Revisiting Timed Specification Theories: A Linear-Time Perspective

Chris Chilton Marta Kwiatkowska Xu Wang
Department of Computer Science, University of Oxford, UK

July 28, 2012

Abstract

We consider the setting of component-based design for real-time systems with critical timing constraints. Based on our earlier work, we propose a compositional specification theory for timed automata with I/O distinction, which supports substitutive refinement. Our theory provides the operations of parallel composition for composing components at run-time, logical conjunction/disjunction for independent development, and quotient for incremental synthesis. The key novelty of our timed theory lies in a weakest congruence preserving safety as well as bounded liveness properties. We show that the congruence can be characterised by two linear-time semantics, *timed-traces* and *timed-strategies*, the latter of which is derived from a game-based interpretation of timed interaction.

1 Introduction

Component-based design methodologies can be encapsulated in the form of compositional specification theories, which allow the mixing of specifications and implementations, admit substitutive refinement to facilitate reuse, and provide a rich collection of operators. Several such theories have been introduced in the literature, but none simultaneously address the following requirements: support for the asynchronous communication model, as opposed to handshake communication; linear-time refinement preorder, so as to interface with automata and learning techniques; substitutivity of refinement, to allow for component reuse at runtime without introducing errors; and strong algebraic and compositionality properties, to enable reasoning at runtime as well as offline. Previously [1], we developed a linear-time specification theory for reasoning about untimed components that interact by synchronisation of input and output (I/O) actions. Models can be specified operationally by means of transition systems augmented by an inconsistency predicate on states, or declaratively using traces. The theory admits non-determinism, a substitutive refinement preorder based on traces, and the operations of parallel composition, conjunction and quotient. The refinement is strictly weaker than alternating simulation and is actually the weakest pre-congruence preserving inconsistent states.

In this paper we target component-based development for real-time systems with critical timing constraints, such as embedded systems components, the middleware layer and asynchronous hardware. Amongst notable works in the literature, we surveyed the theory of timed interfaces [2] and the theory of timed specifications [3]. Though both support I/O distinctions, their refinement relations are not linear time: in [2] refinement (compatibility) is based on timed games, and in [3] it is a timed version of the alternating simulation originally defined for interface automata [4]. Consequently, it is too strong for determining when a component can be safely substituted for another. As an example, consider the transition systems P and Q in Figure 3: these should be equivalent in the sense of substitutivity under any environment, and are equivalent in our formulation (Definition 5), but they are not so according to timed alternating simulation.

Contributions. We formulate an elegant timed, asynchronous specification theory based on finite traces which supports substitutive refinement, as a timed extension of the linear-time specification theory of [1]. We allow for both operational descriptions of components, as well as declarative specifications based on traces. Our operational models are a variant of timed automata with I/O distinction (although we do not insist on input-enabledness, cf [5]), augmented by two special states: $\perp$ is the *inconsistent* state, used to represent safety and bounded-liveness errors, and $\top$ is the *timestop* state, a novel addition representing either unrealisable output (if the component is not willing to produce that output) or unrealisable time-delay (if the delay would violate the invariant on that state).

Timestop models the ability to stop the clock and has been used before [6, 7] in embedded system and circuit design. It is notationally convenient, accounting for simpler definitions and a cleaner formalism. By also enhancing the automata with the notion of *co-invariant*, we can, for the first time, distinguish the roles of input/output guards and invariant/co-invariants as specifying, respectively, safety and bounded-liveness timed assumptions/guarantees. We emphasise that this is achieved with finite traces only; note that in the untimed case it would be necessary to extend to infinite traces to model liveness. In addition to *timed-trace semantics*, we present *timed-strategy semantics*, which coincides with the former but relates our work closer to the timed-game frameworks used by [3] and [2], and could in future serve as a guide to implementation of the theory. Finally, the *substitutive refinement* of our framework gives rise to the weakest congruence preserving $\perp$, which is not the case in the formalism of [3].

Our work could be seen as an alternative to the timed theories of [2, 3]. Being linear-time in spirit, it is also a generalisation of [8], an untimed theory inspired by asynchronous circuits, and Dill's trace theory [9]. For more detailed comparison with related works see Section 5.

Outline. In Section 2 we introduce timed I/O automata, their semantic mapping to timed I/O transition systems, and supply the operational definitions for the operations of parallel composition, conjunction, disjunction and quotient. In Section 3 we use the timed-game framework to introduce timed-strategy semantics, which we relate to the operational framework. Similarly in Section 4, we present timed-trace semantics and relate these to the operational definitions. Section 5 discusses related work, and finally Section 6 concludes.

2 Formal Framework

In this section we introduce timed I/O automata, timed I/O transition systems and a semantic mapping from the former to the latter. Timed I/O automata are compact representations of timed I/O transition systems. Our theory will be developed using timed I/O transition systems, which are endowed with a richer repertoire of semantic machinery.

2.1 Timed I/O Automata

Clock constraints. Given a set X of real-valued clock variables, a *clock constraint* over X, $cc : CC(X)$, is a boolean combination of atomic constraints of the form $x \bowtie d$ and $x - y \bowtie d$ where $x, y \in X$, $\bowtie \in \{\leq, <, =, >, \geq\}$, and $d \in \mathbb{N}$.

A *clock valuation* over X is a map t that assigns to each clock variable x in X a real value from $\mathbb{R}^{\geq 0}$. We say t satisfies cc, written $t \in cc$, if cc evaluates to true under valuation t. $t + d$ denotes the valuation derived from t by increasing the assigned value on each clock variable by $d \in \mathbb{R}^{\geq 0}$ time units. $t[rs \mapsto 0]$ denotes the valuation obtained from t by resetting the clock variables in rs to 0. Sometimes we use 0 for the clock valuation that maps all clock variables to 0.

Definition 1. *A* timed I/O automaton *(TIOA) is a tuple* $(C, I, O, L, l^0, AT, \mathit{Inv}, \mathit{coInv})$*, where:*

- $C \subseteq X$ *is a finite set of clock variables*
- $A\ (= I \cup O)$ *is a finite alphabet, where* I *and* O *are disjoint sets of input actions and output actions respectively*
- L *is a finite set of* locations
- $l^0 \in L$ *is the* initial location
- $AT \subseteq L \times CC(C) \times A \times 2^C \times L$ *is a set of* action transitions
- $\mathit{Inv} : L \to CC(C)$ *and* $\mathit{coInv} : L \to CC(C)$ *assign* invariants *and* co-invariants *to states, each of which is a downward-closed clock constraint.*

We use l, l', l_i to range over L and use $l \xrightarrow{g,a,rs} l'$ as a shorthand for $(l, g, a, rs, l') \in AT$. $g : CC(C)$ is the enabling guard of the transition, $a \in A$ the action, and rs the subset of clock variables to be reset.

Our TIOAs are timed automata which distinguish input from output and invariant from co-invariant. They are similar to existing variants of timed automata with input/output distinction, except for the introduction of co-invariants and non-insistence on input-enabledness. While invariants specify the bounds beyond which time may not progress, co-invariants specify the bounds beyond which the system will *time-out* and enter error states. It is designed for the assume/guarantee specification of timed components, i.e. specifying both the assumptions made by the component on the inputs and the guarantees provided by the component on the outputs.

Such assumptions and guarantees can be time constrained. Guards on output transitions express *safety timing guarantees* while guards on input transitions express *safety timing assumptions*. On the other hand, invariants (urgency) express *liveness timing guarantees* on the outputs at the locations they decorate while co-invariants (time-out) express *liveness timing assumptions* on the inputs at those locations.

When two components are composed, the parallel composition automatically checks whether the guarantees provided by one component meet the assumptions required by the other. For instance, the arrival of an input at a location and time of a component when it is not expected (i.e. the input is disabled at the location and time) leads to a *safety error* in the parallel composition. Or the non-arrival of an expected input at a location before its time-out (specified by the co-invariant) leads to a *bounded-liveness error* in the parallel composition.

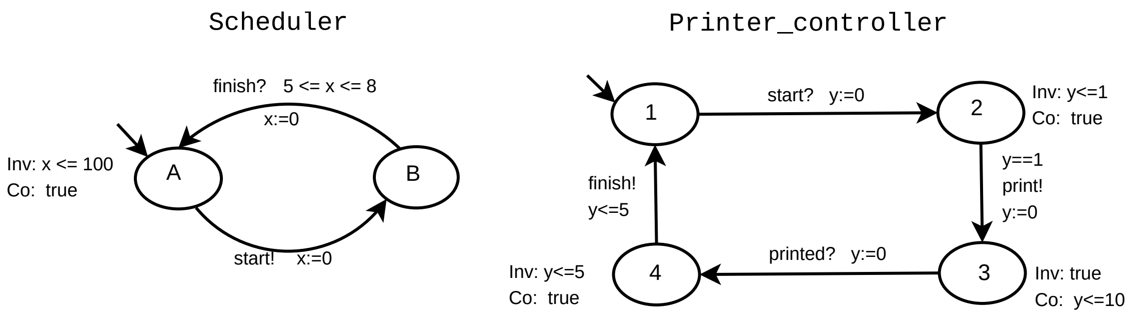

Figure 1: Job scheduler and printer controller.

Example. Figure 1 depicts TIOAs representing a job scheduler together with a printer controller. The invariant at location A of the scheduler forces a bounded-liveness guarantee on outputs in that location. As time must be allowed to progress beyond $t = 100$, the *start* action must be fired within the range $0 \leq t \leq 100$. After *start* has been fired, the clock x is reset to 0 and the scheduler waits (possibly indefinitely) for the job to *finish*. If the job does finish, the scheduler is only willing for this to take place between $5 \leq t \leq 8$ after the job started (safety assumption), otherwise an unexpected input error will be thrown.

The controller waits for the job to *start*, after which it will wait exactly 1 time unit before issuing *print* (forced by the invariant $y \leq 1$ on state 2 and the guard $y = 1$). The controller now requires the printer to indicate the job is *printed* within 10 time units of being sent to the printer, otherwise a time-out error on inputs will occur (co-invariant $y \leq 10$ in state 3 as liveness assumption). After the job has finished printing, the controller must indicate to the scheduler that the job has *finish*ed within 5 time units.

2.2 Timed Actions and Words

In this section we introduce some notation relating to timed actions and timed words that will be of use to us in later sections.

Timed actions. For a set of input actions I and a set of output actions O, define $tA = I \uplus O \uplus \mathbb{R}^{>0}$ to be the set of *timed actions*, $tI = I \uplus \mathbb{R}^{>0}$ to be the set of *timed inputs*, and $tO = O \uplus \mathbb{R}^{>0}$ to be the set of *timed outputs*. We use symbols like α, β, etc. to range over tA.

Timed words. A *timed word* (ranged over by w, w', w_i etc.) is a finite mixed sequence of positive real numbers ($\mathbb{R}^{>0}$) and visible actions such that *no two numbers are adjacent to one another*. For instance, $\langle 0.33, a, 1.41, b, c, 3.1415\rangle$ is a timed word denoting the observation that action a occurs at 0.33 time units, then another 1.41 time units lapse before the simultaneous occurrence of b and c, which is followed by 3.1415 time units of no event occurrence. The empty word is denoted by ϵ.

