4a), (5) and (6) are proved in [2]. Laws (2a) and (2b) can be shown with Prop. 5(2) and (3), distributivity and idempotence. Laws (3a) and (7a) are direct from the definitions. For Law (7b) we have already argued in Sec. 3.2; its validity can also be checked by consulting Thm. 15. Finally, Laws (3b) and (4b) follow from Laws (6) and (7b).

For the temporal-logic fragment we have the laws in Table 2. These are standard except for Laws (13)–(15) which involve the atomic propositions $en(a)$ and $dis(a)$. The notation $[\mathcal{A}]$ has to be understood as the conjunction over $[a]$ with $a \in \mathcal{A}$, i.e., we assume here that the alphabet $\mathcal{A}$ is finite; of course, one can also generalise $[a]$ to $[A]$ for arbitrary sets A of actions. All laws in Table 2 can be proved by appealing to the satisfaction relation (cf. Def. 7) and entailment (cf. Cor. 17). This argument works only if p and q in Table 2 are temporal formulas. But in fact, the laws are also valid for general LLTSs. To see this, we have to generalise our results on compatibility and entailment. We first define an extended satisfaction relation $\models'$ for general LLTS p and such LLTS q that have a logic operator at top-level, i.e.,

$$tt,\ ff,\ en(a),\ dis(a),\ p_1 \vee p_2,\ p_1 \wedge p_2,\ [a]p',\ \Box p,\ \text{or}\ p_1 \,\mathsf{W}\, p_2$$

as in Def. 7 but with $\models$ replaced by $\sqsubseteq_{\mathrm{RS}}$ in the if-clauses. Then, Thm. 15 and hence Cor. 17 also hold for $\models'$. This is because the proof of Thm. 15 works for $\models'$, too; it repeatedly appeals to induction to conclude $p \sqsubseteq_{\mathrm{RS}} \phi$ from $p \models \phi$ (or vice versa), and this can simply be omitted when dealing with $\models'$. Now the laws in Table 2 can be proved by referring to $\models'$ and employing Cor. 17 for $\models'$; in other words, temporal-logic arguments within the LLTS framework can directly be lifted to our "process-algebraic" setting.

Next, we turn our attention to laws of the form $p \parallel_A q \sqsubseteq_{\mathrm{RS}} r$ which mix process-algebraic operators and temporal-logic operators in case p, q and r contain a logic operator. Such laws support *modular verification* as is shown by the following result:

Theorem 18 (Modular verification). *Let ϕ_1, ϕ_2 and ϕ_3 be temporal-logic formulas and $S \subseteq \mathcal{A}$ a synchronisation set. Then, $\phi_1 \parallel_S \phi_2 \sqsubseteq_{RS} \phi_3$ if and only if $p \parallel_S q \models \phi_3$ for all LLTS processes $p \models \phi_1$ and $q \models \phi_2$.*

The appeal of this theorem is that one can check $\phi_1 \parallel_S \phi_2 \sqsubseteq_{\mathrm{RS}} \phi_3$ without considering all processes p and q. In addition, we do not have to develop a separate temporal-logic counterpart to process-algebraic parallel composition.

Proof. "$\Longrightarrow$": If $p \models \phi_1$ and $q \models \phi_2$ then $p \parallel_S q \sqsubseteq_{\mathrm{RS}} \phi_1 \parallel_S \phi_2 \sqsubseteq_{\mathrm{RS}} \phi_3$ by compatibility, compositionality and assumption. Thus, $p \parallel_S q \models \phi_3$ by compatibility again.

"$\Longleftarrow$": Choose $p =_{\mathrm{df}} \phi_1$ and $q =_{\mathrm{df}} \phi_2$. Then, $p \sqsubseteq_{\mathrm{RS}} \phi_1$ and $q \sqsubseteq_{\mathrm{RS}} \phi_2$; $p \models \phi_1$ and $q \models \phi_2$ by compatibility; $p \parallel_S q \models \phi_3$ by assumption; and $\phi_1 \parallel_S \phi_2 \sqsubseteq_{\mathrm{RS}} \phi_3$ by compatibility again. $\Box$

To illustrate Thm. 18, we prove the following two simple instances:

$$en(a) \parallel_S en(a) \quad \sqsubseteq_{\mathrm{RS}} \quad en(a) \tag{1}$$

$$dis(a) \parallel_S tt \quad \sqsubseteq_{\mathrm{RS}} \quad dis(a) \qquad \text{if } a \in S \tag{2}$$

For the proof of Instance (1), first note that $en(a) \parallel_S en(a)$ can stabilise only to some $A \parallel_S A'$ with $a \in A \cap A'$ (cf. Fig. 2(c)). Process $en(a)$ can match this by stabilising to the process $p =_{\mathrm{df}} (A \cap A') \cup (A \cup A') \setminus S$, since this set contains action a. The transitions of $A \parallel_S A'$ are of the form $A \parallel_S A' \xrightarrow{b} tt \parallel_S tt$ for $b \in S \cap A \cap A'$ and, without loss of generality, $A \parallel_S A' \xrightarrow{b} tt \parallel_S A'$ for $b \in A \setminus S$. These transitions can be matched by $p \xrightarrow{b} tt$ since $tt \parallel_S tt \sqsubseteq_{\mathrm{RS}} tt$ and $tt \parallel_S A \sqsubseteq_{\mathrm{RS}} tt$ by Law (7b) in Table 1, as desired. The proof of Instance (2) is analogous, except that $a \notin A$ and hence $a \notin (A \cap A') \cup (A \cup A') \setminus S$ by assumption.

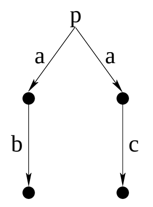

Figure 4: Example LLTS.

As an aside, we observe that laws like $p \wedge (q \parallel_{\mathcal{A}} r) =_{\mathrm{RS}} (p \wedge q) \parallel_{\mathcal{A}} (p \wedge r)$ do not hold. Consider $q =_{\mathrm{df}} r =_{\mathrm{df}} tt$ and p as in Fig. 4, for which $p \wedge (tt \parallel_{\mathcal{A}} tt) =_{\mathrm{RS}} p \wedge tt =_{\mathrm{RS}} p$ cannot deadlock after action a while $(p \wedge tt) \parallel_{\mathcal{A}} (p \wedge tt) =_{\mathrm{RS}} p \parallel_{\mathcal{A}} p$ can.

3.4. Duality

We conclude this section by briefly discussing negation. Since our setting of LLTS is not expressive enough to encode liveness properties, such as the formula $\neg\Box\phi$, we do not have negation. Furthermore, Thm. 15 implies for the stable process ff that $ff \models tt$ and $ff \models ff$. Hence, we cannot define "$p \models \neg tt$ if not $p \models tt$" for inconsistent p, since $\neg tt$ should be equivalent to ff. However, for *consistent* processes and propositional formulas, we can express negation in our $\neg$-less logic. To show this, we define for consistent p and propositional ϕ: $p \models \neg\phi$ if $\forall p_0.\ \ p \overset{\epsilon}{\Longrightarrow}\!| \, p_0 \implies$ not $p_0 \models \phi$; as well as for formulas ϕ and ψ: $\phi \mathrel{=\!\!|\!\models} \psi$ if $\forall p \notin F.\ \ p \models \phi \iff p \models \psi$. Now, the proof of the following proposition is an easy exercise.

Proposition 19 (Dualities).

$\neg tt$	$=\!\!\mid\!\models$	ff	$\neg en(a)$	$=\!\!\mid\!\models$	$dis(a)$	$\neg(\phi \wedge \psi)$	$=\!\!\mid\!\models$	$\neg\phi \vee \neg\psi$
$\neg ff$	$=\!\!\mid\!\models$	tt	$\neg dis(a)$	$=\!\!\mid\!\models$	$en(a)$	$\neg(\phi \vee \psi)$	$=\!\!\mid\!\models$	$\neg\phi \wedge \neg\psi$

As a consequence, we can specify implications for consistent processes, e.g., $en(a) \longrightarrow dis(b)$ can be expressed as $dis(a) \vee dis(b)$. Finally, note that one cannot replace $=\!\!\mid\!\models$ by $=_{\mathrm{RS}}$ in Prop. 19 since $=_{\mathrm{RS}}$ also relates inconsistent processes.

4. Example

We illustrate the utility of our setting involving mixtures of process-algebraic operators and temporal-logic operators via a small example. Consider the specification of a very simple networking component. Sender S (cf. Fig. 5) receives messages from a user process on port `send` and passes them on, via port `in`, to channel C. The specification of C employs an off-the-shelf design P (cf. Fig. 5), a generic channel that may lose messages; additionally, the behaviour of P is restricted by a constraint $\psi =_{\mathrm{df}} \Box[\texttt{in}][\texttt{in}](en(\texttt{out}) \wedge dis(\texttt{in}))$. Intuitively, ψ ensures that at most one message may be lost in a row.

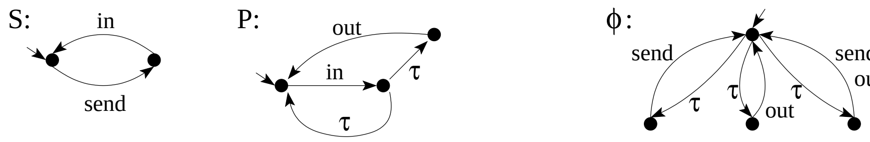

Figure 5: Some LLTSs that occur in the example.

As an aside and assuming the availability of the standard process-algebraic prefix operator, ψ could equivalently be specified as $\Box[\texttt{in}][\texttt{in}]\texttt{out}.tt$, where $\texttt{out}.tt$ denotes the LLTS consisting of an `out`-transition from an initial state to process tt. Here, prefixing is employed as a compact notation for specifying that only a single action is allowed, which is especially useful (or even necessary) if the underlying alphabet is large (or infinite). This demonstrates one of the advantages of mixing operators from process algebras and temporal logics.

