Compositional Reasoning for Asynchronous Systems (DRAFT)

Nina Amla[1], E. Allen Emerson[2], Kedar Namjoshi[3], and Richard Trefler[4]

[1] Cadence Design Systems
[2] Department of Computer Sciences, University of Texas at Austin
[3] Bell Laboratories, Lucent Technologies
[4] AT&T Research Laboratories

Abstract. In this paper, we develop a compositional reasoning rule for asynchronous process composition, and show it to be both sound and complete.

1 Preliminaries

We assume an unbounded set of variable names, U. A *process* is given by a tuple (V, I, T, F), where V is a set of variables, partitioned into V_l, a non-empty set of "local" variables, and V_i is a set of "interface" variables; $I(V_l)$ is an initial condition; $T(U, U')$ is a transition relation where U' is a set of variable names that define the next state; and $F(V, V')$ is a fairness condition. The transition relation must satisfy the following condition (non-interference): for any variable $w \notin V$, $[T \Rightarrow w' = w]$. For the definition of closure processes, we require an additional restriction on internal nondeterminism and divergence, which we define later.

The (interleaving) *composition* of processes P and Q, written as $P \,[]\, Q$, is defined provided that, for each process, its local variables are disjoint from the variables of the other (i.e., $V_l(P) \cap V(Q) = \emptyset$ and $V_l(Q) \cap V(P) = \emptyset$). It has local variables $V_l(P) \cup V_l(Q)$, interface variables $V_i(P) \cup V_i(Q)$, initial condition $I(P) \wedge I(Q)$, transition relation $T(P) \vee T(Q)$, and fairness condition $F(P) \wedge F(Q)$. The transition relation, as defined, satisfies the non-interference condition.

The set of infinite executions of process P, $L_\infty(P)$, is given by the set of infinite sequences satisfying the formula[1] $I(P) \wedge \mathsf{G}(T(P) \vee (V_l'(P) = V_l(P))) \wedge F(P)$. The set of finite executions, $L_*(P)$, is given by the set of finite sequences satisfying $I(P) \wedge \mathsf{G}(T(P) \vee V_l'(P) = V_l(P))$. The *language* of P, denoted by $L(P)$ is $L_\infty(P) \cup L_*(P)$. The formula for $L_\infty(P)$ expresses the constraints that the first state is an initial one, every transition is either that of P or an "environment" transition, which keeps the local variables of P unchanged, and the fairness condition holds. The use of environment moves to define the language is similar to the definition in [AL95]. Not only is it the appropriate choice for "open" programs, but it also has the consequence that composition can be viewed as

[1] The notation $W' = W$, for a set of variables W, means $(\forall w : w \in W : w' = w)$.

the conjunction of languages. The external language of process P, $L_{ext}(P)$, is the projection of its language on the interface variables, i.e., $[L_{ext}(P) \equiv (\exists V_l(P) : L(P))]$. Here, the $\exists$ quantifier is stuttering-insensitive [AL95]: a sequence σ over variables V is in $(\exists x : L)$ iff there is a sequence δ over $V \cup \{x\}$ such that σ and δ are stuttering equivalent over V and δ satisfies L. If L is itself stuttering-insensitive, then so is $(\exists x : L)$. Note that the $L(P)$ is stuttering insensitive so, therefore, is $L_{ext}(P)$.

We also need the notion of *safety closure*. The safety closure of a language is the strongest safety property (in terms of [AS85]) that includes the language. For a language L of sequences over variables V, the safety closure of L, denoted by $cl(L)$, is defined as the set of sequences over V where $x \in cl(L)$ iff for all $j < |x|$ there exists y such that $x[0..j];y$ belongs to L.

Proposition 0. For processes P and Q such that $P \,[]\, Q$ is defined, $T(P \,[]\, Q)$ satisfies the non-interference condition.