Operations on timed words. Concatenation of timed words w and w' is obtained by appending w' onto the end of w and coalescing adjacent reals (summing them). For instance, $\langle a, 1.41\rangle ^\frown \langle 0.33, b, 3.1415\rangle = \langle a, (1.41 + 0.33), b, 3.1415\rangle = \langle a, 1.74, b, 3.1415\rangle$. Prefix/extension are defined as usual by concatenation, and we use $\leq$ for the prefix partial order. We write $w \restriction tA_0$ for the projection of w onto timed alphabet tA_0, which is defined by removing from w all actions not inside tA_0 and coalescing adjacent reals.

2.3 Semantics as Timed I/O Transition Systems

The semantics of TIOAs are given as timed I/O transition systems, which are a special class of infinite labelled transition systems.

Definition 2. *A* timed I/O transition system *(TIOTS) is a tuple $\mathcal{P} = \langle I, O, S, s^0, \rightarrow\rangle$, where I and O are the input and output actions respectively, $S = (L \times \mathbb{R}^C) \uplus \{\bot, \top\}$ is a set of states, $s^0 \in S$ is the designated initial state, and $\rightarrow \subseteq S \times I \uplus O \uplus \mathbb{R}^{>0} \times S$ is the action and time-labelled transition relation.*

The states of the TIOTS (for a TIOA) capture the configurations of the TIOA, i.e. its location and clock valuation. Therefore, each state of the TIOTS is a pair drawn from $L \times \mathbb{R}^C$, which we refer to as the set of *plain states*. In addition, we introduce two special states $\bot$ and $\top$, which are required for the semantic mapping of disabled inputs/outputs, invariants and co-invariants. In the rest of the paper we use p, p', p_i to range over $P = L \times \mathbb{R}^C$ while s, s', s_i range over S.

$\bot$ is the so-called *inconsistent state*, representing the erroneous state generated by the assumption/guarantee mismatch (i.e. both safety and bounded-liveness errors). $\top$ is the so-called *timestop state*, representing the *magic moment* from which time stops elapsing (and thus removes the subsequent possibility of errors). We assume that $\top$ refines plain states, which in turn refine $\bot$.

On TIOTSs, a disabled input in a state p is equated to an input transition from p to $\bot$, while a disabled output/delay in p is equated to an output/delay from p to $\top$. The intuition here comes from the input/output game perspective. The component controls output and delay while the environment controls input. $\bot$ is the losing state for the environment. So an input transition from p to $\bot$ is a transition that the environment controls and tries to avoid at all cost (unless there is no choice at all). $\top$ is the losing state for the component. So an output/delay transition from p to $\top$ is a transition that the component controls and tries to avoid at all cost. Thus we can have two semantics-preserving transformations on TIOTSs.

The $\bot$-*completion* of a TIOTS $\mathcal{P}$, denoted $\mathcal{P}^{\bot}$, adds an a-labelled transition from p to $\bot$ for every $p \in P$ $(= L \times \mathbb{R}^C)$ and $a \in I$ s.t. a is not enabled at p.[1] The $\top$-*completion*, denoted $\mathcal{P}^{\top}$, adds an α-labelled transition from p to $\top$ for every $p \in P$ and $\alpha \in tO$ s.t. α is not enabled at p.

Furthermore, for technical convenience (e.g. ease of defining time additivity and trace semantics), the definition of TIOTSs requires that $\top$ and $\bot$ are a *chaotic state*, i.e. a state in which the set of outgoing transitions are all self-loops, one for each $\alpha \in tA$.

The transition relation $\rightarrow$ of the TIOTS is derived from the execution semantics of the TIOA.

Definition 3. *Let $\mathcal{P}$ be a TIOA. The execution semantics of $\mathcal{P}$ is a TIOTS $\langle I, O, S, s^0, \rightarrow\rangle$, where:*

- $S = (L \times \mathbb{R}^C) \uplus \{\bot, \top\}$
- $s^0 = \top$ *providing* $0 \notin Inv(l^0)$, $s^0 = \bot$ *providing* $0 \in Inv(l^0) \wedge \neg coInv(l^0)$ *and* $s^0 = (l^0, 0)$ *providing* $0 \in Inv(l^0) \wedge coInv(l^0)$,
- $\rightarrow$ *is the smallest relation satisfying:*
  1. *If* $l \xrightarrow{g,a,rs} l'$, $t' = t[rs \mapsto 0]$, $t \in Inv(l) \wedge coInv(l) \wedge g$, *then:*
     (a) plain action: $(l, t) \xrightarrow{a} (l', t')$ *providing* $t' \in Inv(l') \wedge coInv(l')$

[1] $\bot$-completion will make a TIOTS *input-receptive*, i.e. input-enabled at all states.

(b) error action: $(l,t) \xrightarrow{a} \bot$ *providing* $t' \in Inv(l') \wedge \neg coInv(l')$

(c) magic action: $(l,t) \xrightarrow{a} \top$ *providing* $t' \in \neg Inv(l')$ *and* $a \in I$.

2. plain delay: $(l,t) \xrightarrow{d} (l, t+d)$ *if* $t, t+d \in Inv(l) \wedge coInv(l)$

3. time-out delay: $(l,t) \xrightarrow{d} \bot$ *if* $t \in Inv(l) \wedge coInv(l)$, $t+d \notin coInv(l)$ *and* $\exists 0 < \delta \leq d : t+\delta \in Inv(l) \wedge \neg coInv(l)$.

Note that our semantics tries to minimise the use of transitions leading to $\top/\bot$ states. Thus there are no delay or output transitions leading to $\top$. However, there are *implicit timestops*, which we capture using the concept of *semi-timestop* (i.e. semi-$\top$). We say a plain state p is a *semi-*$\top$ iff 1) all output transitions enabled in p or any of its time-passing successors lead to the $\top$ state, and 2) there exists $d \in \mathbb{R}^{>0}$ s.t. $p \xrightarrow{d} \top$ or d is not enabled in p. Thus a semi-$\top$ is a state in which it is impossible for the component to avoid the timestop without suitable inputs from the environment.

TIOTS terminology. We say a TIOTS is *deterministic* iff $s \xrightarrow{\alpha} s' \wedge s \xrightarrow{\alpha} s''$ implies $s' = s''$, and is *time additive* providing $p \xrightarrow{d_1+d_2} s'$ iff $p \xrightarrow{d_1} s$ and $s \xrightarrow{d_2} s'$ for some s. In the sequel, we only consider time-additive TIOTSs.

Given a TIOTS $\mathcal{P}$, a timed word can be derived from a finite execution of $\mathcal{P}$ by extracting the labels in each transition and coalescing adjacent reals. The timed words derived from such executions are called *traces* of $\mathcal{P}$. We use tt, tt', tt_i to range over the set of traces and write $s^0 \stackrel{tt}{\Rightarrow} s$ to denote a finite execution that produces trace tt and leads to s.

Remark on timestop. An unconventional aspect of our semantics is the introduction of timestop ($\top$). Timestop (and semi-timestop) can be introduced explicitly into a specification to model the operation of stopping the system clock. It is well known that parallel composition of components will not introduce new timestop (or semi-timestop).

Certain real-world systems have an inherent ability to stop the clock, e.g. [6, 7], related to embedded systems and circuit design. For systems where the suspension of clocks is not meaningful, a theory that can remove timestops to keep only the so-called realisable behaviours will be developed as future work. Note that even for timestop-free systems, timestop can play an important role as an imaginary state exploited at the intermediate steps of theory development in order to significantly simplify operations (e.g. quotient and conjunction).

2.4 Operational Specification Theory

In this section we develop a compositional specification theory for TIOTSs based on the operations of parallel composition $\|$, conjunction $\wedge$, disjunction $\vee$ and quotient %. The operators are defined via transition rules that are a variant on synchronised product.

Parallel composition yields a TIOTS that represents the combined effect of its operands interacting with one another. The remaining operations must be explained with respect to a refinement relation, which corresponds to safe-substitutivity in our theory. A TIOTS is a refinement of another if it will work in any environment that the original worked in without introducing safety or bounded-liveness errors. Conjunction yields the coarsest TIOTS that is a refinement of its operands, while disjunction yields the finest TIOTS that is refined by both of its operands. The

Table 1: State representations under composition operators.

$\|$	$\top$	p_0	$\bot$	$\wedge$	$\top$	p_0	$\bot$	$\vee$	$\top$	p_0	$\bot$	$\%$	$\top$	p_0	$\bot$
$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	$\top$	p_0	$\bot$	$\top$	$\bot$	$\bot$	$\bot$
p_1	$\top$	$p_0{\times}p_1$	$\bot$	p_1	$\top$	$p_0{\times}p_1$	p_1	p_1	p_1	$p_0{\times}p_1$	$\bot$	p_1	$\top$	$p_0{\times}p_1$	$\bot$
$\bot$	$\top$	$\bot$	$\bot$	$\bot$	$\top$	p_0	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$	$\bot$	$\top$	$\top$	$\bot$

operators are thus equivalent to the join and meet operations on TIOTSs[2]. Quotient is the adjoint of parallel composition, meaning that $\mathcal{P}_0\%\mathcal{P}_1$ is the coarsest TIOTS such that $(\mathcal{P}_0\%\mathcal{P}_1)\|\mathcal{P}_1$ is a refinement of $\mathcal{P}_0$.

Let $\mathcal{P}_i = \langle I_i, O_i, S_i, s_i^0, \rightarrow_i\rangle$ for $i \in \{0,1\}$ be two TIOTSs that are both $\bot$ and $\top$-completed, satisfying (wlog) $S_0 \cap S_1 = \{\bot, \top\}$. The composition of $\mathcal{P}_0$ and $\mathcal{P}_1$ under the operation $\otimes \in \{\|, \wedge, \vee, \%\}$, written $\mathcal{P}_0 \otimes \mathcal{P}_1$, is only defined when certain *composability* restrictions are imposed on the alphabets of the TIOTSs. $\mathcal{P}_0 \parallel \mathcal{P}_1$ is only defined when the output sets of $\mathcal{P}_0$ and $\mathcal{P}_1$ are disjoint, because an output should be controlled by at most one component. Conjunction and disjunction are only defined when the TIOTSs have *identical alphabets* (i.e. $O_0 = O_1$ and $I_0 = I_1$). This restriction can be relaxed at the expense of more cumbersome notation, which is why we focus on the simpler case in this paper. For the quotient, we require that the alphabet of $\mathcal{P}_0$ *dominates* that of $\mathcal{P}_1$ (i.e. $A_1 \subseteq A_0$ and $O_1 \subseteq O_0$), in addition to $\mathcal{P}_1$ being a deterministic TIOTS. As quotient is a synthesis operator, it is difficult to give a definition using just *state-local* transition rules, since quotient needs global information about the transition systems. This is why we insist on $\mathcal{P}_1$ being deterministic[3].