The overall specification of our example is now $\texttt{Spec} =_{\mathrm{df}} ((P \wedge \psi) \parallel_{\{\texttt{in}\}} S)/\texttt{in}$, where $/\texttt{in}$ is a *hiding operator* on action `in`, similar to the identically named operator in CSP [13], which restricts the scope of `in` to `Spec` (cf. [2] for details). `Spec` is a truly *mixed* specification that conjunctively composes an operational component with a temporal-logic formula, and puts the result in parallel with another operational component while synchronising on the internal channel `in`. The LLTS semantics of `Spec` is successively developed in Fig. 6: (a) depicts the LLTS of $[\texttt{in}][\texttt{in}]\texttt{out}.tt$; (b) depicts the LLTS of ψ when reduced with respect to $=_{\mathrm{RS}}$ (recall that there is a standard finite-state definition of the $\Box$-operator); (c) depicts the LLTS of $C =_{\mathrm{df}} P \wedge \psi$ as well as a simplified, $=_{\mathrm{RS}}$-equivalent version; and (d) depicts the simplified LLTS (omitting inconsistent states) of `Spec`, where label (`in`) stands for a τ that results from hiding action `in` [2].

Assume that the designer wishes to verify that `Spec` does not deadlock, i.e., always `send` or `out` is enabled: $\phi =_{\mathrm{df}} \Box(en(\texttt{send}) \vee en(\texttt{out}))$. To demonstrate $\texttt{Spec} \models \phi$, it is by Thm. 15 sufficient to prove $\texttt{Spec} \sqsubseteq_{\mathrm{RS}} \phi$. This is easy when considering the LLTSs of `Spec` and ϕ, which are depicted in Figs. 6(d) and 5. We also know that, whenever we implement the channel design $C = P \wedge \psi$ by some C_i so that $C_i \sqsubseteq_{\mathrm{RS}} C$, the implementation $\texttt{Impl} =_{\mathrm{df}} (C_i \parallel_{\{\texttt{in}\}} S)/\texttt{in}$

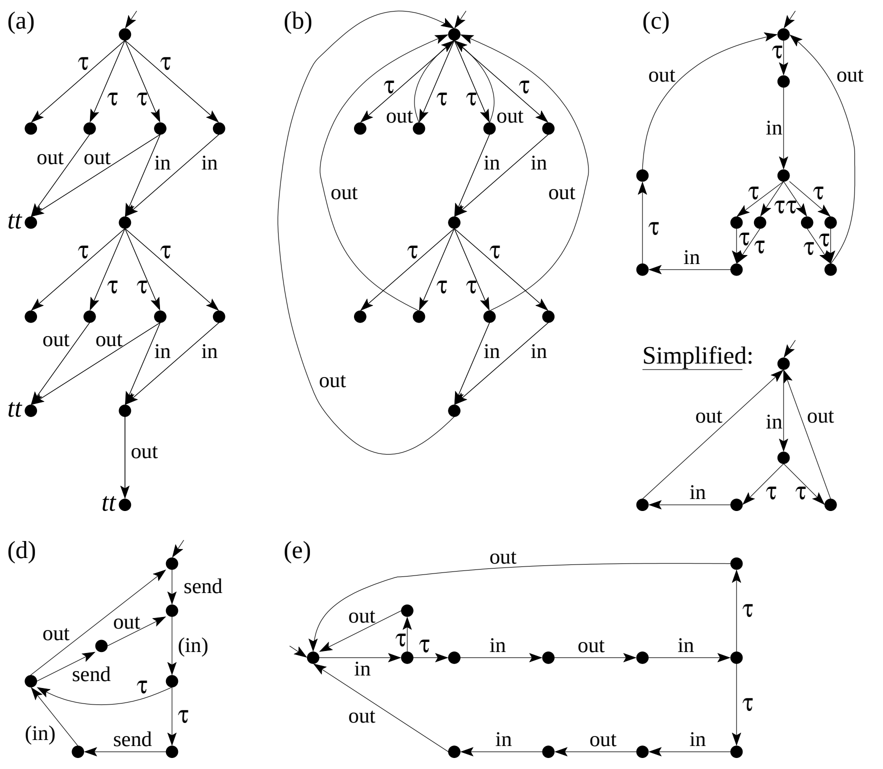

Figure 6: (a)–(d): Developing the LLTS of `Spec`; (e): Possible implementation C_3 of C.

satisfies ϕ, too. This is because `Impl` $\sqsubseteq_{\text{RS}}$ `Spec` by compositionality and Prop. 5(2); thus, by transitivity, `Impl` $\sqsubseteq_{\text{RS}} \phi$. Hence, `Impl` $\models \phi$ by Thm. 15.

Possible implementations C_i of C include the LTS C_1 that engages in an `in`-`out`-loop, C_2 that behaves as an `in`-`in`-`out`-loop, or C_3 depicted in Fig. 6(e); the latter requires that at most one of each two messages and at most two of five messages are lost. Rather than proving $C_3 \sqsubseteq_{\text{RS}} C$, one could establish $C_3 \sqsubseteq_{\text{RS}} P$ and $C_3 \sqsubseteq_{\text{RS}} \psi$ separately, and then infer $C_3 \sqsubseteq_{\text{RS}} P \wedge \psi = C$ by Prop. 5(2).

5. Modal Logic LTS

If one wishes to write an LLTS specification that permits a large number of ready sets initially, one needs to insert a τ-branch for each single one of these ready sets. This can be seen, e.g., in Fig. 2 and leads to a cluttered and sometimes difficult to comprehend presentation of the desired specification. A more compact representation can, however, be achieved by employing *may*- and *must*-transitions, as inspired by the *modal transition systems* of Larsen [7].

In this section, we first introduce *modal LLTS* as a shorthand notation for LLTS, and apply them to the embedding of temporal logic into our setting and to our example above (cf. Sec. 5.1). This paves the way for comparing our setting to the one of Larsen, for which we adapt ready simulation to modal LLTS and show that the resulting *modal ready simulation* is finer than ready simulation but coarser than Larsen's *modal refinement* [7] (cf. Sec. 5.2). We also provide some intuitive insights behind *may* and *must* in our setting (cf. Sec. 5.3), and discuss conjunction operators in modal transition systems in the light of our work (cf. Sec. 5.4).

5.1. Definition, Expansion & Application

In modal LLTS we distinguish required transitions $p \xrightarrow{\alpha} p'$, called *must-transitions*, and allowed transitions $p \overset{\alpha}{\dashrightarrow} p'$, called *may-transitions*. We demand *syntactic consistency*, i.e., every required transition must also be allowed. In the following, we write $p \overset{\alpha}{\not\dashrightarrow}$ for $\nexists p'.\, p \overset{\alpha}{\dashrightarrow} p'$. Analogously, $p \overset{\alpha}{\not\longrightarrow}$ if $\nexists p'.\, p \xrightarrow{\alpha} p'$; note that the absence of some must-transition $p \xrightarrow{\alpha} p'$ does not preclude the existence of the may-transition $p \overset{\alpha}{\dashrightarrow} p'$.

Definition 20 (Modal LLTS). *Consider a quadruple $\langle P, \longrightarrow, \dashrightarrow, F\rangle$ such that (i) P is a set of states or processes, (ii) $\longrightarrow \subseteq \dashrightarrow \subseteq P{\times}\mathcal{A}_\tau{\times}P$, i.e., every must-transition is also a may-transition, (iii) $\dashrightarrow \cap\, (P \times \{\tau\} \times P) \subseteq \longrightarrow$, i.e., every τ-transition is a may-transition* and *a must-transition, and (iv) $F \subseteq P$.*

We define $\mathcal{I}may(p) =_{df} \{\alpha \in \mathcal{A}_\tau \,|\, p \overset{\alpha}{\dashrightarrow}\}$ for modal LLTS and, analogously, write $\mathcal{I}must(p)$ for $\{\alpha \in \mathcal{A}_\tau \,|\, p \xrightarrow{\alpha}\}$; obviously, $\mathcal{I}must(p) \subseteq \mathcal{I}may(p)$. In addition, we define $\overset{\alpha}{\Longrightarrow}\!|$ as before but based on the may-transition relation $\dashrightarrow$.

Then, the above quadruple $\langle P, \longrightarrow, \dashrightarrow, F\rangle$ is a modal LLTS *if it satisfies the following three conditions:*

(τ-purity) $\forall p \in P.\ p \xrightarrow{\tau} \ \Longrightarrow\ \forall a \in \mathcal{A}.\ p \overset{a}{\not\dashrightarrow}$;

(mLTS1) *$p \in F$ if $\exists \alpha {\in} \mathcal{I}may(p).\ (p \xrightarrow{\alpha}$ and $\forall p' \in P.\ p \xrightarrow{\alpha} p' \Longrightarrow p' \in F)$ or*
$(p \overset{\alpha}{\not\longrightarrow}$ and $\exists p' \in F.\ p \overset{\alpha}{\dashrightarrow} p')$;

(mLTS2) *p cannot stabilise (i.e., $\nexists p' \in P.\ p \overset{\epsilon}{\Longrightarrow}\!| p'$) $\ \Longrightarrow\ p \in F$.*

In drawings of examples later on, we let an ordinary arrow represent a may-transition *and* a must-transition; may-transitions that are not also must-transitions are drawn as dashed arrows. Analogous to [7], syntactic consistency is formalised by requiring $\longrightarrow \subseteq \dashrightarrow$. The details of Cond. (mLTS1) are justified by the following expansion of modal LLTS to LLTS (cf. Def. 21 and Remark 24), which explains modal LLTS as a shorthand notation for LLTS. Further insights regarding the intuition of *may* and *must* in modal LLTS will be offered in Sec. 5.3.

The idea behind the expansion of modal LLTS to LLTS is to replace each process p by a disjunction, where each disjunct captures all must-transitions and some may-transitions of p such that each collection of may-transitions that are not also must-transitions is represented. This clearly reflects the intuition of *may*- and *must*-transitions.

Definition 21 (Expansion). *Let P be a modal LLTS and, for $p \in P$, let $MO(p)$ denote the set $\{p \overset{\alpha}{\dashrightarrow} p' \,|\, p \overset{\alpha}{\not\longrightarrow} p'\}$ of all* may-only *transitions of p, i.e., all outgoing may-transitions of p that are not also must-transitions. Note that $(p \overset{\alpha}{\dashrightarrow} p') \in MO(p)$ implies $\alpha \neq \tau$ by (iii) in Def. 20.*

To construct the expansion LLTS $\hat{P}$ of P, we (i) add to P processes of the form (p, M), for each $p \in P$ and $M \subseteq MO(p)$, (ii) define $\hat{F} =_{df} F \cup \{(p, M) \in \hat{P} \,|\, p \in F\}$, and (iii) replace the outgoing transitions of each process $p \in P$ by the following new transitions:

- $p \xrightarrow{\tau} (p, M)$ *for $M \subseteq MO(p)$, and*
- $(p, M) \xrightarrow{\alpha} p'$ *for $(p \overset{\alpha}{\dashrightarrow} p') \in M$ or $p \xrightarrow{\alpha} p'$.*

Before presenting an example of such an expansion, we briefly remark on an alternative definition where we replace a process $p \in P$ and its outgoing transitions only if $\mathsf{MO}(p) \neq \emptyset$ and, hence, $\mathcal{I}may(p) \neq \emptyset$. The resulting expansion is $=_{\mathrm{RS}}$-equivalent to the one obtained by applying Def. 21. The advantage of this alternative definition is a practical one, namely that the resulting expansion is smaller. Indeed, the expansion is P itself if may- and must-transitions coincide, i.e., LLTSs are preserved. In contrast, every process p where all outgoing transitions are must-transitions, is split into $p \xrightarrow{\tau} (p, \emptyset)$ in our definition. The disadvantage of the alternative definition concerns proofs requiring the expansion construction. This is because there would be two cases to consider for each process $p \in P$, which can lead to many subcases in proofs in which one has to compare several may-transitions. For this reason we prefer Def. 21.