Proof. For any w not in $V(P \,[]\, Q)$:

$$
\begin{array}{ll}
 & T(P \,[]\, Q) \\
\equiv & \quad (\text{ by definition }) \\
 & T(P) \lor T(Q) \\
\Rightarrow & \quad (\ w \notin V(P), w \notin V(Q), \text{ non-interference for } P, Q \ ) \\
 & w' = w \lor w' = w \\
\equiv & \quad (\text{ propositional logic }) \\
 & w' = w
\end{array}
$$

□

Theorem 0. For processes P and Q such that $P \,[]\, Q$ is defined, $[L_\infty(P \,[]\, Q) \equiv L_\infty(P) \land L_\infty(Q)]$

Proof.

$$
\begin{array}{ll}
 & L_\infty(P \,[]\, Q) \\
\equiv & \quad (\text{ by definition of } L \text{ and definition of } P \,[]\, Q \ ) \\
 & I(P) \land I(Q) \land F(P) \land F(Q) \land \\
 & \mathsf{G}((T(P) \lor T(Q)) \lor (V_l'(P) = V_l(P) \land V_l'(Q) = V_l(Q))) \\
\equiv & \quad (\text{ distributing } \lor \text{ over } \land \ ) \\
 & I(P) \land I(Q) \land F(P) \land F(Q) \land \\
 & \mathsf{G}((T(P) \lor T(Q) \lor V_l'(P) = V_l(P)) \land (T(P) \lor T(Q) \lor V_l'(Q) = V_l(Q))) \\
\equiv & \quad (\text{ distributing } \mathsf{G} \text{ over } \land \ ) \\
 & I(P) \land I(Q) \land F(P) \land F(Q) \land \\
 & \mathsf{G}(T(P) \lor T(Q) \lor V_l'(P) = V_l(P)) \land \mathsf{G}(T(P) \lor T(Q) \lor V_l'(Q) = V_l(Q)) \\
\equiv & \quad (\text{ by definition of } P \,[]\, Q \text{ and non-interference for } P,) \\
 & \quad (\ [T(P) \Rightarrow V_l'(Q) = V_l(Q)] \text{ and symmetrically for } Q\text{; simplifying }) \\
 & I(P) \land I(Q) \land F(P) \land F(Q) \land \\
 & \mathsf{G}(T(P) \lor V_l'(P) = V_l(P)) \land \mathsf{G}(T(Q) \lor V_l'(Q) = V_l(Q)) \\
\equiv & \quad (\text{ definition of } L_\infty \ ) \\
 & L_\infty(P) \land L_\infty(Q)
\end{array}
$$

□

Theorem 1. For processes P and Q such that $P \,[]\, Q$ is defined, $[L_*(P \,[]\, Q) \equiv L_*(P) \wedge L_*(Q)]$

Proof Sketch. The proof of this theorem is obtained from the previous one by leaving out the fairness terms. □

Theorem 2. For processes P and Q such that $P \,[]\, Q$ is defined, $[L(P \,[]\, Q) \equiv L(P) \wedge L(Q)]$

Proof. Let the proposition *finite* be $\mathsf{FX}_w(\mathit{false})$ (this formula is true of a sequence if eventually there is a state without a successor). Then, we can represent $L(P)$ by $[L(P) \equiv (\neg \mathit{finite} \Rightarrow L_\infty(P)) \wedge (\mathit{finite} \Rightarrow L_*(P))]$. Therefore,

$$
\begin{array}{ll}
 & L(P \,[]\, Q) \\
\equiv & \quad (\text{ by definition }) \\
 & (\neg \mathit{finite} \Rightarrow L_\infty(P \,[]\, Q)) \wedge (\mathit{finite} \Rightarrow L_*(P \,[]\, Q)) \\
\equiv & \quad (\text{ by previous theorems }) \\
 & (\neg \mathit{finite} \Rightarrow (L_\infty(P) \wedge L_\infty(Q))) \wedge \\
 & (\mathit{finite} \Rightarrow (L_*(P) \wedge L_*(Q))) \\
\equiv & \quad (\text{ distributing } \Rightarrow \text{ over } \wedge \text{ and re-arranging }) \\
 & ((\neg \mathit{finite} \Rightarrow L_\infty(P)) \wedge (\mathit{finite} \Rightarrow L_*(P))) \wedge \\
 & ((\neg \mathit{finite} \Rightarrow L_\infty(Q)) \wedge (\mathit{finite} \Rightarrow L_*(Q))) \\
\equiv & \quad (\text{ by definition }) \\
 & L(P) \wedge L(Q)
\end{array}
$$