Definition 4. *Let $\mathcal{P}_0$ and $\mathcal{P}_1$ be TIOTSs composable under $\otimes \in \{\|, \wedge, \vee, \%\}$. Then $\mathcal{P}_0 \otimes \mathcal{P}_1 = \langle I, O, S, s^0, \rightarrow\rangle$ is the TIOTS where:*

- *If $\otimes = \|$, then $I = (I_0 \cup I_1) \setminus O$ and $O = O_0 \cup O_1$*
- *If $\otimes \in \{\wedge, \vee\}$, then $I = I_0 = I_1$ and $O = O_0 = O_1$*
- *If $\otimes = \%$, then $I = I_0 \cup O_1$ and $O = O_0 \setminus O_1$*
- *$S = (P_0 \times P_1) \uplus P_0 \uplus P_1 \uplus \{\top, \bot\}$*
- *$s^0 = s_0^0 \otimes s_1^0$*
- *$\rightarrow$ is the smallest relation containing $\rightarrow_0 \cup \rightarrow_1$, and satisfying the rules:*

$$\frac{p_0 \xrightarrow{\alpha}_0 s_0' \quad p_1 \xrightarrow{\alpha}_1 s_1'}{p_0 \otimes p_1 \xrightarrow{\alpha} s_0' \otimes s_1'} \qquad \frac{p_0 \xrightarrow{a}_0 s_0' \quad a \notin A_1}{p_0 \otimes p_1 \xrightarrow{a} s_0' \otimes p_1} \qquad \frac{p_1 \xrightarrow{a}_0 s_1' \quad a \notin A_0}{p_0 \otimes p_1 \xrightarrow{a} p_0 \otimes s_1'}$$

We adopt the notation of $s_0 \otimes s_1$ for states, where the associated interpretation is supplied in Table 1. Furthermore, given two plain states $p_i = (l_i, t_i)$ for $i \in \{0,1\}$, we define $p_0 \times p_1 = ((l_0, l_1), t_0 \uplus t_1)$.

[2] As we write $A \sqsubseteq B$ to mean A is refined by B, our operators $\wedge$ and $\vee$ are reversed in comparison to the standard symbols for meet and join.

[3] Technically speaking, the problem lies in that state quotient operator is right-distributive but not left-distributive over state disjunction (cf Table 1).

Table 1 tells us how states should be combined under the composition operators. For parallel, a state is magic if one component state is magic, and a state is error if one component is error while the other is not magic. For conjunction, encountering error in one component implies the component can be discarded and the rest of the composition behaves like the other component. The conjunction table follows the intuition of the join operation on the refinement preorder. Similarly for disjunction. Quotient is the adjoint of parallel composition. If the second component state does not refine the first, the quotient will try to rescue the refinement by producing $\top$ (so that its composition with the second will refine the first). If the second component state does refine the first, the quotient will produce the least refined value so that its composition with the second will not break the refinement.

An *environment* for a TIOTS $\mathcal{P}$ is any TIOTS $\mathcal{Q}$ such that the alphabet of $\mathcal{Q}$ is *complementary* to that of $\mathcal{P}$, meaning $I_{\mathcal{P}} = O_{\mathcal{Q}}$ and $O_{\mathcal{P}} = I_{\mathcal{Q}}$. Refinement in our framework corresponds to contextual substitutability, in which the context is an arbitrary environment.

Definition 5. *Let $\mathcal{P}_{imp}$ and $\mathcal{P}_{spec}$ be TIOTSs with identical alphabets. $\mathcal{P}_{imp}$* refines *$\mathcal{P}_{spec}$, denoted $\mathcal{P}_{spec} \sqsubseteq \mathcal{P}_{imp}$, iff for all environments $\mathcal{Q}$, $\mathcal{P}_{spec} \parallel \mathcal{Q}$ is $\bot$-free implies $\mathcal{P}_{imp} \parallel \mathcal{Q}$ is $\bot$-free. We say $\mathcal{P}_{imp}$ and $\mathcal{P}_{spec}$ are* substitutively equivalent*, i.e. $\mathcal{P}_{spec} \simeq \mathcal{P}_{imp}$, iff $\mathcal{P}_{imp} \sqsubseteq \mathcal{P}_{spec}$ and $\mathcal{P}_{spec} \sqsubseteq \mathcal{P}_{imp}$.*

It is obvious that $\simeq$ induces an equivalance on TIOTSs and no equivalence that preserves the $\bot$ state can be weaker than $\simeq$. In the sequel we will give two concrete characterisations of $\simeq$ and show that $\simeq$ is also a congruence w.r.t. the parallel composition, conjunction, disjunction and quotient operators.

The operational definition of quotient requires that $\mathcal{P}_1$ is deterministic, which can be accomplished by a semantics-preserving determinisation procedure. We define the *determinisation* $\mathcal{P}^D$ of $\mathcal{P}$ as a modified subset construction procedure on $(\mathcal{P}^{\bot})^{\top}$: given a subset S_0 of states reachable by a given trace, we only keep those which are minimal w.r.t. the state refinement relation. So if the current state subset S_0 contains $\bot$, the procedure reduces S_0 to $\bot$; if $\bot \notin S_0 \neq \{\top\}$, it reduces S_0 by removing any potential $\top$ in S_0.[4]

Proposition 1. *Any TIOTS $\mathcal{P}$ is substitutively equivalent to the deterministic TIOTS $\mathcal{P}^D$.*

Equipped with determinisation, quotient is a fully defined operator on any pair of TIOTSs. Furthermore, we can give an alternative (although substitutively equivalent) formulation of quotient as the derived operator $(\mathcal{P}_0^{\neg} \parallel \mathcal{P}_1)^{\neg}$, where $\neg$ is a mirroring operation that first determinises its argument, then interchanges the input and output sets, as well as the $\top$ and $\bot$ states.

Example. Figure 2 shows the parallel composition of the job scheduler with the printer controller. In the transition from $B4$ to $A1$, the guard combines the effects of the constraints on the clocks x and y. As *finish* is an output of the controller, it can be fired at a time when the scheduler is not expecting it, meaning that a safety error will occur. This is indicated by the transition to $\bot$ when the guard constraint $5 \leq x \leq 8$ is not satisfied.

[4]For a more detailed definition of transforming non-deterministic systems into substitutivity-equivalent deterministic systems, we refer readers to the Definition 4.2 in [8]. That is for the untimed case.

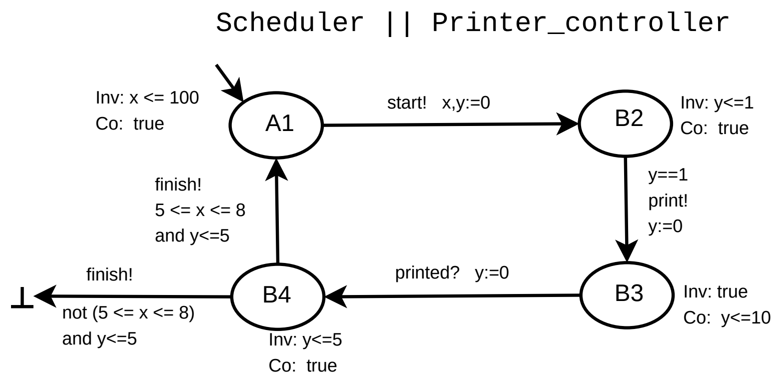

Figure 2: Parallel composition of the job scheduler and printer controller.

3 Timed I/O Game

Our specification theory can be understood from a game-theoretical point of view. It is an input-output game between a *component* and an *environment* that uses a *coin* to break ties. The specification of a component (in the form of a TIOA or TIOTS) is built to encode the set of strategies possible for the component in the game (just like an NFA encodes a set of words).

- Given two TIOTSs $\mathcal{P}$ and $\mathcal{Q}$ with identical alphabets, we say $\mathcal{P}$ is a partial unfolding [10] of $\mathcal{Q}$ if there exists a function f from $S_{\mathcal{P}}$ to $S_{\mathcal{Q}}$ s.t. 1) f maps $\top$ to $\top$, $\bot$ to $\bot$, and plain states to plain states, 2) $f(s^0_{\mathcal{P}}) = s^0_{\mathcal{Q}}$, and 3) $p \xrightarrow{\alpha}_{\mathcal{P}} s \Rightarrow f(p) \xrightarrow{\alpha}_{\mathcal{Q}} f(s)$.

- We say an acyclic TIOTS is a *tree* if 1) there does not exist a pair of transitions in the form of $p \xrightarrow{a} p''$ and $p' \xrightarrow{d} p''$, 2) $p \xrightarrow{a} p'' \wedge p' \xrightarrow{b} p''$ implies $p = p'$ and $a = b$ and 3) $p \xrightarrow{d} p'' \wedge p' \xrightarrow{d} p''$ implies $p = p'$.

- We say an acyclic TIOTS is a *simple path* if 1) $p \xrightarrow{a} s' \wedge p \xrightarrow{\alpha} s''$ implies $s' = s''$ and $a = \alpha$ and 2) $p \xrightarrow{d} s' \wedge p \xrightarrow{d} s''$ implies $s' = s''$.

- We say a simple path $\mathcal{L}$ is a *run* of $\mathcal{P}$ if $\mathcal{L}$ is a partial unfolding of $\mathcal{P}$.

Strategies. A *strategy* $\mathcal{G}$ is a deterministic tree TIOTS s.t. each plain state in $\mathcal{G}$ is ready to accept all possible inputs by the environment, but allows a single move (delay or output) by the component, i.e. $eb_{\mathcal{G}}(p) = I \uplus mv_{\mathcal{G}}(p)$ s.t. $mv_{\mathcal{G}}(p) = \{a\}$ for some $a \in O$ or $mv_{\mathcal{G}}(p) \subseteq \mathbb{R}^{>0}$, where $eb_{\mathcal{G}}(p)$ denotes the set of enabled timed actions in state p of LTS $\mathcal{G}$, and $mv_{\mathcal{G}}(p)$ denotes the unique component move allowed by $\mathcal{G}$ at p.

A TIOTS $\mathcal{P}$ *contains* a strategy $\mathcal{G}$ if $\mathcal{G}$ is a partial unfolding of $(\mathcal{P}^{\bot})^{\top}$. The set of strategies[5] contained in $\mathcal{P}$ is denoted $stg(\mathcal{P})$. Since it makes little sense to distinguish strategies that are isomorphic, we will freely use strategies to refer to their isomorphism classes and write $\mathcal{G} = \mathcal{G}'$ to mean $\mathcal{G}$ and $\mathcal{G}'$ are isomorphic.

Let us give some examples in Figure 3. For the sake of simplicity we use two untimed transition systems $\mathcal{P}$ and $\mathcal{Q}$, which have identical alphabets $I = \{e, f\}$ and $O = \{a, b, c\}$, to illustrate the idea

[5] In this paper we use a set of strategies (say Π) to mean a set of strategies with identical alphabets.