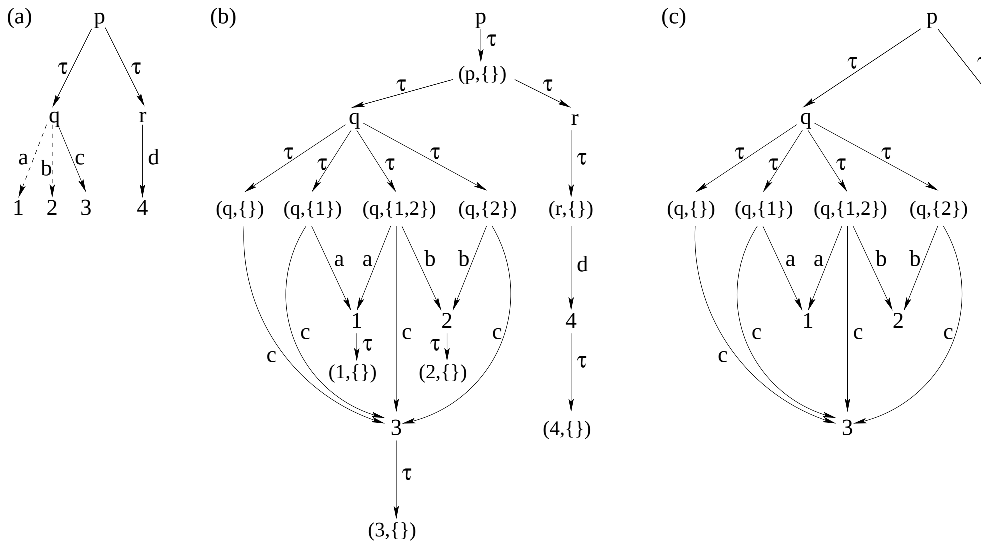

Figure 7: Expansion example: (a) Modal LLTS P; (b) Expansion LLTS $\hat{P}$; (c) Alternative expansion LLTS.

We now turn to an example of our expansion construction. Consider the modal LLTS P in Fig. 7(a). Its expansion $\hat{P}$ is depicted in Fig. 7(b), where we represent, in the states (q, M), the elements $q \overset{a}{\dashrightarrow} 1$ and $q \overset{b}{\dashrightarrow} 2$ of $\mathsf{MO}(q)$ by 1 and 2. For completeness, the result of applying our alternative expansion is depicted in Fig. 7(c). This example also shows that it is convenient that all τ-transitions are must-transitions: if the initial part of P were as shown in Fig. 8(a), then this would be translated to Fig. 8(b) which just represents $q \vee r$ as well, but in a more complex way.

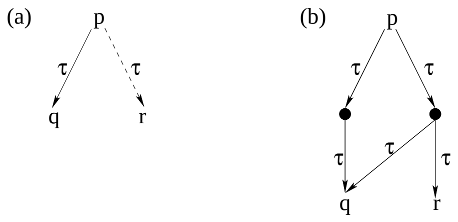

Figure 8: Illustration that demonstrates the convenience of the fact that all τ-transitions are must-transitions.

In addition, observe that, with our interpretation of an instable state as a disjunction, it is sufficient to implement one of its outgoing τ-(must-)transitions. Thus, these transitions correspond to one *disjunctive must-transition* as in [14]. Before proving that the expansion of a modal LLTS is indeed an LLTS, we first state an easy lemma which will be used several times in this section and is proved in the appendix:

Lemma 22. *Let P be a modal LLTS.*

1. *If $p \overset{\epsilon}{\Longrightarrow\!|}\, p'$ in P, then $p \overset{\epsilon}{\Longrightarrow\!|}\, (p', M')$ in $\hat{P}$ whenever $(p', M') \in \hat{P}$.*
2. *If $p \overset{\epsilon}{\Longrightarrow\!|}\, (p', M')$ in $\hat{P}$, then $p \overset{\epsilon}{\Longrightarrow\!|}\, p'$ in P.*

The details of Def. 20, and in particular of Cond. (mLTS1) therein, are tuned to make the expansion $\hat{P}$ of P well-defined, i.e., $\hat{P}$ is an LLTS without the need for any backward propagation.

Proposition 23 (Well-Definedness). *Given a modal LLTS P, its expansion $\hat{P}$ is an LLTS.*

Proof. We check the requirements of Def. 1. Firstly, all $p \in P$ are instable in $\hat{P}$; all processes (p, M) are stable if p is stable, and they only have τ-transitions if p is instable.

Regarding Cond. (LTS1) and $p \in P$, we have that p is in $F \subseteq \hat{F}$ or all $(p, M) \notin \hat{F}$. For processes of the form (p, M) we assume $\exists \alpha \in \mathcal{I}((p, M)) \forall p'. (p, M) \xrightarrow{\alpha} p' \implies p' \in \hat{F}$ (i.e., $p' \in F$) and distinguish the following cases:

$p \overset{\alpha}{\not\longrightarrow}$**:** Then, $\exists (p \overset{\alpha}{\dashrightarrow} p') \in M$ by $\alpha \in \mathcal{I}((p, M))$, and we have $(p, M) \xrightarrow{\alpha} p'$. Hence, $p' \in \hat{F}$ by assumption. Thus, the second disjunct of Cond. (mLTS1) holds for α, i.e., $p \in F$ and $(p, M) \in \hat{F}$.

$p \xrightarrow{\alpha} p'$**:** Here we have $\forall p'. p \xrightarrow{\alpha} p' \implies (p, M) \xrightarrow{\alpha} p'$ by construction of $\hat{P}$, and $p' \in F$ by assumption. Thus, the first disjunct of Cond. (mLTS1) holds for α, whence $p \in F$ and $(p, M) \in \hat{F}$.

We now turn our attention to establishing Cond. (LTS2) of Def. 1. If $p \notin \hat{F}$ then $p \notin F$; hence, p can stabilise in P. Therefore, p can also stabilise in $\hat{P}$ by Lemma 22(1). If $(p, M) \notin \hat{F}$, i.e., $p \notin F$, then: (a) p and (p, M) are stable and we are done; or (b) p is not stable (i.e., $\alpha = \tau$), $M = \emptyset$ (since all τ-transitions are must-transitions), and $p \xrightarrow{\tau} (p, \emptyset)$. Due to $p \notin F$, we obtain in Case (b) that p can stabilise with some $p \overset{\epsilon}{\Longrightarrow\!|}\, p'$. Now, by Lemma 22(1), $p \overset{\epsilon}{\Longrightarrow\!|}\, (p', \emptyset)$ in $\hat{P}$; this involves $p \xrightarrow{\tau} (p, \emptyset)$, i.e., $(p, \emptyset) \overset{\epsilon}{\Longrightarrow\!|}\, (p', \emptyset)$. □

Remark 24. *For the interested reader, we now explain the details of Cond. (mLTS1) in Def. 20. First consider Fig. 9(a) and part of its expansion in Fig. 9(b), and contemplate the following simple adaptation of Cond. (LTS1): $p \in F$ if $\forall p'. p \overset{a}{\dashrightarrow} p' \implies p' \in F$. Then, $p \notin F$ is justified by the may-transition, but the disjunct $(p, \{\})$ of p in $\hat{P}$ must be in $\hat{F}$. This is a backward propagation in the construction of $\hat{P}$, which we want to avoid. The problem arises due to $(p, \{\})$ which represents all must-transitions; therefore we require $p \in F$ if $\exists \alpha \in \mathcal{I}may(p). p \xrightarrow{\alpha} \wedge \forall p'. p \xrightarrow{\alpha} p' \implies p' \in F$ (cf. the first disjunct in Cond. (mLTS1)). Thus, if $\alpha \in \mathcal{I}must(p)$ and $p \notin F$, then there is some $p' \notin F$ with $p \xrightarrow{\alpha} p'$, and all disjuncts of p in $\hat{P}$ also have an α-transition to p'.*

For the remaining case $\alpha \in \mathcal{I}may(p) \setminus \mathcal{I}must(p)$, consider Fig. 9(c) and part of its expansion in Fig. 9(d). In this case, we have the same problem with the second disjunct, i.e., just one α-may-transition to a process in F should force $p \in F$. This justifies the second disjunct in Cond. (mLTS1).

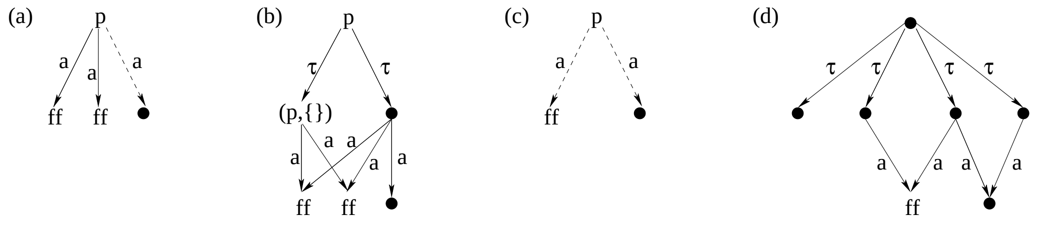

Figure 9: Illustrating the motivation behind Cond. (mLTS1).