□

Theorem 3. For processes P and Q such that $P \,[]\, Q$ is defined, $[L_{ext}(P \,[]\, Q) \equiv L_{ext}(P) \wedge L_{ext}(Q)]$

Proof Sketch. This proof parallels the proof in [AL95]. The difficult step is the direction from right to left. In this direction, a sequence σ satisfying $L_{ext}(P) \wedge L_{ext}(Q)$ may do so with a transition that, adding values for local variables of P and Q, satisfies the transition conditions for both P and Q, and does so by modifying both sets of local variables. However, to satisfy both conditions, it must keep all interface variables constant. Thus, this transition can be split into two: the first modifying variables only of P, and the second only of Q. Doing so enables the resulting sequence to satisfy $L(P \,[]\, Q)$. □

We write $P \models Q$, where P and Q are processes, provided $V_i(Q) \subseteq V_i(P)$, to mean that $[L_{ext}(P) \Rightarrow L_{ext}(Q)]$. We need two other operators on processes: *choice*, represented by $+$, with the property that $[L_{ext}(P + Q) \equiv L_{ext}(P) \vee L_{ext}(Q)]$, and *closure*, represented by CL, with the property that $[L_{ext}(CL(P)) \equiv cl(L_{ext}(P))]$. The safety closure of a language is the strongest safety property (in the sense of [AS85]) that contains the language.

For processes P, Q, define the process $P + Q$ (read "P choice Q") under the same condition that $P \,[]\, Q$ is defined, for a fresh variable c, with local variables $V_l(P) \cup V_l(Q) \cup c$, interface variables $V_i(P) \cup V_i(Q)$, initial condition $(c \wedge I(P)) \vee (\neg c \wedge I(Q))$, transition relation $(c' = c) \wedge ((c \wedge T(P)) \vee (\neg c \wedge T(Q)))$, fairness condition $(\overset{\infty}{\mathsf{G}}(c) \wedge F(P)) \vee (\overset{\infty}{\mathsf{G}}(\neg c) \wedge F(Q))$. The variable c serves to choose between processes P and Q initially. This intuition is made precise by the following theorems.

Lemma 0. For processes P, Q such that $P+Q$ is defined, $[L(P+Q) \equiv L(P) \wedge G(c) \wedge G(Vl'(Q) = V_l(Q)) \vee L(Q) \wedge G(\neg c) \wedge G(Vl'(P) = V_l(P))]L_{ext}(P) \vee L_{ext}(Q)]$

Proof.

First, we compute $L_\infty(P+Q)$.