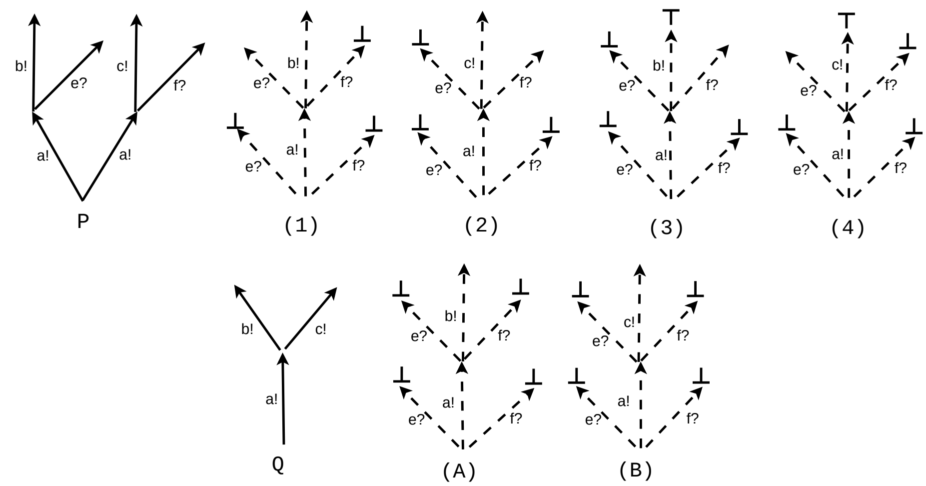

Figure 3: Strategy example.

of strategies. The transition systems use solid lines while strategies use dotted lines. Plain states are unmarked while the $\top$ and $\bot$ states are marked by $\top$ and $\bot$ resp.[6] We show four strategies of $\mathcal{P}$ and two strategies of $\mathcal{Q}$ on the right hand side of $\mathcal{P}$ and $\mathcal{Q}$ resp. in Figure 3. (They are not the complete sets of strategies for $\mathcal{P}$ and $\mathcal{Q}$.) Note that the strategies 3 and 4 owe their existence to the $\top$ completion.

Comparing strategies. When the game is played, the component tries to avoid reaching $\top$ while the environment tries to avoid reaching $\bot$. Different strategies in $stg(\mathcal{P})$ vary in their effectiveness to achieve the objective. Such effectiveness can be compared if two strategies closely resemble each other: we say $\mathcal{G}$ and $\mathcal{G}'$ are *affine* if $s^0_{\mathcal{G}} \stackrel{tt}{\Rightarrow} p$ and $s^0_{\mathcal{G}'} \stackrel{tt}{\Rightarrow} p'$ implies $mv_{\mathcal{G}}(p) = mv_{\mathcal{G}'}(p')$. Intuitively, it means $\mathcal{G}$ and $\mathcal{G}'$ propose the same move at the 'same' states. For instance, the strategies 1, 3 and A in Figure 3 are pairwise affine and so are the strategies 2, 4 and B.

Given two affine strategies $\mathcal{G}$ and $\mathcal{G}'$, we say $\mathcal{G}$ is *more aggressive* than $\mathcal{G}'$, denoted $\mathcal{G} \preceq \mathcal{G}'$, if 1) $s^0_{\mathcal{G}'} \stackrel{tt}{\Rightarrow} \bot$ implies there is a prefix tt_0 of tt s.t. $s^0_{\mathcal{G}} \stackrel{tt_0}{\Longrightarrow} \bot$ and 2) $s^0_{\mathcal{G}} \stackrel{tt}{\Rightarrow} \top$ implies there is a prefix tt_0 of tt s.t. $s^0_{\mathcal{G}'} \stackrel{tt_0}{\Longrightarrow} \top$. Intuitively, it means $\mathcal{G}$ can reach $\bot$ faster but $\top$ slower than $\mathcal{G}'$. $\preceq$ forms a partial order over $stg(\mathcal{P})$, or more generally, over any set of strategies with identical alphabets. For instance, strategy A is more aggressive than 1 and 3, while strategy B is more aggressive than 2 and 4.

When the game is played, the component $\mathcal{P}$ prefers to use the maximally aggressive strategies in $stg(\mathcal{P})$[7]. Thus two components that differ only in non-maximally aggressive strategies should be equated. We define the *strategy semantics* of component $\mathcal{P}$ to be $[\mathcal{P}]_s = \{\mathcal{G}' \mid \exists \mathcal{G} \in stg(\mathcal{P}) : \mathcal{G} \preceq \mathcal{G}'\}$, i.e. the upward-closure of $stg(\mathcal{P})$ w.r.t. $\preceq$.

Game rules. When a component strategy $\mathcal{G}$ is played against an environment strategy $\mathcal{G}'$, at each game state (i.e. a product state $p_{\mathcal{G}} \times p_{\mathcal{G}'}$) $\mathcal{G}$ and $\mathcal{G}'$ each propose a move (i.e. $mv_{\mathcal{G}}(p_{\mathcal{G}})$ and $mv_{\mathcal{G}'}(p_{\mathcal{G}'})$). If one of them is a delay and the other is an action, the action will prevail. If both

[6]To simplify drawing, multiple copies of $\top$ and $\bot$ are allowed but the self-loops on them are omitted.
[7]This is because our semantics is designed to preserve $\bot$ rather than $\top$.

propose delay moves (i.e. $mv_{\mathcal{G}}(p_{\mathcal{G}}), mv_{\mathcal{G}'}(p_{\mathcal{G}'}) \subseteq \mathbb{R}^{\geq 0}$), the smaller one (w.r.t. set containment) will prevail.[8]

Since a delay move proposed at a strategy state is the maximal set of possible delays enabled at that state, the next move proposed at the new state after firing the set must be an action move (due to time additivity). Thus a play cannot have two consecutive delay moves.

If, however, both propose action moves, there will be a tie, which will be resolved by tossing the coin. For uniformity's sake, the coin can be treated as a special component. A strategy of the coin is a function h from tA^* to $\{0, 1\}$. We denote the set of all possible coin strategies as H.

A play of the game can be formalised as a composition of three strategies, one each from the component, environment and coin, denoted $\mathcal{G}_{\mathcal{P}} \parallel_h \mathcal{G}_{\mathcal{Q}}$. At a current game state $p_{\mathcal{P}} \times p_{\mathcal{Q}}$, if the prevailing action is α and we have $p_{\mathcal{P}} \xrightarrow{\alpha} s'_{\mathcal{P}}$ and $p_{\mathcal{Q}} \xrightarrow{\alpha} s'_{\mathcal{Q}}$, then the next game state is $s_{\mathcal{P}} \parallel s_{\mathcal{Q}}$. The play will stop when it reaches either $\top$ or $\perp$. The composition will produce a simple path $\mathcal{L}$ that is a run of $\mathcal{P} \parallel \mathcal{Q}$. Since $\mathcal{P} \parallel \mathcal{Q}$ gives rise to a *closed system* (i.e. the input alphabet is empty), a run of $\mathcal{P} \parallel \mathcal{Q}$ is a strategy of $\mathcal{P} \parallel \mathcal{Q}$.

This is crucial since it reveals that strategy composition of $\mathcal{P}$ and $\mathcal{Q}$ is closely related to their parallel composition: $stg(\mathcal{P} \parallel \mathcal{Q}) = \{\mathcal{G}_{\mathcal{P}} \parallel_h \mathcal{G}_{\mathcal{Q}} \mid \mathcal{G}_{\mathcal{P}} \in stg(\mathcal{P}), \mathcal{G}_{\mathcal{Q}} \in stg(\mathcal{Q}) \text{ and } h \in H\}$.

Parallel composition. Strategy composition, like component (parallel) composition, can be generalised to any pair of components $\mathcal{P}$ and $\mathcal{Q}$ with *composable alphabets*. That is, $O_{\mathcal{P}} \cap O_{\mathcal{Q}} = \{\}$. For such $\mathcal{P}$ and $\mathcal{Q}$, $\mathcal{G}_{\mathcal{P}} \parallel_h \mathcal{G}_{\mathcal{Q}}$ gives rise to a tree rather than a simple path TIOTS. That is, at each game state $p_{\mathcal{P}} \times p_{\mathcal{Q}}$, besides firing the prevailing $\alpha \in tO_{\mathcal{P}} \cup tO_{\mathcal{Q}}$, we need also to fire 1) all the synchronised inputs, i.e. $e \in I_{\mathcal{P}} \cap I_{\mathcal{Q}}$, and reach the new game state $s_{\mathcal{P}} \parallel s_{\mathcal{Q}}$ (assuming $p_{\mathcal{P}} \xrightarrow{e} s_{\mathcal{P}}$ and $p_{\mathcal{Q}} \xrightarrow{e} s_{\mathcal{Q}}$) and 2) all the independent inputs, i.e. $e \in (I_{\mathcal{P}} \cup I_{\mathcal{Q}}) \setminus (A_{\mathcal{P}} \cap A_{\mathcal{Q}})$, and reach the new game state $s_{\mathcal{P}} \times p_{\mathcal{Q}}$ or $p_{\mathcal{P}} \times s_{\mathcal{Q}}$. It is easy to verify that $\mathcal{G}_{\mathcal{P}} \parallel_h \mathcal{G}_{\mathcal{Q}}$ is a strategy of $\mathcal{P} \parallel \mathcal{Q}$.

Conjunction/disjunction. Besides strategy composition, *strategy conjunction* (&) and *strategy disjunction* (+) are also definable. They are binary operators defined only on pairs of affine strategies. We define $\mathcal{G} \& \mathcal{G}' = \mathcal{G} \wedge \mathcal{G}'$ and $\mathcal{G} + \mathcal{G}' = \mathcal{G} \vee \mathcal{G}'$. Note that, if $\mathcal{G}$ and $\mathcal{G}'$ are not affine, $\mathcal{G} \wedge \mathcal{G}'$ and $\mathcal{G} \vee \mathcal{G}'$ do not necessarily produce a strategy. For instance the disjunction of the strategies 1 and 2 in Figure 3 will produce a transition system that stops to output after the a transition.

Refinement. Strategy semantics induces an equivalence on TIOTSs. That is, $\mathcal{P}$ and $\mathcal{Q}$ are *strategy equivalent* iff $[\mathcal{P}]_s = [\mathcal{Q}]_s$. However, strategy equivalence is too fine for the purpose of *substitutive refinement* (cf Definition 5). For instance, transition systems $\mathcal{P}$ and $\mathcal{Q}$ in Figure 3 are substitutively equivalent, but are not strategy equivalent, because 1, 2, 3 and 4 are strategies of $\mathcal{Q}$ (due to upward-closure w.r.t. $\preceq$), and A and B are not strategies of $\mathcal{P}$.

However, we demonstrate that *substitutive equivalence is reducible to strategy equivalence* providing we perform *disjunction closure* on strategies.