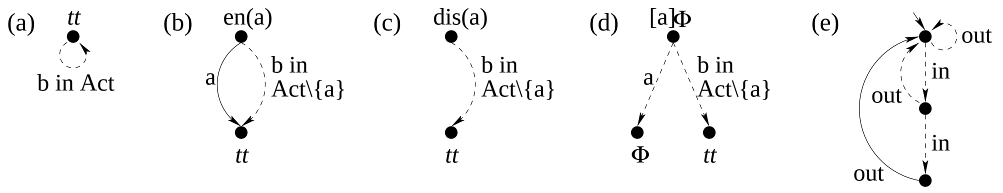

Figure 10: (a)–(d) Embedding of temporal-logic formulas into modal LLTS; (e) Compact representation of Fig. 6(b).

As an application of modal LLTS we show in Fig. 10 how the LLTSs for the formulas *tt*, *en*(a), *dis*(a), and $[a]\phi$ (cf. Fig. 2) can be represented more compactly as modal LLTSs, where 'Act' stands for $\mathscr{A}$. Compared to Fig. 2 we can do without the τ-transitions selecting the ready sets; note that there are exponentially many such transitions for finite $\mathscr{A}$. In addition and as a concrete example, we give a compact representation of Fig. 6(b) in Fig. 10(e). This modal LLTS requires only 3 instead of 11 processes and only 5 instead of 17 transitions, and shows much more clearly that any implementation of this specification must exhibit an `out` action after two `in` actions.

5.2. Ready Simulation & Modalities

Having defined modal LLTS as a shorthand notation for LLTS, it would be interesting to define ready simulation directly on modal LLTS. In this section we present a very natural candidate for such a variation. Surprisingly, this *modal ready simulation* turns out to be more strict than ready simulation which, however, still gives us a sound method for checking on modal LLTSs whether ready simulation holds for the LLTSs they stand for.

Another interesting question, given that we employ may- and must-transitions as a shorthand notation, is how our approach is related to the original refinement preorder, known as *modal refinement*, of modal transition systems [7]. To answer this question, we employ the modal ready simulation just mentioned and show that modal refinement implies modal ready simulation and thus ready simulation. For this comparison only, we will restrict ourselves to modal LLTS without τ actions, since τ has a special interpretation (related to disjunction) in our approach; hence, we identify modal transition systems with τ-free modal LLTS for which $F = \emptyset$. It is not surprising that the reverse implication, i.e., ready simulation implies modal refinement, does not hold since modal refinement is bisimulation-based.

We first introduce our notion of *modal ready simulation*. Recall that $\overset{\alpha}{\Longrightarrow\!\!|}$ is based on may-transitions, and note that $\overset{\epsilon}{\Longrightarrow\!\!|}$ could equivalently be based on must-transitions since all τ-transitions are may- *and* must-transitions.

Definition 25 (Modal Ready Simulation). *Let $\langle P, \longrightarrow_P, \dashrightarrow_P, F_P\rangle$ and $\langle Q, \longrightarrow_Q, \dashrightarrow_Q, F_Q\rangle$ be two modal LLTSs. Relation $\mathcal{R} \subseteq P \times Q$ is a* modal stable ready simulation relation, *or* modal stable rs-relation *for short, if the following conditions hold, for any $\langle p, q\rangle \in \mathcal{R}$ and $a \in \mathcal{A}$:*

(mRS1) *p, q stable* ***(mRS3)*** *$p \overset{a}{\Longrightarrow\!\!|}\, p' \implies \exists q'.\, q \overset{a}{\Longrightarrow\!\!|}\, q'$ and $\langle p', q'\rangle \in \mathcal{R}$*

(mRS2) *$p \notin F_P \implies q \notin F_Q$* ***(mRS4)*** *$p \notin F_P \implies \mathcal{I}may(p) \subseteq \mathcal{I}may(q) \wedge \mathcal{I}must(q) \subseteq \mathcal{I}must(p)$*

We write $p \underset{\sim}{\sqsubset}_{mRS} q$ if there exists a modal stable rs-relation $\mathcal{R}$ such that $\langle p, q\rangle \in \mathcal{R}$. Further, p is modal ready simulated *by q, in symbols $p \sqsubseteq_{mRS} q$, if $\forall p'.\, p \overset{\epsilon}{\Longrightarrow\!\!|}\, p' \implies \exists q'.\ q \overset{\epsilon}{\Longrightarrow\!\!|}\, q'$ and $p' \underset{\sim}{\sqsubset}_{mRS} q'$.*

Modal ready simulation has textually the same definition as ready simulation (cf. Def. 4), except for Cond. (RS4). Conds. (mRS1) and (mRS2) do not deal with transitions and thus stay unchanged. For Conds. (mRS3) and (mRS4), consider a modal LLTS P and some $p \in P$. Each step $p \overset{a}{\Longrightarrow\!\!|}\, p'$ (based on may-transitions) is a possible behaviour of p, so it must be matched as for LLTS. Regarding Cond. (mRS4), p represents all ready sets between $\mathcal{I}must(p)$ and $\mathcal{I}may(p)$, whence each of them must lie between $\mathcal{I}must(q)$ and $\mathcal{I}may(q)$ for a matching q. Observe that the inclusion $\mathcal{I}may(p) \subseteq \mathcal{I}may(q)$ already follows from Cond. (mRS3), as in Def. 4.

While Conds. (mRS1)–(mRS4) are the naturally expected ones, it is not clear that they – by themselves – treat the subtleties of may- and must-transitions in sufficient detail. Prop. 27 below shows, however, that this is indeed the case.

Remark 26. *The above definition of modal ready simulation somewhat reminds us of De Alfaro and Henzinger's* alternating simulation *for* interface automata *[15]. Alternating simulation is also a simulation, with additional requirements for initial actions. Their setting is, however, quite different from ours as it relies on an explicit distinction of input and output actions. Still, one merit of modal ready simulation is that it makes the vague conceptual similarity between our approach and interface automata more precise.*

The additional requirements of alternating simulation are that an implementation p allows all inputs of a matching q (corresponding to $\mathcal{I}must(q) \subseteq \mathcal{I}must(p)$), while it may only perform outputs allowed by q (corresponding to $\mathcal{I}may(p) \subseteq \mathcal{I}may(q)$). Of course, an important technical difference is that inputs and outputs are disjoint in the setting of interface automata, while $\mathcal{I}must(r) \subseteq \mathcal{I}may(r)$ for all processes r in our setting.

Another version of alternating refinement [16] (where simulation works one way for inputs and the other way for outputs) is very close to the so-called modal refinement *[7] (see Def. 29); this relation has been worked out in [17].*

We now prove that modal ready simulation on modal LLTS is finer than ready simulation on their LLTS expansions:

Proposition 27 (Expansion Preserves Refinement). *Let P and Q be modal LLTSs, and $p \in P$ and $q \in Q$. Then, $p \sqsubseteq_{mRS} q$ wrt. P and $Q \implies p \sqsubseteq_{RS} q$ wrt. $\hat{P}$ and $\hat{Q}$.*

Proof. We first show the following statement. Let $\mathcal{R} \subseteq P \times Q$ be a modal stable rs-relation, and define $\hat{\mathcal{R}} \subseteq \hat{P} \times \hat{Q}$ to consist of all pairs $\langle (p, M), (q, N)\rangle$ where

- $\langle p, q\rangle \in \mathcal{R}$ and $M \subseteq \mathsf{MO}(p)$;

- N is the set of all transitions $q \overset{a}{\dashrightarrow} q'$ in $\mathsf{MO}(q)$ such that there exist p', p'', q'' with (i) $(p \overset{a}{\dashrightarrow} p') \in M$ or $p \xrightarrow{a} p'$, (ii) $p \overset{a}{\dashrightarrow} p' \overset{\epsilon}{\Longrightarrow\!|} p''$, (iii) $q \overset{a}{\dashrightarrow} q' \overset{\epsilon}{\Longrightarrow\!|} q''$, and (iv) $\langle p'', q'' \rangle \in \mathcal{R}$.

The latter item ensures that (q, N) allows exactly the steps matching some $(p, M) \overset{a}{\Longrightarrow\!|}$. Now, we claim that $\hat{\mathcal{R}}$ is a stable rs-relation. For the proof, we consider some arbitrary $\langle (p, M), (q, N) \rangle \in \hat{\mathcal{R}}$ and $a \in \mathcal{A}$, and check the conditions stated in Def. 4:

(RS1) $\langle p, q \rangle \in \mathcal{R}$ implies, by Cond. (mRS1), that p and q are stable. Hence, processes (p, M) and (q, N) are stable.

(RS2) $(p, M) \notin \hat{F}$ implies, by the construction of $\hat{P}$, that $p \notin F$. Thus, $q \notin F$ by Cond. (mRS2) and, hence, $(q, N) \notin \hat{F}$.

(RS3) Let $(p, M) \overset{a}{\Longrightarrow\!|} (p'', M'')$; hence, there is some p' with $(p \overset{a}{\dashrightarrow} p') \in M$ or $p \xrightarrow{a} p'$, and $(p, M) \xrightarrow{a} p' \overset{\epsilon}{\Longrightarrow\!|} (p'', M'')$. Therefore, $p \overset{a}{\dashrightarrow} p' \overset{\epsilon}{\Longrightarrow\!|} p''$ in P (cf. Lemma 22(2)). By Cond. (mRS3), there exist q', q'' with $q \overset{a}{\dashrightarrow} q' \overset{\epsilon}{\Longrightarrow\!|} q''$ and $\langle p'', q'' \rangle \in \mathcal{R}$. Thus, $(q \overset{a}{\dashrightarrow} q') \in N$ or $q \xrightarrow{a} q'$ by the definition of $\hat{\mathcal{R}}$. Moreover, $(q, N) \xrightarrow{a} q'$ by the construction of $\hat{Q}$. Since $\langle p'', q'' \rangle \in \mathcal{R}$, there is a unique N'' with $\langle (p'', M''), (q'', N'') \rangle \in \hat{R}$, and we have $q' \overset{\epsilon}{\Longrightarrow\!|} (q'', N'')$ in $\hat{P}$ by Lemma 22(1). Together with $(q, N) \xrightarrow{a} q'$, this finishes this case.

(RS4) Let $(p, M) \notin \hat{F}$. The inclusion $\mathcal{I}((p, M)) \subseteq \mathcal{I}((q, N))$ is clear from Cond. (RS3) above. Now let $a \in \mathcal{I}((q, N))$, i.e., there exists a q' such that $(q \overset{a}{\dashrightarrow} q') \in N$ or $q \xrightarrow{a} q'$. In the case $(q \overset{a}{\dashrightarrow} q') \in N$, we have some p' with $(p \overset{a}{\dashrightarrow} p') \in M$ or $p \xrightarrow{a} p'$ by the definition of $\hat{\mathcal{R}}$, and thus $(p, M) \xrightarrow{a} p'$ in $\hat{P}$. In the case $q \xrightarrow{a} q'$, we have $a \in \mathcal{I}must(q) \subseteq \mathcal{I}must(p)$ by Cond. (mRS4), and $\mathcal{I}must(p) \subseteq \mathcal{I}((p, M))$ by the definition of $\hat{P}$.