$$
\begin{array}{ll}
 & L_\infty(P+Q) \\
\equiv & \quad (\text{ by definition }) \\
 & ((c \wedge I(P)) \vee (\neg c \wedge I(Q))) \wedge \\
 & \mathsf{G}((c' = c \wedge ((c \wedge T(P)) \vee (\neg c \wedge T(Q)))) \vee (V_l'(P) = V_l(P) \wedge V_l'(Q) = V_l(Q) \wedge c' = c)) \wedge \\
 & ((\overset{\infty}{\mathsf{G}}(c) \wedge F(P)) \vee (\overset{\infty}{\mathsf{G}}(\neg c) \wedge F(Q))) \\
\equiv & \quad (\text{ distributing } \wedge \text{ over the } \vee \text{ in the initial condition, }) \\
 & \quad (\text{ and factoring } c' = c \text{ as common term and distributing } \mathsf{G} \text{ over } \wedge \text{ }) \\
 & c \wedge I(P) \wedge \mathsf{G}(c' = c) \wedge \\
 & \mathsf{G}((c \wedge T(P)) \vee (\neg c \wedge T(Q)) \vee (V_l'(P) = V_l(P) \wedge V_l'(Q) = V_l(Q))) \wedge \\
 & ((\overset{\infty}{\mathsf{G}}(c) \wedge F(P)) \vee (\overset{\infty}{\mathsf{G}}(\neg c) \wedge F(Q))) \\
 & \vee \\
 & \neg c \wedge I(Q) \wedge \mathsf{G}(c' = c) \wedge \\
 & \mathsf{G}((c \wedge T(P)) \vee (\neg c \wedge T(Q)) \vee (V_l'(P) = V_l(P) \wedge V_l'(Q) = V_l(Q))) \wedge \\
 & ((\overset{\infty}{\mathsf{G}}(c) \wedge F(P)) \vee (\overset{\infty}{\mathsf{G}}(\neg c) \wedge F(Q))) \\
\equiv & \quad (\text{ } [c \wedge \mathsf{G}(c' = c) \equiv \mathsf{G}(c)], \text{ similarly for other case }) \\
 & c \wedge I(P) \wedge \mathsf{G}(c) \wedge \\
 & \mathsf{G}((c \wedge T(P)) \vee (\neg c \wedge T(Q)) \vee (V_l'(P) = V_l(P) \wedge V_l'(Q) = V_l(Q))) \wedge \\
 & ((\overset{\infty}{\mathsf{G}}(c) \wedge F(P)) \vee (\overset{\infty}{\mathsf{G}}(\neg c) \wedge F(Q))) \\
 & \vee \\
 & \neg c \wedge I(Q) \wedge \mathsf{G}(\neg c) \wedge \\
 & \mathsf{G}((c \wedge T(P)) \vee (\neg c \wedge T(Q)) \vee (V_l'(P) = V_l(P) \wedge V_l'(Q) = V_l(Q))) \wedge \\
 & ((\overset{\infty}{\mathsf{G}}(c) \wedge F(P)) \vee (\overset{\infty}{\mathsf{G}}(\neg c) \wedge F(Q))) \\
\equiv & \quad (\text{ using } \mathsf{G}(c) \text{ and } \mathsf{G}(\neg c) \text{ to simplify other terms }) \\
 & I(P) \wedge \mathsf{G}(c) \wedge \mathsf{G}(T(P) \vee (V_l'(P) = V_l(P) \wedge V_l'(Q) = V_l(Q))) \wedge F(P) \\
 & \vee \\
 & I(Q) \wedge \mathsf{G}(\neg c) \wedge \mathsf{G}(T(Q) \vee (V_l'(P) = V_l(P) \wedge V_l'(Q) = V_l(Q))) \wedge F(Q) \\
\equiv & \quad (\text{ distributing } \vee \text{ over } \wedge \text{ in the } \mathsf{G}()\text{'s followed by distributing } \mathsf{G} \text{ over } \wedge \text{ }) \\
 & I(P) \wedge \mathsf{G}(c) \wedge \mathsf{G}(T(P) \vee V_l'(P) = V_l(P)) \wedge \mathsf{G}(T(P) \vee V_l'(Q) = V_l(Q)) \wedge F(P) \\
 & \vee \\
 & I(Q) \wedge \mathsf{G}(\neg c) \wedge \mathsf{G}(T(Q) \vee V_l'(P) = V_l(P)) \wedge \mathsf{G}(T(Q) \vee V_l'(Q) = V_l(Q)) \wedge F(Q) \\
\equiv & \quad (\text{ using non-interference property for } T(P) \text{ and } T(Q) \text{ }) \\
 & I(P) \wedge \mathsf{G}(c) \wedge \mathsf{G}(T(P) \vee V_l'(P) = V_l(P)) \wedge \mathsf{G}(V_l'(Q) = V_l(Q)) \wedge F(P) \\
 & \vee \\
 & I(Q) \wedge \mathsf{G}(\neg c) \wedge \mathsf{G}(V_l'(P) = V_l(P)) \wedge \mathsf{G}(T(Q) \vee V_l'(Q) = V_l(Q)) \wedge F(Q) \\
\equiv & \quad (\text{ definition of } L_\infty \text{ }) \\
 & L_\infty(P) \wedge \mathsf{G}(c) \wedge \mathsf{G}(V_l'(Q) = V_l(Q)) \\
 & \vee \\
 & L_\infty(Q) \wedge \mathsf{G}(\neg c) \wedge \mathsf{G}(V_l'(P) = V_l(P))
\end{array}
$$