Lemma 1. *Given a pair of affine component strategies $\mathcal{G}_0$ and $\mathcal{G}_1$, $\mathcal{G}_0 \parallel_h \mathcal{G}$ and $\mathcal{G}_1 \parallel_h \mathcal{G}$ are $\perp$-free for a pair of environment and coin strategies $\mathcal{G}$ and h iff $\mathcal{G}_0 + \mathcal{G}_1 \parallel_h \mathcal{G}$ is $\perp$-free.*

[8] Note that all invariants and co-invariants are downward-closed. Thus a delay move can be respresented as a time interval from 0 to some $d \in \mathbb{R}^{\geq 0}$.

We say Π^+ is a *disjunction closure* of Π iff it is the least superset of Π s.t. $\mathcal{G} + \mathcal{G}' \in \Pi^+$ for all pairs of affine strategies $\mathcal{G}, \mathcal{G}' \in \Pi^+$. It is easy to see the disjunction closure operation preserves the upward-closedness of strategy sets.

Proposition 2. *Disjunction closure is determinisation:* $[\mathcal{P}^D]_s = [\mathcal{P}^D]_s^+ = [\mathcal{P}]_s^+$.

Lemma 2. *For any TIOTS* $\mathcal{P}$, $[\mathcal{P}^\neg]_s^+ = \{\mathcal{G}_{\mathcal{P}^\neg} \mid \forall \mathcal{G}_\mathcal{P} \in [\mathcal{P}]_s^+, h \in H : \mathcal{G}_{\mathcal{P}^\neg} \parallel_h \mathcal{G}_\mathcal{P}$ *is* $\bot$*-free*$\}$.

Theorem 1. *Given TIOTSs* $\mathcal{P}$ *and* $\mathcal{Q}$, $\mathcal{P} \sqsubseteq \mathcal{Q}$ *iff* $[\mathcal{Q}]_s^+ \subseteq [\mathcal{P}]_s^+$.

For instance, the disjunction of strategies 1 and 3 produces A, while the disjunction of strategies 2 and 4 produces B. Thus $[\mathcal{P}]_s^+ = [\mathcal{Q}]_s^+$.

Relating operational composition to strategies. The operations of parallel composition, conjunction and disjunction defined on the operational models of TIOTSs (Section 2.4) can be characterised by simple operations on strategies in the game-based setting.

Lemma 3. *For* $\parallel$*-composable TIOTSs* $\mathcal{P}$ *and* $\mathcal{Q}$, $[\mathcal{P} \parallel \mathcal{Q}]_s^+ = \{\mathcal{G}_{\mathcal{P}\parallel\mathcal{Q}} \mid \exists \mathcal{G}_\mathcal{P} \in [\mathcal{P}]_s^+, \mathcal{G}_\mathcal{Q} \in [\mathcal{Q}]_s^+, h \in H : \mathcal{G}_\mathcal{P} \parallel_h \mathcal{G}_\mathcal{Q} \preceq \mathcal{G}_{\mathcal{P}\parallel\mathcal{Q}}\}$.

Lemma 4. *For* $\vee$*-composable TIOTSs* $\mathcal{P}$ *and* $\mathcal{Q}$, $[\mathcal{P} \vee \mathcal{Q}]_s^+ = ([\mathcal{P}]_s^+ \cup [\mathcal{Q}]_s^+)^+$.

Lemma 5. *For* $\wedge$*-composable TIOTSs* $\mathcal{P}$ *and* $\mathcal{Q}$, $[\mathcal{P} \wedge \mathcal{Q}]_s^+ = [\mathcal{P}]_s^+ \cap [\mathcal{Q}]_s^+$.

Lemma 6. *For* $\%$*-composable TIOTSs* $\mathcal{P}$ *and* $\mathcal{Q}$, $[\mathcal{P}\%\mathcal{Q}]_s^+ = \{\mathcal{G}_{\mathcal{P}\%\mathcal{Q}} \mid \forall \mathcal{G}_\mathcal{Q} \in [\mathcal{Q}]_s^+, h \in H : \mathcal{G}_{\mathcal{P}\%\mathcal{Q}} \parallel_h \mathcal{G}_\mathcal{Q} \in [\mathcal{P}]_s^+\}$.

Thus conjunction and disjunction are the join and meet operations and quotient produces the coarsest TIOTS s.t. $(\mathcal{P}_0\%\mathcal{P}_1)\|\mathcal{P}_1$ is a refinement of $\mathcal{P}_0$.

Theorem 2. $\simeq$ *is a congruence w.r.t.* $\parallel$, $\vee$, $\wedge$ *and* $\%$ *subject to composability.*

Summary. Strategy semantics has given us a weakest $\bot$-preserving congruence (i.e. $[\mathcal{P}]_s^+$) for timed specification theories based on operators for (parallel) composition, conjunction, disjunction and quotient. Strategy semantics captures nicely the game-theoretical nature as well as the operational intuition of the specification theories. However, in a more declarative manner, the equivalence can also be characterised by timed traces, as we see in the next section.

4 Declarative Specification Theory

In this section, we develop a compositional specification theory based on timed traces. We introduce the concept of a timed-trace structure, which is an abstract representation for a timed component. The timed-trace structure contains essential information about the component, for checking whether it can be substituted with another in a safety and liveness preserving manner.

Given any TIOTS $\mathcal{P} = \langle I, O, S, s^0, \rightarrow \rangle$, we can extract three sets of traces from $(\mathcal{P}^\bot)^\top$: *TP* (plain traces) is a set of timed traces leading to plain states, *TE* (error traces) a set of timed traces leading to $\bot$ and *TM* (magic traces) a set of timed traces leading to $\top$. *TE* and *TM* are extension-closed[9] while TP is prefix-closed. It is easy to verify $TE \cup TP \cup TM$ gives rise to the full set of timed traces tA^*.[10] Thus TP and TE suffice to capture all the information.

However, TP and TE contain more information than necessary for our substitutive refinement, which is designed to preserve $\bot$ rather than $\top$. For instance, adding any trace $tt \in TE$ to TP should not change the semantics of the component. Based on a slight abstraction of the two sets we can define a *dual-trace structure* as the semantics of $\mathcal{P}$.

Definition 6 (Dual-trace structure)**.** *$\mathcal{TT}(\mathcal{P}) := (I, O, TR, TE)$, where $TR := TE \cup TP$ the set of realisable traces. Obviously, TR is prefix-closed.*

From hereon let $\mathcal{P}_0$ and $\mathcal{P}_1$ be two TIOTSs with dual-trace structures $\mathcal{TT}(\mathcal{P}_i) := (I_i, O_i, TR_i, TE_i)$ for $i \in \{0, 1\}$. Define $\bar{i} = 1 - i$.

The substitutive refinement relation $\sqsubseteq$ in Section 2.4 can equally be characterised by means of trace containment. Consequently, $\mathcal{TT}(\mathcal{P}_0)$ can be regarded as providing an alternative encoding of the set $[\mathcal{P}_0]_s^+$ of strategies.

Theorem 3. *$\mathcal{P}_0 \sqsubseteq \mathcal{P}_1$ iff $TR_1 \subseteq TR_0$ and $TE_1 \subseteq TE_0$.*

We are now ready to define the timed-trace structure semantics for the operators of our specification theory. Intuitively, the timed-trace semantics mimics the synchronised product of the operational definitions in Section 2.4. An important fact utilised in formulating these operations on traces is that, for any trace $tt \in tA^*$ and TIOTS $\mathcal{P}$, either tt is a trace of $\mathcal{P}$ or there is some prefix tt_0 of tt s.t. tt_0 is an error or magic trace of $\mathcal{P}$.

Parallel composition. The idea behind parallel composition is that the projection of any trace in the composition onto the alphabet of one of the components should be a trace of that component.

Proposition 3. *If $\mathcal{P}_0$ and $\mathcal{P}_1$ are $\|$-composable, then $\mathcal{TT}(\mathcal{P}_0 \parallel \mathcal{P}_1) = (I, O, TR, TE)$ where $I = (I_0 \cup I_1) \setminus O$, $O = O_0 \cup O_1$ and the trace sets are given by:*

- $TE = \{tt \mid tt \upharpoonright tA_i \in TE_i \wedge tt \upharpoonright tA_{\bar{i}} \in TR_{\bar{i}}\} \cdot tA^*$
- $TR = TE \uplus \{tt \mid tt \upharpoonright tA_i \in (TR_i \setminus TE_i) \wedge tt \upharpoonright tA_{\bar{i}} \in (TR_{\bar{i}} \setminus TE_{\bar{i}})\}$

The above says tt is an error trace if the projection of tt on one component is an error trace while the projection of tt on the other component is a realisable trace. tt is a realisable trace if tt is either an error trace or a (strictly) plain trace. tt is a (strictly) plain trace if the projection of tt on both components are (strictly) plain traces.

Disjunction. From any composite state in the disjunction of two components, the composition should only be willing to accept inputs that are accepted by both components, but should accept the union of outputs. After witnessing an output enabled by only one of the components, the disjunction should behave like that component. Because of the way that $\bot$ and $\top$ work in Table 1, this loosely corresponds to taking the union of the traces from the respective components.

Proposition 4. *If $\mathcal{P}_0$ and $\mathcal{P}_1$ are $\vee$-composable, then $\mathcal{TT}(\mathcal{P}_0 \vee \mathcal{P}_1) = (I, O, TR_0 \cup TR_1, TE_0 \cup TE_1)$, where $I = I_0 = I_1$ and $O = O_0 = O_1$.*

[9] This is due to the fact that $\top$ and $\bot$ are modelled as chaotic states.
[10] This is due to $\top/\bot$-completion.

Conjunction. Similarly to disjunction, from any composite state in the conjunction of two components, the composition should only be willing to accept outputs that are accepted by both components, and should accept the union of inputs, until a stage when one of the component's input assumptions has been violated, after which it should behave like the other component. Because of the way that both $\bot$ and $\top$ work in Table 1, this essentially corresponds to taking the intersection of the traces from the respective components.

Proposition 5. *If $\mathcal{P}_0$ and $\mathcal{P}_1$ are $\wedge$-composable, then $\mathcal{TT}(\mathcal{P}_0 \wedge \mathcal{P}_1) = (I, O,\ TR_0 \cap TR_1, TE_0 \cap TE_1)$, where $I = I_0 = I_1$ and $O = O_0 = O_1$.*

Quotient. Quotient ensures its composition with the second component is a refinement of the first. Given the synchronised running of $\mathcal{P}_0$ and $\mathcal{P}_1$, if $\mathcal{P}_0$ is in a more refined state than $\mathcal{P}_1$, the quotient will try to rescue the refinement by taking $\top$ as its state (so that its composition with $\mathcal{P}_1$'s state will refine $\mathcal{P}_0$'s). If $\mathcal{P}_0$ is in a less or equally refined state than $\mathcal{P}_1$'s, the quotient will take the worst possible state without breaking the refinement.