We now prove the statement of the proposition and assume $p \sqsubseteq_{\text{mRS}} q$. Let $p \overset{\epsilon}{\Longrightarrow\!|} (p', M')$ in $\hat{P}$, i.e., $p \overset{\epsilon}{\Longrightarrow\!|} p'$ in P by Lemma 22(2). Then, there exists a q' with $q \overset{\epsilon}{\Longrightarrow\!|} q'$ and $p' \underset{\sim}{\sqsubset}_{\text{mRS}} q'$ due to some modal stable rs-relation $\mathcal{R}$. Consider $\hat{\mathcal{R}}$ as constructed above and the N' such that $\langle (p', M'), (q', N') \rangle \in \hat{\mathcal{R}}$. Since $q \overset{\epsilon}{\Longrightarrow\!|} q'$ in Q we get $q \overset{\epsilon}{\Longrightarrow\!|} (q', N')$ in $\hat{Q}$ by Lemma 22(1), which finishes the proof. □

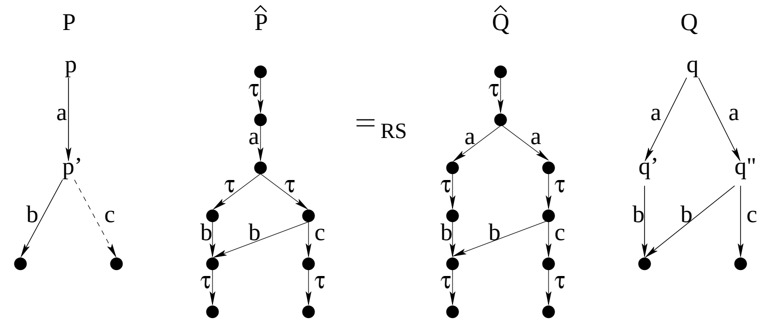

Figure 11: Counterexample: The inverse implication regarding Prop. 27 does not hold.

In the following, we also write $p \sqsubseteq_{\mathrm{RS}} q$ for processes p and q in modal LLTSs P and Q, respectively, if $p \sqsubseteq_{\mathrm{RS}} q$ wrt. $\hat{P}$ and $\hat{Q}$. As an aside, also note that $\sqsubseteq_{\mathrm{mRS}}$ and $\sqsubseteq_{\mathrm{RS}}$ coincide for those modal LLTS for which each may-transition is also a must-transition, i.e., for ordinary LLTS.

The reverse implication regarding Prop. 27 does, however, not hold as is testified by the counterexample depicted in Fig. 11. Observe that, in $\hat{P}$ and $\hat{Q}$, the steps $\overset{a}{\Longrightarrow\!|}$ choose between the two branches. This is not the case in P. Technically, when trying to prove $p \sqsubseteq_{\mathrm{mRS}} q$ in P and Q, one must match p' with q' or q''. Neither of these matches is possible since $c \in \mathcal{I}may(p') \setminus \mathcal{I}may(q')$ and $c \in \mathcal{I}must(q'') \setminus \mathcal{I}must(p')$. In order to give a characterisation for ready simulation on the level of modal LLTS, it seems one would have to relate p' with the set $\{q', q''\}$. Since this suggests that a characterisation will necessarily be complicated and less appealing for applications, we do not investigate this issue further here.

Lately, some researchers have shown interest in *deterministic* modal transition systems (see, e.g., [18]). Intuitively, determinism of a modal transition system means determinism of its may-transition relation, and thus also of its must-transition relation. Adapting this notion to modal LLTS, we say that P is deterministic if it is deterministic with respect to the transition relation $\Longrightarrow\!|_P$. This means that each $\overset{a}{\Longrightarrow\!|}_P$ step from some process $p \in P$ leads to the same process $q \in P$, so we can assume that there is a direct transition $p \overset{a}{\dashrightarrow} q$. In particular, P has no τ-transitions, and all $p \in P$ are stable. While the reverse implication regarding Prop. 27 is not valid in general as seen above, we now prove that it holds for deterministic modal LLTS.

Proposition 28 (Reverse of Prop. 27). *Let P, Q be* deterministic *modal LLTSs, and $p \in P$ and $q \in Q$. Then, $p \sqsubseteq_{RS} q$ wrt. $\hat{P}$ and $\hat{Q}$ $\implies$ $p \sqsubseteq_{mRS} q$ wrt. P and Q.*

Proof. In the sequel we write $\mathcal{I}(M)$, where $M \subseteq \mathsf{MO}(p)$ and $p \in P$, for the action set $\{a \in \mathcal{A} \,|\, \exists p'. (p \overset{a}{\dashrightarrow} p') \in M\}$. We also employ the notation $\mathcal{I}(N)$ analogously for $N \subseteq \mathsf{MO}(q)$ and $q \in Q$. Given deterministic modal LLTSs P, Q we first show that

$$\mathcal{R} =_{\mathrm{df}} \{\langle p, q\rangle \subseteq P \times Q \mid \exists N \subseteq \mathsf{MO}(q).\, \mathcal{I}(\mathsf{MO}(p)) \subseteq \mathcal{I}(N) \text{ and } (p, \mathsf{MO}(p)) \underset{\sim}{\sqsubset}_{\mathrm{RS}} (q, N)\}$$

is a modal stable rs-relation. To this end, consider some $\langle p, q\rangle \in \mathcal{R}$, i.e., $\mathcal{I}((p, \mathsf{MO}(p))) \subseteq \mathcal{I}((q, N))$ for a suitable $N \in \mathsf{MO}(q)$, and $a \in \mathcal{A}$. We check the conditions of Def. 25:

(mRS1) Since P, Q are deterministic, we have that p and q are stable.

(mRS2) $p \notin F$ implies $(p, \mathsf{MO}(p)) \notin F$, by the construction of $\hat{P}$. Thus, $(q, N) \notin \hat{F}$ by Cond. (RS2) and, hence, $q \notin F$.

(mRS3) Let $p \overset{a}{\Longrightarrow\!|} p'$ for some p', i.e., $p \overset{a}{\dashrightarrow} p'$ since P is deterministic. By expansion, we have both $(p, \mathsf{MO}(p)) \overset{a}{\longrightarrow} p' \overset{\tau}{\longrightarrow} (p', \mathsf{MO}(p'))$ and $(p, \mathsf{MO}(p)) \overset{a}{\longrightarrow} p' \overset{\tau}{\longrightarrow} (p', \emptyset)$, whence $(p, \mathsf{MO}(p)) \overset{a}{\Longrightarrow\!|} (p', \mathsf{MO}(p'))$ and $(p, \mathsf{MO}(p)) \overset{a}{\Longrightarrow\!|} (p', \emptyset)$ since $p' \notin F$. Exploiting Cond. (RS3), there exist q', N' and q'', N'' such that $(q, N) \overset{a}{\Longrightarrow\!|} (q', N')$, $(q, N) \overset{a}{\Longrightarrow\!|} (q'', N'')$, $(p', \mathsf{MO}(p')) \underset{\sim}{\sqsubset}_{\mathrm{RS}} (q', N')$ and $(p', \emptyset) \underset{\sim}{\sqsubset}_{\mathrm{RS}} (q'', N'')$. By expansion again, $q \overset{a}{\dashrightarrow} q'$ and $q \overset{a}{\dashrightarrow} q''$, which means $q' = q''$ and $q \overset{a}{\Longrightarrow\!|} q'$ since Q is deterministic. It remains for us to establish $\mathcal{I}(\mathsf{MO}(p')) \subseteq \mathcal{I}(N')$ to be able to conclude $\langle p', q'\rangle \in \mathcal{R}$, as desired. Exploiting Cond. (RS4) and considering $p' \notin F$, we have $\mathcal{I}must(p') \cup \mathcal{I}(\mathsf{MO}(p')) = \mathcal{I}((p', \mathsf{MO}(p'))) = \mathcal{I}((q', N')) = \mathcal{I}must(q') \cup \mathcal{I}(N')$, as well as $\mathcal{I}must(p') = \mathcal{I}((p', \emptyset)) = \mathcal{I}((q', N'')) = \mathcal{I}must(q') \cup \mathcal{I}(N'')$. Hence, $\mathcal{I}must(p') \supseteq \mathcal{I}must(q')$ and, because $\mathcal{I}must(p') \cap \mathcal{I}(\mathsf{MO}(p')) = \emptyset$ and $\mathcal{I}must(q') \cap \mathcal{I}(N') = \emptyset$ by the definition of may-only transitions, $\mathcal{I}(\mathsf{MO}(p')) \subseteq \mathcal{I}(N')$.

(mRS4) $p \notin F$ implies $(p, \mathsf{MO}(p)) \notin F$, which in turn implies $\mathcal{I}((p, \mathsf{MO}(p))) = \mathcal{I}((q, N))$ by Cond. (RS4). To show $\mathcal{I}may(p) \subseteq \mathcal{I}may(q)$ we let $a \in \mathcal{I}may(p)$, i.e., $a \in \mathcal{I}((p, \mathsf{MO}(p)))$ by expansion. Hence, $a \in \mathcal{I}((q, N))$, which implies $a \in \mathcal{I}may(q)$. For establishing the inclusion $\mathcal{I}must(q) \subseteq \mathcal{I}must(p)$, let $a \in \mathcal{I}must(q)$ so that $a \in \mathcal{I}((q, N))$ by expansion. Therefore, $a \notin \mathcal{I}(N)$ and $a \in \mathcal{I}((p, \mathsf{MO}(p)))$. Since $\mathcal{I}(\mathsf{MO}(p)) \subseteq \mathcal{I}(N)$ and thus $a \notin \mathsf{MO}(p)$, this proves the existence of an a-must-transition of p, i.e., $a \in \mathcal{I}must(p)$.