□

Eliminating the fairness conditions from the previous calculation shows that $[L_*(P+Q) = (L_*(P) \wedge \mathsf{G}(c) \wedge \mathsf{G}(V_l'(Q) = V_l(Q))) \vee L_*(Q) \wedge \mathsf{G}(\neg c) \wedge \mathsf{G}(V_l'(P) = V_l(P))]$

Therefore,

$$
\begin{array}{ll}
 & L(P+Q) \\
\equiv & \quad (\text{ by definition }) \\
 & (\mathit{finite} \wedge L_\infty(P+Q)) \vee (\mathit{finite} \wedge L_*(P+Q)) \\
\equiv & \quad (\text{ by above calculations and re-arranging }) \\
 & (L(P) \wedge \mathsf{G}(c) \wedge \mathsf{G}(V_l'(Q) = V_l(Q))) \vee (L(Q) \wedge \mathsf{G}(\neg c) \wedge \mathsf{G}(V_l'(P) = V_l(P)))
\end{array}
$$

□

Theorem 4. For processes P, Q such that $P+Q$ is defined, $[L_{ext}(P+Q) \equiv L_{ext}(P) \vee L_{ext}(Q)]$

Proof.

$$
\begin{array}{ll}
 & L_{ext}(P+Q) \\
\equiv & \quad (\text{ definition of } L_{ext} \text{ and previous calculation }) \\
 & (\exists V_l(P), V_l(Q), c : (L(P) \wedge \mathsf{G}(c) \wedge \mathsf{G}(V_l'(Q) = V_l(Q))) \vee \\
 & (L(Q) \wedge \mathsf{G}(\neg c) \wedge \mathsf{G}(V_l'(P) = V_l(P)))) \\
\equiv & \quad (\text{ distributing } \exists \text{ over } \vee \text{ }) \\
 & (\exists V_l(P), V_l(Q), c : L(P) \wedge \mathsf{G}(c) \wedge \mathsf{G}(V_l'(Q) = V_l(Q))) \\
 & \vee \\
 & (\exists V_l(P), V_l(Q), c : L(Q) \wedge \mathsf{G}(\neg c) \wedge \mathsf{G}(V_l'(P) = V_l(P))) \\
\equiv & \quad (\text{ } c \text{ is a fresh variable; definition of } P+Q \text{ }) \\
 & (\exists V_l(P) : L(P)) \vee (\exists V_l(Q) : L(Q)) \\
\equiv & \quad (\text{ definition of } L_{ext} \text{ }) \\
 & L_{ext}(P) \vee L_{ext}(Q)
\end{array}
$$

□

1.1 Closure Process

Let $P = (V, I, T, F)$. Define $CL(P) = (V, I, T, \mathit{true})$; i.e., the fairness condition has been replaced with *true*.

Lemma 1. $[L(CL(P)) \equiv cl(L(P))]$.

Proof. Suppose $x \in L(CL(P))$. Then for all $j < |x|$, $x_0 ... x_j$ belongs to $L_*(P)$; hence, $x \in cl(P)$.