Proposition 6. *If $\mathcal{P}_0$ dominates $\mathcal{P}_1$, then $\mathcal{TT}(\mathcal{P}_0 \% \mathcal{P}_1) = (I, O, TR, TE)$, where $I = I_0 \cup O_1$, $O = O_0 \setminus O_1$, and the trace sets satisfy:*

- $TE = TE_0 \cup \{tt \mid tt \upharpoonright tA_1 \notin TR_1\} \cdot tA^*$
- $TR = TE \uplus \{tt \mid tt \in (TR_0 \setminus TE_0) \wedge tt \upharpoonright tA_1 \in (TR_1 \setminus TE_1)\}$.

The above says tt is an error trace if either 1) tt is an error trace in $\mathcal{P}_0$, or 2) the projection of tt on $\mathcal{P}_1$ is not a realisable trace.

Mirroring of dual-trace structures is straightforward: $\mathcal{TT}(\mathcal{P}_0)^\neg = (O_0, I_0, tA^* \setminus TE_0, tA^* \setminus TR_0)$. Consequently, quotient can also be defined as the derived operator $(\mathcal{TT}(\mathcal{P}_0)^\neg \parallel \mathcal{TT}(\mathcal{P}_1))^\neg$.

5 Comparison with Related Works

Our framework can be seen as a linear-time alternative to the timed specification theories of [2] and [3], albeit with significant differences. The specification theory in [3] also introduces parallel, conjunction and quotient, but uses timed alternating simulation as refinement, which does not admit the weakest precongruence. An advantage of [3] is the algorithmic efficiency of branching-time simulation checking and implementation reported in [11].

The work of [2] on timed games also bears conceptual similarities, although they do not define conjunction and quotient. We adopt most of the game rules in [2], except that, due to our requirement that proposed delay moves are maximal delays allowed by a strategy, a play cannot have consecutive delay moves. This enables us to avoid the complexity of time-blocking strategies and blame assignment, but does not ensure non-Zenoness[11]. Secondly, we do not use timestop/semi-timestop to model time errors (i.e. bounded-liveness errors). Rather, we introduce the explicit inconsistent state $\bot$ to model both time and immediate (i.e. safety) errors. This enables us to avoid the complexity of having two transition relations and well-formedness of timed interfaces.

Based on linear time, our timed theory owes much to the pioneering work of trace theories in asynchronous circuit verification, such as Dill's trace theory [9]. Our mirror operator is essentially

[11]Zeno behaviours (infinite action moves within finite time) in a play are not regarded as abnormal behaviours in our semantics.

a timed extension of the mirror operator from asynchronous circuit verification [12]. The definition of quotient based on mirroring (for the untimed case) was first presented by Verhoeff as his Factorisation Theorem [13].

In comparison with our untimed theory [1], our timed extension requires new techniques (e.g. those related to timestop) to handle delay transitions since time can be modelled neither as input nor as output. In the timed theory the set of realisable traces (TR) is not required to be input-enabled, which is necessary for the set of untimed traces in [1]. Thus, the domain of trace structures are significantly enlarged. Furthermore, the timed theory supports the modelling of liveness assumption/guarantee. It can further reduce such checking of liveness (assumption/guarantee) mismatches to the $\perp$-reachability. Therefore, finite traces suffice to model and verify liveness properties. In contrast, the untimed theory must employ infinite traces to treat liveness in a proper way.

We briefly mention other related works, which include timed modal transition systems [14, 15], the timed I/O model [5, 16] and embedded systems [17, 18].

6 Conclusions

We have formulated a rich compositional specification theory for components with real-time constraints based on a linear-time notion of substitutive refinement. The operators of hiding and renaming can also be defined, based on our previous work [8]. We believe that our theory can be reformulated as a timed extension of Dill's trace theory [9]. Future work will include an investigation of realisability and assume-guarantee reasoning.

Acknowledgments. The authors are supported by EU FP7 project CONNECT, ERC Advanced Grant VERIWARE and EPSRC project EP/F001096.

References

[1] Chen, T., Chilton, C., Jonsson, B., Kwiatkowska, M.: A compositional specification theory for component behaviours. In: ESOP'12. Volume 7211 of LNCS., Springer-Verlag (2012) 148–168

[2] de Alfaro, L., Henzinger, T.A., Stoelinga, M.: Timed interfaces. In: EMSOFT'02. Volume 2491 of LNCS. Springer-Verlag (2002) 108–122

[3] David, A., Larsen, K.G., Legay, A., Nyman, U., Wasowski, A.: Timed I/O automata: a complete specification theory for real-time systems. In: HSCC '10, ACM (2010) 91–100

[4] de Alfaro, L., Henzinger, T.A.: Interface automata. SIGSOFT Softw. Eng. Notes **26** (2001) 109–120

[5] Kaynar, D.K., Lynch, N.A., Segala, R., Vaandrager, F.W.: Timed I/O Automata: A mathematical framework for modeling and analyzing real-time systems. In: RTSS. (2003)

[6] Lim, W.: Design methodology for stoppable clock systems. Computers and Digital Techniques, IEE Proceedings E **133** (1986) 65 –72

[7] Moore, S., Taylor, G., Cunningham, P., Mullins, R., Robinson, P.: Using stoppable clocks to safely interface asynchronous and synchronous subsystems. In: AINT (Asynchronous INTerfaces) Workshop, Delft, Netherlands (2000)

[8] Wang, X., Kwiatkowska, M.Z.: On process-algebraic verification of asynchronous circuits. Fundam. Inform. **80** (2007) 283–310

[9] Dill, D.L.: Trace theory for automatic hierarchical verification of speed-independent circuits. ACM distinguished dissertations. MIT Press (1989)

[10] Wang, X.: Maximal Confluent Processes. In: Petri Nets'12. Volume 7347 of LNCS., Springer-Verlag (2012)

[11] David, A., Larsen, K.G., Legay, A., Nyman, U., Wasowski, A.: Ecdar: An environment for compositional design and analysis of real time systems. In: ATVA. Volume 6252 of LNCS., Springer (2010) 365–370

[12] Zhou, B., Yoneda, T., Myers, C.: Framework of timed trace theoretic verification revisited. IEICE Trans. on Information and Systems **85** (2002) 1595–1604

[13] Verhoeff, T.: A Theory of Delay-Insensitive Systems. PhD thesis, Dept. of Math. and C.S., Eindhoven Univ. of Technology (1994)

[14] Bertrand, N., Legay, A., Pinchinat, S., Raclet, J.B.: A compositional approach on modal specifications for timed systems. In: ICFEM. Volume 5885 of LNCS., Springer (2009) 679–697

[15] Cerans, K., Godskesen, J.C., Larsen, K.G.: Timed modal specification - theory and tools. In: CAV. (1993) 253–267

[16] Berendsen, J., Vaandrager, F.W.: Compositional abstraction in real-time model checking. In: FORMATS. Volume 5215 of LNCS., Springer (2008) 233–249

[17] Thiele, L., Wandeler, E., Stoimenov, N.: Real-time interfaces for composing real-time systems. In: EMSOFT. (2006)

[18] Lee, I., Leung, J., Song, S.: Handbook of Real-Time and Embedded Systems. Chapman (2007)

A The proofs

Proposition 1. *Any TIOTS $\mathcal{P}$ is substitutively equivalent to the deterministic TIOTS $\mathcal{P}^D$.*

Proof. If $\mathcal{P}$ can run a trace tt and reach $\bot$, then the subset S_0 of states reachable by tt in $\mathcal{P}$ will be reduced to $\bot$ by the modified subset constrcution. Thus a run of tt on $\mathcal{P}^D$ will lead to $\bot$.

If $\mathcal{P}^D$ can run a trace tt and reach $\bot$, then the subset S_0 of states reachable by tt in $\mathcal{P}$ contains $\bot$. Thus there exists a run of tt on $\mathcal{P}$ which leads to $\bot$. □

Lemma 1. *Given a pair of affine component strategies $\mathcal{G}_0$ and $\mathcal{G}_1$, $\mathcal{G}_0 \parallel_h \mathcal{G}$ and $\mathcal{G}_1 \parallel_h \mathcal{G}$ are $\bot$-free for a pair of environment and coin strategies $\mathcal{G}$ and h iff $\mathcal{G}_0 + \mathcal{G}_1 \parallel_h \mathcal{G}$ is $\bot$-free.*

Proof. From $\mathcal{G}_0 + \mathcal{G}_1 \parallel_h \mathcal{G}$ to $\mathcal{G}_0 \parallel_h \mathcal{G}$ and $\mathcal{G}_1 \parallel_h \mathcal{G}$: obvious since $\mathcal{G}_0 + \mathcal{G}_1$ is more aggressive than both $\mathcal{G}_0$ and $\mathcal{G}_1$.

From $\mathcal{G}_0 \parallel_h \mathcal{G}$ and $\mathcal{G}_1 \parallel_h \mathcal{G}$ to $\mathcal{G}_0 + \mathcal{G}_1 \parallel_h \mathcal{G}$: If $\mathcal{G}_0 + \mathcal{G}_1 \parallel_h \mathcal{G}$ gives rise to a simple path ending in $\bot$, it is impossible that $\mathcal{G}_0 \parallel_h \mathcal{G}$ and $\mathcal{G}_1 \parallel_h \mathcal{G}$ both give rise to a $\bot$-free simple path since state $\parallel$ operation distributes over state $\vee$ operation. $\square$

Proposition 2. *Disjunction closure is determinisation:* $[\mathcal{P}^D]_s = [\mathcal{P}^D]_s^+ = [\mathcal{P}]_s^+$.

Proof. Follows from Lemma 1. $\square$

Lemma 2. *For any TIOTS $\mathcal{P}$, $[\mathcal{P}^\neg]_s^+ = \{\mathcal{G}_{\mathcal{P}^\neg} \mid \forall \mathcal{G}_\mathcal{P} \in [\mathcal{P}]_s^+, h \in H : \mathcal{G}_{\mathcal{P}^\neg} \parallel_h \mathcal{G}_\mathcal{P}$ is $\bot$-free$\}$.*

Proof. Since $\mathcal{P}^\neg$ is deterministic, $[\mathcal{P}^\neg]_s^+ = [\mathcal{P}^\neg]_s$. If there exists $\mathcal{G} \notin [\mathcal{P}^\neg]_s$ s.t. for all $\mathcal{G}_\mathcal{P} \in [\mathcal{P}^D]_s, h \in H$ we have $\mathcal{G} \parallel_h \mathcal{G}_\mathcal{P}$ is $\bot$-free, then $\mathcal{G}$ is $\preceq$-incomparable to all strategies in $stg(\mathcal{P}^\neg)$. Thus there exists a trace tt s.t. the state s reached after tt in $\mathcal{P}^\neg$ strictly refines the state s' reached after tt in $\mathcal{G}$. If $s = \top$ and $s' \neq \top$, then the state s'' reached after tt in $\mathcal{P}^D$ is $\bot$ and $s' \parallel s'' = \bot$. Contradition! If $s = p$ and $s' = \bot$, then the state s'' reached after tt in $\mathcal{P}^D$ is p' and $s' \parallel s'' = \bot$. Contradition! $\square$

Theorem 1. *Given TIOTSs $\mathcal{P}$ and $\mathcal{Q}$, $\mathcal{P} \sqsubseteq \mathcal{Q}$ iff $[\mathcal{Q}]_s^+ \subseteq [\mathcal{P}]_s^+$.*

Proof. Due to propositions 1 and 2, the proof is reduced to prove $\mathcal{P}^D \sqsubseteq \mathcal{Q}^D$ iff $[\mathcal{Q}^D]_s \subseteq [\mathcal{P}^D]_s$.