Using this result we can now establish $p \sqsubseteq_{\mathrm{mRS}} q$. Since p, q are stable by assumption, it is sufficient to prove $\langle p, q\rangle \in \mathcal{R}$. To do so, consider $p \xrightarrow{\tau} (p, \mathsf{MO}(p))$ and $p \xrightarrow{\tau} (p, \emptyset)$ in the expansion of P. Thus, by Cond. (RS3) and expansion, $q \xrightarrow{\tau} (q, N)$, $q \xrightarrow{\tau} (q, N')$, $(p, \mathsf{MO}(p)) \sqsubset\!\!\!\sim_{\mathrm{RS}} (q, N)$ and $(p, \emptyset) \sqsubset\!\!\!\sim_{\mathrm{RS}} (q, N')$, for some $N, N' \subseteq \mathsf{MO}(q)$. By reasoning analogously as in (mRS3) above, we obtain $\mathcal{I}(\mathsf{MO}(p)) \subseteq \mathcal{I}(N)$, which finishes the proof. □

We end this section (Sec. 5.2) by proving that modal refinement implies modal ready simulation and thus, by Prop. 27, it also implies ready simulation. For this result we only consider modal transition systems without τ, as announced above. Thus, a standard modal transition system in the sense of Larsen [7] corresponds in our setting to a τ-free modal LLTS with $F = \emptyset$.

Definition 29 (Modal Refinement [7]). *A* modal refinement relation $\mathcal{R} \subseteq P \times Q$ *satisfies for all* $\langle p, q\rangle \in \mathcal{R}$ *and* $a \in \mathcal{A}$*:*

1. $p \overset{a}{\dashrightarrow} p'$ *implies* $\exists q'.\, q \overset{a}{\dashrightarrow} q'$ *and* $\langle p', q'\rangle \in \mathcal{R}$*;*
2. $q \xrightarrow{a} q'$ *implies* $\exists p'.\, p \xrightarrow{a} p'$ *and* $\langle p', q'\rangle \in \mathcal{R}$.

We write $p \leq_L q$ *if* $\langle p, q\rangle \in \mathcal{R}$ *for a modal refinement relation* $\mathcal{R}$*, and call* $\leq_L$ modal refinement.

It is easy to see that such a relation $\mathcal{R}$ is also a modal stable rs-relation: Cond. (mRS1) and Cond. (mRS2) hold trivially; Cond. (mRS3) reduces to Cond. (1) above; for Cond. (mRS4) we recall that $\mathcal{I}may(p) \subseteq \mathcal{I}may(q)$ by Cond. (mRS3), and $\mathcal{I}must(q) \subseteq \mathcal{I}must(p)$ by Cond. (2) above. Hence, we have the following proposition:

Proposition 30 (Modal Refinement Refines (Modal) Ready Simulation). $p \leq_L q$ *implies* $p \sqsubseteq_{mRS} q$ *and, hence,* $p \leq_L q$ *also implies* $p \sqsubseteq_{RS} q$.

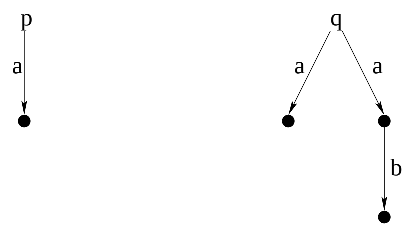

Figure 12: Counterexample: Ready simulation and modal ready simulation do not refine modal refinement.

It is not surprising that the reverse implication regarding this proposition fails in general, since modal refinement is of bisimulation-type. A counterexample is depicted in Fig. 12, for which $p \sqsubseteq_{\mathrm{mRS}} q$ and $p \sqsubseteq_{\mathrm{RS}} q$ hold, but not $p \leq_{\mathrm{L}} q$. However, the reverse implication is true for *deterministic* modal LLTS, as is not difficult to check.

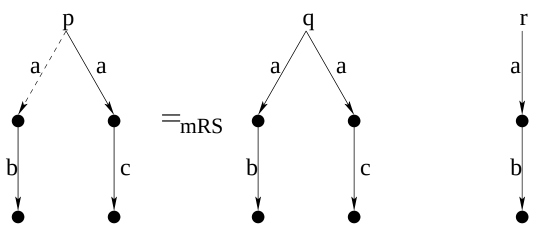

Figure 13: Example illustrating action-modality.

5.3. May & Must in Modal LLTS, Intuitively

We have set up modal LLTS in such a way that we can treat general modal transition systems as we have done above. However, this generality allows us to write down some modal LLTS whose meaning regarding *may* and *must* is not quite intuitive. The situation arises when a process possesses several a-transitions for some action a, at least one of which is a must-transition. Then, any a-may-transition of the process has a *must* character, as is illustrated by the example processes p, q and r in Fig. 13: somewhat surprisingly, r modal rs-refines p since it modal rs-refines q. Indeed, the specification p is more clearly expressed by q.

As suggested by this example, it is sufficient to focus our attention on a subset of modal LLTS where *may* and *must* do not depend on single transitions but only on each process and action. In other words, we can restrict ourselves to what we call *action-modal* LLTS:

Definition 31 (Action-Modal LLTS). *A modal LLTS P is an* action-modal LLTS *if, for all processes $p, p' \in P$ and $a \in \mathcal{I}must(p)$, we have $p \overset{a}{\dashrightarrow} p'$ implies $p \overset{a}{\longrightarrow} p'$.*

To the best of our knowledge, the subclass of action-modal LLTS within modal transition systems (i.e., τ-free LLTS with $F = \emptyset$) has not been considered in the literature before. The following theorem, whose proof again relies on our notion of modal ready simulation, shows that the restriction imposed by action-modal LLTS does not affect our setting's expressiveness:

Theorem 32 (Generality of Action-Modal LLTS). *For each modal LLTS P, there exists an action-modal LLTS $\overline{P}$ and a bijection $\overline{\cdot} : P \to \overline{P}$ such that, for all $p \in P$, $p =_{mRS} \overline{p}$ and, thus, $p =_{RS} \overline{p}$ wrt. $\hat{P}$ and $\hat{\overline{P}}$.*

Proof. Given a modal LLTS P, we construct the action-modal LLTS $\overline{P}$ as follows:

- $\overline{P} =_{\mathrm{df}} \{\overline{p} \mid p \in P\}$;
- $\longrightarrow_{\overline{P}} =_{\mathrm{df}} \{\overline{p} \overset{\alpha}{\longrightarrow} \overline{p}' \mid p \overset{\alpha}{\dashrightarrow} p'$ and $\alpha \in \mathcal{I}must(p)\}$ (containing $\longrightarrow_P$);
- $\dashrightarrow_{\overline{P}} =_{\mathrm{df}} \{\overline{p} \overset{\alpha}{\dashrightarrow} \overline{p}' \mid p \overset{\alpha}{\dashrightarrow} p'\}$;
- $\overline{F} =_{\mathrm{df}} \{\overline{p} \mid p \in F\}$.

Let $\mathcal{R} =_{\mathrm{df}} \{\langle p, \overline{p}\rangle \mid p \in P$ is stable$\}$. We show that $\mathcal{R}$ and its inverse $\mathcal{R}^{-1}$ are modal stable rs-relations:

(mRS1) & (mRS2) Both these conditions are straightforward to establish.

(mRS3) Since $\dashrightarrow_{\overline{P}}$ and $\dashrightarrow_P$ — as well as F and $\overline{F}$ — are essentially the same, we have $p \overset{\alpha}{\Longrightarrow}\!| \, p'$ if and only if $\overline{p} \overset{\alpha}{\Longrightarrow}\!| \, \overline{p}'$ for all $p, p' \in P$.

(mRS4) Analogously, $\mathcal{I}may(p) = \mathcal{I}may(\overline{p})$ for all $p \in P$. Furthermore, $\overline{p} \xrightarrow{\alpha}_{\overline{P}}$ only if $\alpha \in \mathcal{I}must(p)$, i.e., $\mathcal{I}must(\overline{p}) \subseteq \mathcal{I}must(p)$. Conversely, $\alpha \in \mathcal{I}must(p)$ implies $\exists p'.\, p \xrightarrow{\alpha}_{P} p'$; hence, $\overline{p} \xrightarrow{\alpha}_{\overline{P}} \overline{p}'$, i.e., $\alpha \in \mathcal{I}must(\overline{p})$.

Now the result follows since P and $\overline{P}$ are isomorphic on may-transitions and, in particular, on τ-may-transitions. □

5.4. Conjunction in Modal Transition Systems

A conjunction operator (and also a disjunction operator) has been defined for modal transition systems by Larsen in [7]. To be able to accommodate conjunction, Larsen generalised modal transition systems to deal with syntactic inconsistency, whereby must-transitions do not necessarily also have to be may-transitions. He then defined conjunction such that it gives the greatest lower bound with respect to $\leq_{\mathrm{L}}$ (with the same definition as above, see Def. 29), thus satisfying one of our benchmark results (cf. Prop. 5(2)). Of course, this works for the stricter $\leq_{\mathrm{L}}$ which is not justified by a full-abstractness result as $\sqsubseteq_{\mathrm{RS}}$ is [2] (cf. Thm. 6).

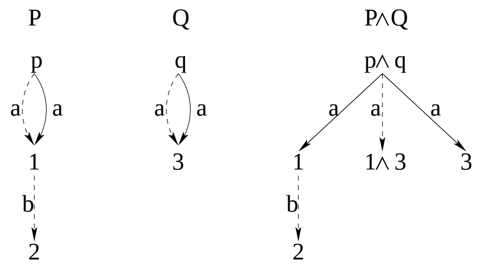

Figure 14: Example demonstrating the difficulty of understanding Larsen's conjunction operationally.

It must be mentioned that, although $\leq_{\mathrm{L}}$ has a very elegant definition, the result of Larsen's conjunction can be difficult to understand operationally since inconsistencies (i) are not directly related to unsatisfiability and (ii) are not "first-class citizens" as in our setting. To illustrate this we consider the example in Fig. 14, where ordinary arcs represent must-transitions without representing a may-transition. Here, $q \leq_{\mathrm{L}} p$ due to the modal refinement relation $\{\langle q, p\rangle, \langle 3, 1\rangle\}$, so we must have $q \leq_{\mathrm{L}} p \wedge q$. For obtaining this result, the must-transitions of $p \wedge q$ are matched by $q \xrightarrow{a} 3$, while the may-transition $q \overset{a}{\dashrightarrow} 3$ is matched by the separate $p \wedge q \overset{a}{\dashrightarrow} 1 \wedge 3$, which is surprising. Thus, although $p \wedge q$ "is syntactically inconsistent", it is refined by the consistent q. In line with the modal refinement developed in [19], q is even an implementation since $\dashrightarrow$ and $\longrightarrow$ coincide. Furthermore, it is not very intuitive from the graphical presentation of $p \wedge q$ that $q \leq_{\mathrm{L}} p \wedge q$ while $p \leq_{\mathrm{L}} p \wedge q$ fails. Finally, one cannot remove states or arcs to make $p \wedge q$ consistent without changing its meaning: we would have to remove both must-transitions, but then $p \wedge q \leq_{\mathrm{L}} q$ — which follows from $\wedge$ being a lower bound — would fail since there would be no match for $q \xrightarrow{a} 3$ any more.