Suppose $x \in cl(L(P))$. If x is finite, then $x \in L(CL(P))$ by definition. If x is infinite, then for all $j \in \mathbf{N}$ there exists y such that $x_0 ... xj; y$ is in $L(P)$. Thus, for every j, $T(x_j, x_{j+1})$ is true; hence x is in $CL(P)$. □

For the next lemma on the external language of a closure process, we need the following reasonable restrictions on a process' internal choices. A process P must satisfy (a) *finite internal nondeterminism*: for any x that is an initial state, there are only finitely many initial states that differ from x only in the values of local variables, and for any x, there is only a finite number of successors

$y_0, y_1, \ldots$ such that and the y_i's differ only in the values of local variables, and (b) *no internal divergence*: for any x, there is no infinite computation σ starting at x, consisting only of transitions by P, where the states of σ differ from x only in the values of local variables.

Theorem 5. For any process P, $[L_{ext}(CL(P)) \equiv cl(L_{ext}(P))]$.

Proof.

Suppose $x \in L_{ext}(CL(P))$. By definition of L_{ext}, there exists y that is stuttering equivalent to x relative to $V_i(P)$, and $y \in L(CL(P))$. By the previous lemma, $y \in cl(L(P))$. So, for all j, there exists z such that $y_0 ... y_j; z \in L(P)$. Now choose any prefix $x[0..j]$. Let $y[0..k]$ be the matching prefix in y, and let z be the extension so that $y[0..k]; z \in L(P)$. Let z' be the projection of z on to $V_i(P)$. Then $x[0..j]; z'$ is in $L_{ext}(P)$. Therefore, x is in $cl(L_{ext}(P))$.

In the other direction, suppose $x \in cl(L_{ext}(P))$ and x is finite. Therefore, $x \in L_{ext}(P)$ and, therefore, in $L_{ext}(CL(P))$, since they have the same finite traces. Now suppose x is infinite. As $x \in cl(L_{ext}(P))$, for every j, there exists z_j such that $x[0..j]; z_j \in L_{ext}(P)$. By definition of L_{ext}, there is $y_j \in L(P)$, where y_j is stuttering equivalent to $x[0..j]; z_j$ for variables in $V_i(P)$. Let w_j be the smallest prefix of y_j that matches $x[0..j]$, with stuttering transitions removed. Thus, in w_j, any transition that keeps the external variables unchanged must change the local variables and, thus, be a transition of P. Create a tree t from the finite paths w_j by merging common prefixes. If x has the form $\alpha; a^\omega$, for some α, a, then by non-divergence, infinitely many of the $a; a$ moves must be environment moves; in which case we relabel them temporarily to $a; a'$, where the prime indicates that it is an environment move. The tree t then has infinitely many nodes. It must have finite branching at each node, since the successors of a node labeled with (x_k, l) have the form (x_k, l'), or (x_{k+1}, l'), for various l', and processes have finite internal non-determinism. For the same reason, the tree has finitely many root nodes. Thus, by König's lemma, there must be an infinite branch, call it σ, in the tree. Since each transition in the tree belongs either to P or the environment, σ is in $L(CL(P))$. By the non-divergence condition, this infinite branch must change in its visible (external) state component infinitely often. Since the tree was constructed from sequences matching x in its visible component, the branch σ is stuttering equivalent to x relative to $V_i(P)$. Hence, x is in $L_{ext}(CL(P))$. $\square$

1.2 Compositional Reasoning

We can now present the rule for compositional reasoning. This rule is similar in form to the rule for synchronous composition presented in [AENT01], and has a similar justification. To show that $P_1 \,[]\, P_2 \models T$, for compatible processes P, Q, T, one has to find processes Q_1, Q_2 such that

1. $P_1 \,[]\, Q_2, P_2 \,[]\, Q_1, Q_1 \,[]\, Q_2$ are all defined, and $V_i(Q_j) \subseteq V_i(P_j)$,
2. $P_1 \,[]\, Q_2 \models Q_1$, $Q_1 \,[]\, P_2 \models Q_2$,
3. $Q_1 \,[]\, Q_2 \models T$,