From $[\mathcal{Q}^D]_s \subseteq [\mathcal{P}^D]_s$ to $\mathcal{P}^D \sqsubseteq \mathcal{Q}^D$: If $\mathcal{P}^D \sqsubseteq \mathcal{Q}^D$ is not true, there exists an environment strategy $\mathcal{G}_E$ s.t. $\mathcal{G}_E \parallel \mathcal{P}^D$ is $\bot$-free but $\mathcal{G}_E \parallel \mathcal{Q}^D$ is not. Then there exists $\mathcal{G} \in stg(\mathcal{Q}^D)$ and $h \in H$ s.t. $\mathcal{G} \parallel_h \mathcal{G}_E$ is not $\bot$-free. Obviously $\mathcal{G} \notin [\mathcal{P}^D]_s$. Contradiction!

From $\mathcal{P}^D \sqsubseteq \mathcal{Q}^D$ to $[\mathcal{Q}^D]_s \subseteq [\mathcal{P}^D]_s$: If $[\mathcal{Q}^D]_s \setminus [\mathcal{P}^D]_s$ is non-empty, then there exists $\mathcal{G} \in stg(\mathcal{Q}^D) \setminus [\mathcal{P}^D]_s$. Obviously $\mathcal{G} \parallel (\mathcal{P}^D)^\neg$ is not $\bot$-free. Thus $\mathcal{Q}^D \parallel (\mathcal{P}^D)^\neg$ is not $\bot$-free but $\mathcal{P}^D \parallel (\mathcal{P}^D)^\neg$ is. Contradiction! $\square$

Lemma 3. *For $\parallel$-composable TIOTSs $\mathcal{P}$ and $\mathcal{Q}$, $[\mathcal{P} \parallel \mathcal{Q}]_s^+ = \{\mathcal{G}_{\mathcal{P}\parallel\mathcal{Q}} \mid \exists \mathcal{G}_\mathcal{P} \in [\mathcal{P}]_s^+, \mathcal{G}_\mathcal{Q} \in [\mathcal{Q}]_s^+, h \in H : \mathcal{G}_\mathcal{P} \parallel_h \mathcal{G}_\mathcal{Q} \preceq \mathcal{G}_{\mathcal{P}\parallel\mathcal{Q}}\}$.*

Proof. Due to the distributivity of state $\parallel$ operation over state $\vee$ operation, $(\mathcal{P} \parallel \mathcal{Q})^D$ and $\mathcal{P}^D \parallel \mathcal{Q}^D$ are isomorphic. Thus $[\mathcal{P} \parallel \mathcal{Q}]_s^+ = [(\mathcal{P} \parallel \mathcal{Q})^D]_s = [\mathcal{P}^D \parallel \mathcal{Q}^D]_s = (stg(\mathcal{P}^D \parallel \mathcal{Q}^D))^\uparrow = (\{G_{\mathcal{P}^D} \parallel_h G_{\mathcal{Q}^D} \mid G_\mathcal{P} \in stg(\mathcal{P}^D), G_\mathcal{Q} \in stg(\mathcal{Q}^D),\text{ and } h \in H\})^\uparrow = \{\mathcal{G}_{\mathcal{P}\parallel\mathcal{Q}} \mid \exists \mathcal{G}_\mathcal{P} \in [\mathcal{P}]_s^+, \mathcal{G}_\mathcal{Q} \in [\mathcal{Q}]_s^+, h \in H : \mathcal{G}_\mathcal{P} \parallel_h \mathcal{G}_\mathcal{Q} \preceq \mathcal{G}_{\mathcal{P}\parallel\mathcal{Q}}\}$, where $\Pi^\uparrow$ is the $\preceq$-upward closure operation. $\square$

Lemma 4. *For $\vee$-composable TIOTSs $\mathcal{P}$ and $\mathcal{Q}$, $[\mathcal{P} \vee \mathcal{Q}]_s^+ = ([\mathcal{P}]_s^+ \cup [\mathcal{Q}]_s^+)^+$.*

Proof. Since $(\mathcal{P} \vee \mathcal{Q})^D$ and $\mathcal{P}^D \vee \mathcal{Q}^D$ are isomorphic, $[\mathcal{P} \vee \mathcal{Q}]_s^+ = [(\mathcal{P} \vee \mathcal{Q})^D]_s = [\mathcal{P}^D \vee \mathcal{Q}^D]_s$.

$[\mathcal{P}^D \vee \mathcal{Q}^D]_s \subseteq ([\mathcal{P}^D]_s \cup [\mathcal{Q}^D]_s)^+$: If $\mathcal{G} \in stg(\mathcal{P}^D \vee \mathcal{Q}^D)$, $\mathcal{G}$ can be decomposed into $\mathcal{G}_0 \in stg(\mathcal{P}^D)$ and $\mathcal{G}_1 \in stg(\mathcal{Q}^D)$ s.t. $\mathcal{G} = \mathcal{G}_0 + \mathcal{G}_1$. Starting from the initial state of $\mathcal{G}$ and in a breadth-first manner, decompose (product) states s of $\mathcal{G}$ into (component) states of $\mathcal{G}_0$ and $\mathcal{G}_1$. If $s = (p_0, p_1)$, then p_0 for $\mathcal{G}_0$ and p_1 for $\mathcal{G}_1$. If $((p_0, p_1) \xrightarrow{\alpha})\ s = p_i'$ for $i \in \{0, 1\}$ then p_i' for G_i and $\top$ for $G_{\bar{i}}$. If $((p_0, p_1) \xrightarrow{\alpha})\ s = \top$, then $\top$ for both $\mathcal{G}_0$ and $\mathcal{G}_1$. If $((p_0, p_1) \xrightarrow{\alpha})\ s = \bot$, then $\bot$ for $\mathcal{G}_0$ or $\mathcal{G}_1$ depending on $f_0(p_0) \xrightarrow{\alpha}_{\mathcal{P}^D} \bot$ or $f_1(p_1) \xrightarrow{\alpha}_{\mathcal{P}^D} \bot$. The other component that is not mapped to $\bot$ can, from that point on, unfold $\mathcal{P}^D$ or $\mathcal{Q}^D$ in arbitrary manner it likes.

The other direction: It is obvious $[\mathcal{P}^D]_s \subseteq [\mathcal{P}^D \vee \mathcal{Q}^D]_s$ and $[\mathcal{Q}^D]_s \subseteq [\mathcal{P}^D \vee \mathcal{Q}^D]_s$. If $\mathcal{G} \in ([\mathcal{P}^D]_s \cup [\mathcal{Q}^D]_s)^+ \setminus ([\mathcal{P}^D]_s \cup [\mathcal{Q}^D]_s)$, then there exists $\mathcal{G}_0 \in [\mathcal{P}^D]_s$ and $\mathcal{G}_1 \in [\mathcal{Q}^D]_s$ s.t. $\mathcal{G} = \mathcal{G}_0 + \mathcal{G}_1$. Then $\mathcal{G} \in [\mathcal{P}^D \vee \mathcal{Q}^D]_s$. $\square$

Lemma 5. *For* $\wedge$*-composable TIOTSs* $\mathcal{P}$ *and* $\mathcal{Q}$*,* $[\mathcal{P} \wedge \mathcal{Q}]_s^+ = [\mathcal{P}]_s^+ \cap [\mathcal{Q}]_s^+$.

Proof. Due to the distributivity of state $\wedge$ operation over state $\vee$ operation, $(\mathcal{P} \wedge \mathcal{Q})^D$ and $\mathcal{P}^D \wedge \mathcal{Q}^D$ are isomorphic. The proof is reduced to $[\mathcal{P}^D \wedge \mathcal{Q}^D]_s = [\mathcal{P}^D]_s \cap [\mathcal{Q}^D]_s$.

$[\mathcal{P}^D \wedge \mathcal{Q}^D]_s \subseteq [\mathcal{P}^D]_s \cap [\mathcal{Q}^D]_s$: obvious since $[\mathcal{P}^D \wedge \mathcal{Q}^D]_s \subseteq [\mathcal{P}^D]_s$ and $[\mathcal{P}^D \wedge \mathcal{Q}^D]_s \subseteq [\mathcal{Q}^D]_s$.

$[\mathcal{P}^D]_s \cap [\mathcal{Q}^D]_s \subseteq [\mathcal{P}^D \wedge \mathcal{Q}^D]_s$: If $\mathcal{G} \in [\mathcal{P}^D]_s \cap [\mathcal{Q}^D]_s$, then there exists $\mathcal{G}_0 \in stg(\mathcal{P}^D)$ and $\mathcal{G}_1 \in stg(\mathcal{Q}^D)$ s.t. $\mathcal{G}_0 \preceq \mathcal{G}$ and $\mathcal{G}_1 \preceq \mathcal{G}$. Then $\mathcal{G}_0 \& \mathcal{G}_1 \preceq \mathcal{G}$ and $\mathcal{G} \in [\mathcal{P}^D \wedge \mathcal{Q}^D]_s$ since $\mathcal{G}_0 \& \mathcal{G}_1 = \mathcal{G}_0 \wedge \mathcal{G}_1 \in [\mathcal{P}^D \wedge \mathcal{Q}^D]_s$.

$\square$

Lemma 6. *For* $\%$*-composable TIOTSs* $\mathcal{P}$ *and* $\mathcal{Q}$*,* $[\mathcal{P}\%\mathcal{Q}]_s^+ = \{\mathcal{G}_{\mathcal{P}\%\mathcal{Q}} \mid \forall \mathcal{G}_{\mathcal{Q}} \in [\mathcal{Q}]_s^+, h \in H : \mathcal{G}_{\mathcal{P}\%\mathcal{Q}} \parallel_h \mathcal{G}_{\mathcal{Q}} \in [\mathcal{P}]_s^+\}$.

Proof. Since $\mathcal{P}\%\mathcal{Q}$ is defined to be $\mathcal{P}\%\mathcal{Q}^D$ and state $\%$ operation is left-distributive over state $\vee$ operation, $(\mathcal{P}\%\mathcal{Q})^D$ is isomorphic to $\mathcal{P}^D\%\mathcal{Q}^D$, which in turn is isomorphic to $(\mathcal{P}^\neg \parallel \mathcal{Q}^D)^\neg$. Thus $[\mathcal{P}\%\mathcal{Q}]_s^+ = [\mathcal{P}^D\%\mathcal{Q}^D]_s$ and we only need to prove $\mathcal{G} \in stg(\mathcal{P}^D\%\mathcal{Q}^D) \Rightarrow \mathcal{P}^D \sqsubseteq \mathcal{G} \parallel \mathcal{Q}^D$ and $\mathcal{P}^D \sqsubseteq \mathcal{G} \parallel \mathcal{Q}^D \Rightarrow \mathcal{G} \in [\mathcal{P}^D\%\mathcal{Q}^D]_s$.