This shortcoming has been avoided by Larsen et al. in [20] and by Raclet in [21]. Larsen et al. have limited conjunctive composition to so-called *independent* specifications which avoid inconsistencies, while Raclet has restricted his setting to *deterministic* modal transition systems. Raclet, but not Larsen et al., covers the example above, where $p \wedge q$ yields the desired result q. When expanding $p \wedge q$ to LLTS, the resulting LLTS is also exactly the expansion of q.

6. Related Work

Related work has often avoided mixing operational and logic styles of specification by translating one style into the other, although the use of combined styles has also been described by others, e.g., [22]. Logic content may be translated into operational content, such as in Kurshan's work on *ω-automata* [23] which includes synchronous and asynchronous composition operators and employs trace inclusion for refinement. However, trace inclusion is insensitive to deadlock and is thus inadequate in the presence of concurrency. Recent research on *interface theories* by Raclet et al. [24] mixes conjunction and synchronous product, which also considers some version of ready semantics in [21]. In contrast to our work, however, their line of research utilises *deterministic* modal transition systems, is also not sensitive to deadlock, and does not substantiate refinement via a full-abstraction result.

Dually, operational content may be translated into logic formulas, as is implicitly done by Lamport in [25] where logic implication serves as refinement relation [26]. A similar approach is followed in Hoare and He's UTP [27], the *Unifying Theories of Programming*, where a translation of the process algebra CSP [13] into logic formulas is indicated. Thus, conjunction is, e.g., applicable to processes a and $a + b$ (i.e., the p and r in Fig. 1(a)), which yields a process that can neither refuse b in the sense of failure semantics, nor can it perform b. Hence, $a \wedge (a + b)$ is an inconsistent process, but it is not treated as logically false as in our work. It seems that this inconsistency can be repaired in [27] by adding further choices (e.g., as in $(a \wedge (a+b)) + b = a+b$), which we regard as undesirable.

A seminal step towards a mixed setting was taken by Olderog in [28] where process-algebraic constructs are combined with *trace formulas*, and where failure semantics underlies refinement. In this approach, trace formulas can serve as processes, but not vice versa. Thus, and in contrast to our present work, [28] does not support the unrestricted mixing of operational and logic specification styles, which can be very useful as, e.g., demonstrated by our example in Sec. 4. In [29], a mixing of process-algebraic and temporal-logic operators is advocated by Guerra and Costa, too: a simple process algebra is extended with an operator to express that eventually some action occurs (see also [30]). Again, the semantics is based on traces and is thus not deadlock-sensitive. However, the ideas of Guerra and Costa may help one to extend our approach to liveness properties, as may those in [31].

In the context of a proof methodology based on modal transition systems, the process algebra CCS [32] has been extended by Larsen and others with may- and must-modalities and with a compositional conjunction operator [20]. While conjunction is – as mentioned above – only defined on *independent* processes, parallel composition and conjunction can be mixed more freely than in [28]; in particular, conjunction is shown to distribute over parallel composition. Larsen et al. also employ a typical pattern of modal transition systems within their proof methodology that corresponds to simple *AG* formulas in the temporal logic CTL [33]; however, an algebraic theory of mixing operational and (temporal-)logic operators is not considered in [20].

We also mention the work of Fecher and Grabe [34], where ready simulation is used as implementation relation and where a specific satisfaction for temporal-logic formulas is defined similar to our approach. In [34], whenever a process satisfies a formula, each implementation of the process satisfies the formula; however, [34] does not allow the free mixing of operators. Another consideration of logics in process algebra which does, however, not result in mixing logic and process-algebraic operators, involves *conditions* in if-then-else constructs; see, e.g., [35].

7. Conclusions & Future Work

This article embedded a temporal logic for specifying safety properties into the ready-simulation-equipped setting of Logic Labelled Transition Systems [2] (LLTS). The chosen logic was a branching-time logic that allows one to specify properties regarding the enabledness of actions, using standard temporal operators such as *always* and *unless* (*weak until*), which were shown to be compositional for ready simulation. The embedding is conservative in that ready simulation, when restricted to pairs of processes and temporal formulas, coincides with the logic's satisfaction relation. Moreover, ready simulation, when restricted to formulas, is entailment. The extended setting of LLTS is unique in the literature in that it lends itself to *freely* mixing operational and temporal-logic styles of specification, with ready simulation facilitating *compositional* refinement and model checking.

Regarding future work, we wish to re-phrase our setting in the classic process-algebraic style and to study axiomatisations of ready simulation. In addition, LLTS should be extended so as to be able to express liveness, too. This is, however, a non-trivial task as can be seen when considering the eventuality operator $\Diamond$ in temporal logics. We would define $p \models \Diamond\phi$ if, for all *maximal* runs $p \overset{\epsilon}{\Longrightarrow\!|} p_0 \overset{a_1}{\Longrightarrow\!|} p_1 \overset{a_2}{\Longrightarrow\!|} \cdots$ (where $a_i \in \mathcal{A}$) either ending in some p_n with $\forall \alpha \in \mathcal{A}_\tau.\ p_n \overset{\alpha}{\not\longrightarrow}$ or being infinite, there exists some process p_k with $p_k \models \phi$. Then, we would have to find a suitable LLTS for extending the compatibility theorem, Thm. 15 above.

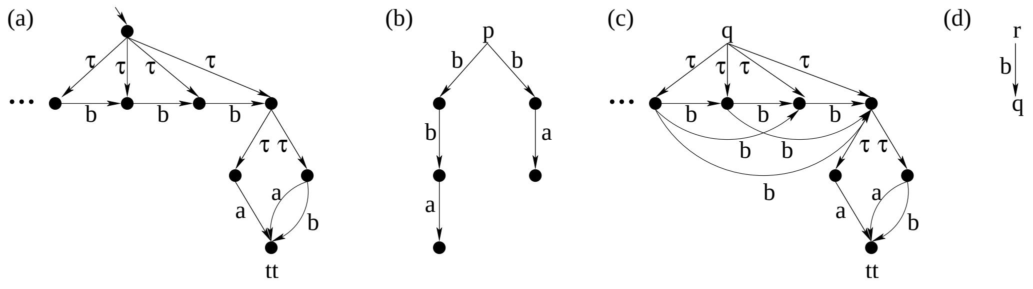

Figure 15: Examples demonstrating the difficulty of dealing with liveness in our setting.

We illustrate the problem of such an extension by taking $\mathcal{A} = \{a, b\}$ and considering the formula $\Diamond en(a)$. This example is particularly simple since we only have ready set $\{b\}$ before reaching a. In the spirit of LTL [33], we could understand $\Diamond en(a)$ as a disjunction over all b-sequences followed by an a, i.e., we would encode $\Diamond en(a)$ as the LLTS depicted in Fig. 15(a). But, for the process p shown in Fig. 15(b), this encoding cannot ready simulate p although $p \models \Diamond en(a)$, i.e., compatibility would be violated. The reason is that the encoding must initially choose a natural number k such that action a is enabled after *exactly* k actions b. To improve our encoding, we could add b-transitions in such a way that the decision when action a occurs can be postponed; see process q in Fig. 15(c). But, again, process r in Fig. 15(d), with q being the process in Fig. 15(c), satisfies $\Diamond en(a)$ while $r \not\sqsubseteq_{\mathrm{RS}} q$. Here, q must decide for a number k such that action a is enabled after *at most* k actions b (and at least one action b), while r can postpone this decision. Therefore, it seems that we must enrich our LLTSs with a *Büchi-type acceptance condition* to deal with liveness. However, it is not clear to us how to handle Büchi states in a simulation setting satisfactorily, e.g., so that a full-abstraction result (cf. Thm. 6) can be obtained.

References

[1] G. Lüttgen, W. Vogler, Conjunction on processes: Full-abstraction via ready-tree semantics, Theoret. Comp. Sc. 373 (2007) 19–40.

[2] G. Lüttgen, W. Vogler, Ready simulation for concurrency: It's logical!, Inform. and Comp. 208 (2010) 845–867.

[3] J. Bergstra, A. Ponse, S. Smolka, Handbook of Process Algebra, Elsevier, 2001.

[4] B. Bloom, Ready Simulation, Bisimulation, and the Semantics of CCS-like Languages, Ph.D. thesis, MIT, 1990.

[5] R. Glabbeek, The linear time – branching time spectrum II, 1993. Available as an electronic download from http://theory.stanford.edu/˜rvg/abstracts.html#26.

[6] I. Ulidowski, Refusal simulation and interactive games, in: AMAST 2002, volume 2422 of *LNCS*, Springer, 2002, pp. 208–222.

[7] K. Larsen, Modal specifications, in: Automatic Verification Methods for Finite State Systems, volume 407 of *LNCS*, Springer, 1989, pp. 232–246.

[8] J. Warmer, A. Kleppe, The Object Constraint Language: Precise Modeling With UML, Addison Wesley, 1999.

[9] D. Harel, Statecharts: A visual formalism for complex systems, Sc. Comp. Prog. 8 (1987) 231–274.

[10] B. Douglass, Real-Time UML: Developing Efficient Objects for Embedded Systems, Addison Wesley, 1998.

[11] S. Brookes, C. Hoare, A. Roscoe, A theory of communicating sequential processes, J. ACM 31 (1984) 560–599.

[12] R. De Nicola, F. Vaandrager, Action versus state based logics for transition systems, in: Semantics of Systems of Concurrent Processes, volume 469 of *LNCS*, Springer, 1990, pp. 407–419.

[13] C. Hoare, Communicating Sequential Processes, Prentice Hall, 1985.

[14] K. Larsen, L. Xinxin, Equation solving using modal transition systems, in: LICS '90, IEEE Computer Society Press, 1990, pp. 108–117.

[15] L. De Alfaro, T. Henzinger, Interface automata, in: FSE 2001, ACM Press, 2001, pp. 109–120.

[16] L. De Alfaro, T. Henzinger, Interface-based design, in: M. Broy, J. Grünbauer, D. Harel, C. Hoare (Eds.), Engineering Theories of Software-intensive Systems, volume 195 of *NATO Science Series: Mathematics, Physics, and Chemistry*, Springer, 2005, pp. 83–104.