$\mathcal{G} \in stg(\mathcal{P}^D\%\mathcal{Q}^D) \Rightarrow \mathcal{P}^D \sqsubseteq \mathcal{G} \parallel \mathcal{Q}^D$: If $\mathcal{G} \in stg(\mathcal{P}^D\%\mathcal{Q}^D)$, then $\mathcal{G} \parallel \mathcal{P}^\neg \parallel \mathcal{Q}^D$ is $\bot$-free, which implies $\mathcal{P}^D \sqsubseteq \mathcal{G} \parallel \mathcal{Q}^D$.

$\mathcal{P}^D \sqsubseteq \mathcal{G} \parallel \mathcal{Q}^D \Rightarrow \mathcal{G} \in [\mathcal{P}^D\%\mathcal{Q}^D]_s$: $\mathcal{P}^D \sqsubseteq \mathcal{G} \parallel \mathcal{Q}^D$ implies $\mathcal{P}^\neg \parallel \mathcal{G} \parallel \mathcal{Q}^D$ is $\bot$-free. Thus $\mathcal{G} \in [(\mathcal{P}^\neg \parallel \mathcal{Q}^D)^\neg]_s = [\mathcal{P}^D\%\mathcal{Q}^D]_s$.

$\square$

Theorem 2. $\simeq$ *is a congruence w.r.t.* $\parallel$*,* $\vee$*,* $\wedge$ *and* $\%$ *subject to composability.*

Proof. Use the lemma 3-6. $\square$

Theorem 3. $\mathcal{P}_0 \sqsubseteq \mathcal{P}_1$ *iff* $TR_1 \subseteq TR_0$ *and* $TE_1 \subseteq TE_0$*.*

Proof. Begin by supposing $\mathcal{P}_0 \sqsubseteq \mathcal{P}_1$. Let t be a smallest trace such that $t \in TR_1 \setminus TR_0$. Now construct a component $\mathcal{Q}$ containing the single inconsistent trace t. We see that t is an inconsistent trace in $\mathcal{P}_1 \| \mathcal{Q}$, while t is not a trace of $\mathcal{P}_0 \| \mathcal{Q}$. This contradicts $\mathcal{P}_0 \sqsubseteq \mathcal{P}_1$. Instead let t be a smallest trace such that $t \in TE_1 \setminus TE_0$. Construct a component $\mathcal{Q}$ containing the trace t leading to a plain state. Then t is an inconsistent trace of $\mathcal{P}_1 \| \mathcal{Q}$, while t is either not a trace or is a consistent trace of $\mathcal{P}_0 \| \mathcal{Q}$. Again, this contradicts $\mathcal{P}_0 \sqsubseteq \mathcal{P}_1$.

For the other direction, let $TR_1 \subseteq TR_0$ and $TE_1 \subseteq TE_0$. Now suppose for a contradiction that $\mathcal{P}_0 \not\sqsubseteq \mathcal{P}_1$. Then there is a component $\mathcal{Q}$ such that $\mathcal{P}_0 \| \mathcal{Q}$ is $\bot$-free, while $\mathcal{P}_1 \| \mathcal{Q}$ is not $\bot$-free. Then there exists a trace t in $\mathcal{P}_1 \| \mathcal{Q}$ leading to $\bot$, such that there is no trace in $\mathcal{P}_0 \| \mathcal{Q}$ leading to $\bot$. This means that either (i) t is an inconsistent trace of $\mathcal{Q}$, t is not a trace of $\mathcal{P}_0$ while t is a trace of $\mathcal{P}_1$; or (ii) t is a consistent trace of $\mathcal{Q}$, a consistent or non-existent trace of $\mathcal{P}_0$ and an inconsistent trace of $\mathcal{P}_1$. However, both of these cases contradict the opening assumption $TR_1 \subseteq TR_0$ and $TE_1 \subseteq TE_0$. Hence $\mathcal{P}_0 \sqsubseteq \mathcal{P}_1$. $\square$

Proposition 3. *If* $\mathcal{P}_0$ *and* $\mathcal{P}_1$ *are* $\|$*-composable, then* $\mathcal{TT}(\mathcal{P}_0 \parallel \mathcal{P}_1) = (I, O, TR, TE)$ *where* $I = (I_0 \cup I_1) \setminus O$*,* $O = O_0 \cup O_1$ *and the trace sets are given by:*

- $TE = \{tt \mid tt \upharpoonright tA_i \in TE_i \wedge tt \upharpoonright tA_{\bar{i}} \in TR_{\bar{i}}\} \cdot tA^*$
- $TR = TE \uplus \{tt \mid tt \upharpoonright tA_i \in (TR_i \setminus TE_i) \wedge tt \upharpoonright tA_{\bar{i}} \in (TR_{\bar{i}} \setminus TE_{\bar{i}})\}$

Proof. Suppose $t \in TE$. Then there is a prefix t' such that wlog $t' \upharpoonright \mathcal{A}_0 \in TE_0$, while $t' \upharpoonright \mathcal{A}_1 \in TR_1$. Therefore, $s_0^0 \overset{t' \upharpoonright \mathcal{A}_0}{\Longrightarrow} \perp$, while $s_1^0 \overset{t' \upharpoonright \mathcal{A}_1}{\Longrightarrow} s_1$, for s_1 being a plain state or $\perp$. By Definition 4 and Table 1 we derive $s_0^0 \| s_1^0 \overset{t'}{\Longrightarrow} \perp$, hence $s_0^0 \| s_1^0 \overset{t}{\Longrightarrow} \perp$ as required. Now suppose $t \in TR \setminus TE$. Then $t \upharpoonright \mathcal{A}_0 \in TR_0 \setminus TE_0$ and $t \upharpoonright \mathcal{A}_1 \in TR_1 \setminus TE_1$. Consequently, $s_0^0 \overset{t \upharpoonright \mathcal{A}_0}{\Longrightarrow} s_0$ and $s_1^0 \overset{t \upharpoonright \mathcal{A}_1}{\Longrightarrow} s_1$, where s_0 and s_1 are plain states. Therefore $s_0^0 \| s_1^0 \overset{t}{\Longrightarrow} s_0 \| s_1$, where $s_0 \| s_1$ is a plain state. The other direction is similar. □

Proposition 4. *If $\mathcal{P}_0$ and $\mathcal{P}_1$ are $\vee$-composable, then $\mathcal{TT}(\mathcal{P}_0 \vee \mathcal{P}_1) = (I, O, TR_0 \cup TR_1, TE_0 \cup TE_1)$, where $I = I_0 = I_1$ and $O = O_0 = O_1$.*

Proof. Let $t \in TE_0 \cup TE_1$. Then wlog $t \in TE_0$. Hence $s_0^0 \overset{t}{\Longrightarrow} \perp$. If there exists a prefix t' of t such that $s_1^0 \overset{t'}{\Longrightarrow} \perp$, then by Table 1, we see $s_0^0 \vee s_1^0 \overset{t'}{\Longrightarrow} \perp$, implying $s_0^0 \vee s_1^0 \overset{t}{\Longrightarrow} \perp$. Instead, if there exists a prefix t' of t such that $s_1^0 \overset{t'}{\Longrightarrow} \top$, then by Table 1 we see $s_0^0 \vee s_1^0 \overset{t'}{\Longrightarrow} s_0$ for some state s_0, such that if $t \equiv t't''$, then $s_0 \overset{t''}{\Longrightarrow} \perp$. Consequently, $s_0^0 \vee s_1^0 \overset{t}{\Longrightarrow} \perp$ as required. The case of $t \in (TR_0 \cup TR_1) \setminus (TE_0 \cup TE_1)$ is just as straightforward, as is the other direction. □

Proposition 5. *If $\mathcal{P}_0$ and $\mathcal{P}_1$ are $\wedge$-composable, then $\mathcal{TT}(\mathcal{P}_0 \wedge \mathcal{P}_1) = (I, O, TR_0 \cap TR_1, TE_0 \cap TE_1)$, where $I = I_0 = I_1$ and $O = O_0 = O_1$.*

Proof. Dual to disjunction. □

Proposition 6. *If $\mathcal{P}_0$ dominates $\mathcal{P}_1$, then $\mathcal{TT}(\mathcal{P}_0 \% \mathcal{P}_1) = (I, O, TR, TE)$, where $I = I_0 \cup O_1$, $O = O_0 \setminus O_1$, and the trace sets satisfy:*

- $TE = TE_0 \cup \{tt \mid tt \upharpoonright tA_1 \notin TR_1\} \cdot tA^*$
- $TR = TE \uplus \{tt \mid tt \in (TR_0 \setminus TE_0) \wedge tt \upharpoonright tA_1 \in (TR_1 \setminus TE_1)\}$.

Proof. Let $t \in TE$. If $t \in TE_0$, then by Definition 4 and Table 1 we see $s_0^0 \% s_1^0 \overset{t}{\Longrightarrow} \perp$ as required. Instead, if $t \upharpoonright tA_1 \notin TR_1$, then by Table 1 we see $s_0^0 \% s_1^0 \overset{t}{\Longrightarrow} \perp$. Instead suppose $t \in TR \setminus TE$. Then $t \in TR_0 \setminus TE_0$ and $t \in TR_1 \setminus TE_1$. Consequently, $s_0^0 \overset{t}{\Longrightarrow} s_0$ for some plain state s_0, and $s_1^0 \overset{t \upharpoonright tA_1}{\Longrightarrow} s_1$ for some plain state s_1. Thus $s_0^0 \% s_1^0 \overset{t}{\Longrightarrow} s_0 \% s_1$, where by Table 1 $s_0 \% s_1$ is a plain state.

Now suppose $s_0^0 \% s_1^0 \overset{t}{\Longrightarrow} \perp$. Then there is a shortest prefix t' of t such that t' is a trace to $\perp$. Then by Table 1, we see $s_0^0 \overset{t'}{\Longrightarrow} \perp$ or $s_1^0 \overset{t' \upharpoonright tA_1}{\Longrightarrow} \top$. In the case of the former, this means $t' \in TE_0$, while for the latter we see $t' \in TT_1$. Now as $\mathcal{P}_1$ is deterministic, it follows $t' \in TT_1 \setminus TR_1$. Thus $t' \in TE$, which means $t \in TE$ as required. Instead suppose $s_0^0 \% s_1^0 \overset{t}{\Longrightarrow} s_0 \% s_1$ and $s_0^0 \% s_1^0 \overset{t}{\not\Longrightarrow} \perp$. Then $s_0^0 \overset{t}{\Longrightarrow} s_0$ and $s_1^0 \overset{t \upharpoonright tA_1}{\Longrightarrow} s_1$, where $s_0 \neq \perp$ and $s_1 \neq \perp$. Hence $t \in TR_0 \setminus TE_0$ and $t \upharpoonright tA_1 \in TR_1 \setminus TE_1$, and so $t \in TR \setminus TE$ as required. □