[17] K. Larsen, U. Nyman, A. Wasowski, Modal I/O automata for interface and product line theories, in: ESOP 2007, volume 4421 of *LNCS*, Springer, 2007, pp. 64–79.

[18] N. Beneš, J. Křetínský, K. Larsen, J. Srba, On determinism in modal transition systems, Theoret. Comp. Sc. 410 (2009) 4026–4043.

[19] K. Larsen, U. Nyman, A. Wasowski, On modal refinement and consistency, in: CONCUR 2007, volume 4703 of *LNCS*, Springer, 2007, pp. 105–119.

[20] K. Larsen, B. Steffen, C. Weise, A constraint oriented proof methodology based on modal transition systems, in: TACAS '95, volume 1019 of *LNCS*, Springer, 1995, pp. 17–40.

[21] J.-B. Raclet, Residual for component specifications, in: Formal Aspects of Component Software, volume 215 of *ENTCS*, Elsevier, 2008.

[22] E. Brinksma, Constraint-oriented specification in a constructive formal description technique, in: REX '89, volume 430 of *LNCS*, Springer, 1990, pp. 130–152.

[23] R. Kurshan, Computer-Aided Verification of Coordinating Processes: The Automata-Theoretic Approach, Princeton Univ. Press, 1994.

[24] J.-B. Raclet, E. Badouel, A. Benveniste, B. Caillaud, A. Legay, R. Passerone, A modal interface theory for component-based design, Fundamenta Informaticae 107 (2011) 1–32.

[25] L. Lamport, The temporal logic of actions, ACM TOPLAS 16 (1994) 872–923.

[26] M. Abadi, G. Plotkin, A logical view of composition, Theoret. Comp. Sc. 114 (1993) 3–30.

[27] C. Hoare, J. He, Unifying Theories of Programming, Prentice Hall, 1998.

[28] E.-R. Olderog, Nets, Terms and Formulas, Cambridge Tracts in Theoretical Computer Science 23, Cambridge Univ. Press, 1991.

[29] H. Guerra, J. Costa, Processes with local and global liveness requirements, J. Logic and Algebraic Programming 78 (2008) 117–137.

[30] A. Puhakka, A. Valmari, Liveness and fairness in process-algebraic verification, in: CONCUR 2001, volume 2154 of *LNCS*, Springer, 2001, pp. 202–217.

[31] T. Henzinger, R. Majumdar, Fair bisimulation, in: TACAS 2000, volume 1785 of *LNCS*, Springer, 2000, pp. 299–314.

[32] R. Milner, Communication and Concurrency, Prentice Hall, 1989.

[33] E. Emerson, Temporal and modal logic, in: Handbook of Theoretical Computer Science, volume B, North-Holland, 1990, pp. 995–1072.

[34] H. Fecher, I. Grabe, Finite abstract models for deterministic transition systems, in: FSEN 2007, volume 4767 of *LNCS*, Springer, 2007, pp. 1–16.

[35] J. Bergstra, A. Ponse, Process algebra and conditional composition, Inform. Process. Lett. 80 (2001) 41–49.

Appendix A. Additional Proofs

For the sake of completeness, this section contains the proofs of the lemmas stated in the main body of the article.

Appendix A.1. Proof of Lemma 14

Proof. We need to check Conds. (W1)–(W4) of $\Box$-witness.

(W1) If $\vec{q} \in W_1$, then $\vec{p} \notin F_{\Box P}$, which implies $p_i \notin F_P$ for all i. Hence, $q_i \notin F_Q$ by $p_i \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} q_i$, for all i. If $\vec{q} \in W_2$, then $q_i \overset{\epsilon}{\Longrightarrow}\!\!|$, for all i, and thus $q_i \notin F_Q$.

(W2) If $\vec{q} \in W_1$ stable, then q_i and q_j are stable for any i, j and, by the above, $q_i, q_j \notin F_Q$. By $p_i \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} q_i$ and $p_j \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} q_j$, we obtain $\mathcal{I}(q_i) = \mathcal{I}(p_i) = \mathcal{I}(p_j) = \mathcal{I}(q_j)$, where the second equality holds due to $\vec{p} \notin F_{\Box P}$.

If $\vec{q} \in W_2$ stable, then $\vec{q} \in W_1$ and we are in the case above.

(W3) We first consider the case $\alpha = \tau$. Then, $\vec{q} \xrightarrow{\tau}$ implies $\exists i.\, q_i \xrightarrow{\tau} \overline{q}_i$ for some $\overline{q}_i$. Moreover, $\vec{q}$ can only be in W_2 and not in W_1 since W_1 requires $\vec{q}$ to be stable. Thus, w.l.o.g., $\overline{q}_i$ is chosen such that $\overline{q}_i \overset{\epsilon}{\Longrightarrow}\!\!|$. By definition of W_2, we have $\vec{q} \xrightarrow{\tau} (q_1, \dots, \overline{q}_i, \dots, q_n) \in W_2$.

If $\alpha \neq \tau$, then $\vec{q} \xrightarrow{\alpha}$ means $\vec{q} \in W_1$. Moreover, $q_i \xrightarrow{\alpha}$ for all i. Thus, due to $\vec{p} \notin F_{\Box P}$ and $p_i \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} q_i$, we have $\forall i.\, p_i \xrightarrow{\alpha}$ by Cond. (RS4). Thus, $\vec{p} \xrightarrow{\alpha}$, and hence $\exists \vec{p}'.\, \vec{p} \overset{\alpha}{\Longrightarrow}\!\!| (\vec{p}', p)$ and $\forall i.\, p_i \overset{\alpha}{\Longrightarrow}\!\!| p_i'$. By Cond. (RS3), there exist q_i' and $\hat{q}_i$ such that $q_i \xrightarrow{\alpha}_{\mathrm{F}} \hat{q}_i \overset{\epsilon}{\Longrightarrow}\!\!| q_i'$ and $p_i' \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} q_i'$. Moreover, we know $p \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} q$ and $p \notin F_P$, so that $(q_1', \dots, q_n', q) \in W_1$. Now, $\vec{q} \xrightarrow{\alpha} (\hat{q}_1, \dots, \hat{q}_n, q) \in W_2$.

(W4) If $\vec{q} \in W_1$, then q_i is stable for all i, which implies that $\vec{q}$ is stable, too. Therefore, $\vec{q}$ can stabilise trivially in W.

If $\vec{q} \in W_2$, then $\vec{q}$ can stabilise since every q_i can stabilise by the definition of W_2. This stabilisation is in W_2 by construction. $\Box$

Appendix A.2. Proof of Lemma 16

Proof. We first show the lemma for $\mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}}$ in place of $\sqsubseteq_{\mathrm{RS}}$, before concluding by establishing the root condition. In order to prove $p \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} q_i$ from $p \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} \vec{q}$ for all $p \in P$ and $\vec{q} \in \Box\Psi$, it is sufficient to establish that

$$\mathcal{R} =_{\mathrm{df}} \{\langle p, q_i \rangle \mid \exists n, q_1, \dots, q_{i-1}, q_{i+1}, \dots, q_n.\, p \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} \vec{q}\}$$

is a stable rs-relation. We verify Conds. (RS1)–(RS4) of Def. 4:

(RS1) Process p is stable, and all q_i are stable since $\vec{q}$ is stable.

(RS2) If $p \notin F$, then $\vec{q} \notin F$ since $p \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} \vec{q}$. Hence, $q_i \notin F$.

(RS3) Let $p \overset{a}{\Longrightarrow}\!\!| p'$. By $p \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} \vec{q}$, there exists some $\vec{q}' = (q_1', \dots, q_{n+1}')$ such that $\vec{q} \overset{a}{\Longrightarrow}\!\!| \vec{q}'$ and $p' \mathrel{\underset{\sim}{\sqsubset}}_{\mathrm{RS}} \vec{q}'$. Therefore, $q_i \overset{a}{\Longrightarrow}\!\!| q_i'$ and $\langle p', q_i' \rangle \in \mathcal{R}$.

(RS4) Let $p \notin F$. Then, $\mathcal{I}(p) = \mathcal{I}(\vec{q})$ due to $p \lesssim_{\mathrm{RS}} \vec{q}$. By construction, $\mathcal{I}(\vec{q}) = \mathcal{I}(q_1) = \ldots = \mathcal{I}(q_n)$ since $\vec{q} \notin F$ by the above. Hence, $\mathcal{I}(p) = \mathcal{I}(q_i)$.

We can now complete the proof of the lemma by establishing the root condition. Let $p \overset{\epsilon}{\Longrightarrow\!\!|}\, p'$ for some p'. Hence, by $p \sqsubseteq_{\mathrm{RS}} \vec{q}$, there exists some $\vec{q}' = (q'_1, \ldots, q'_m)$ such that $\vec{q} \overset{\epsilon}{\Longrightarrow\!\!|}\, \vec{q}'$ and $p' \lesssim_{\mathrm{RS}} \vec{q}'$. This implies $q_i \overset{\epsilon}{\Longrightarrow\!\!|}\, q'_i$ and, by the above, $p' \lesssim_{\mathrm{RS}} q'_i$. □

Appendix A.3. Proof of Lemma 22

Proof. For proving Part (1), we have that each $p'' \neq p'$ on the run underlying $p \overset{\epsilon}{\Longrightarrow\!\!|}\, p'$ is instable in $\hat{P}$. Inserting $p'' \xrightarrow{\tau} (p'', \emptyset)$ in each case (and $p' \xrightarrow{\tau} (p', M')$) we get a run in $\hat{P}$, proving that $p \overset{\epsilon}{\Longrightarrow\!\!|}\, (p', M')$. Note that $p'' \notin F$ implies $p'' \notin \hat{F}$ and $(p'', \emptyset) \notin \hat{F}$, and similarly for $p' \notin F$.

For proving Part (2), observe that the run underlying $p \overset{\epsilon}{\Longrightarrow\!\!|}\, (p', M')$ consists of pairs of transitions $p'' \xrightarrow{\tau} (p'', \emptyset) \xrightarrow{\tau} p'''$ and the last step $p' \xrightarrow{\tau} (p', M')$. Replacing each pair by $p'' \xrightarrow{\tau} p'''$ and omitting the last step, we get a run proving $p \overset{\epsilon}{\Longrightarrow\!\!|}\, p'$ in P. Note that, for each p'', we have $p'' \notin \hat{F}$ and thus $p'' \notin F$. □