4. Either $P_1 \,[]\, CL(T) \models (T + Q_1 + Q_2)$, or $P_2 \,[]\, CL(T) \models (T + Q_1 + Q_2)$

The auxiliary processes Q_1 and Q_2 act as mutual assumptions and guarantees in the so-called, "circular" (really, just proof by mutual induction), second step. The third step ensures that, together, they satisfy T. The fourth step guards against well-known errors (see [NT00,AENT01]) in applying circular reasoning in the presence of fairness.

Lemma 2. Given the assumptions of the rule, it follows that $[L_*(P_1 \,[]\, P_2) \Rightarrow (\exists W : L_*(Q_1 \,[]\, Q_2))]$, where W is the set of local variables of $Q_1 \,[]\, Q_2$.

Proof. We proceed by induction on the length of the string x in $L_*(P_1 \,[]\, P_2)$.

Suppose $|x| = 1$. Since $x \models I(P_1 \,[]\, P_2)$, and therefore $x \models I(P_1)(V_1 l)$. Since $P_1 \,[]\, Q_2$ is defined, $x \in I(P_1 \,[]\, Q_2)$ so, by the third hypothesis, $x \in L_{ext}(Q_1 \,[]\, Q_2)$. Therefore, $x \in (\exists W : L_*(Q_1 \,[]\, Q_2))$.

Suppose $|x| = n + 1, n > 0$. Then, as $x_0 ... x_{n-1} \in L_*(P_1 \,[]\, P_2)$, by induction hypothesis, $x_0 ... x_{n-1} \in (\exists W : L_*(Q_1 \,[]\, Q_2))$. So there exists a sequence $y_0 ... y_{k_n - 1}$ in $L_*(Q_1 \,[]\, Q_2)$ that agrees with $x_0 ... x_{n-1}$ on all variables outside of $V_l(Q_1) \cup V_l(Q_2)$. There are three possible cases for the transition from x_{n-1} to x_n. If it is an environment move, the local variables of P_1 and P_2 remain unchanged. Pick y_{k_n} so that it agrees with x_n and with $y_{k_n - 1}$ on the local variables of Q_1, Q_2. The joint transition becomes an environment move for $Q_1 \,[]\, Q_2$; thus, $x \in (\exists W : L_*(Q_1 \,[]\, Q_2))$. If it is a move by P_1, then $T(P_1)(x_{n-1}, x_n)$ holds. Pick y_{k_n} so that it agrees with x_n, and with $y_{k_n - 1}$ on the local variables of Q_2. Then $y_0, ..., y_{k_n}$ belongs to $L(P_1 \,[]\, Q_2)$. By the second hypothesis, it belongs to $L_{ext}(Q_1 \,[]\, Q_2)$, hence, x satisfies $(\exists W : L_*(Q_1 \,[]\, Q_2))$. The third case, where the transition satisfies $T(P_2)$, is symmetric to the second one. $\square$

Theorem 6. (Soundness) Given the assumptions of the rule, it follows that $P_1 \,[]\, P_2 \models T$.

Proof.

The conclusion can be written as $[L(P_1 \,[]\, P_2) \Rightarrow L_{ext}(T)]$. We divide this into safety and liveness proofs.

Safety: $[L(P_1 \,[]\, P_2) \Rightarrow cl(L_{ext}(T))]$

First, note that $[cl(L1) \Rightarrow cl(L2)]$ holds iff $[L1 \Rightarrow cl(L2)]$. From left-to-right, this is because $[L1 \Rightarrow cl(L1)]$, by definition. The other direction holds because cl is monotonic and idempotent. Therefore,

$$
\begin{array}{ll}
 & [L(P_1 \,[]\, P_2) \Rightarrow cl(L_{ext}(T))] \\
\equiv & \quad \text{(by above argument)} \\
 & [cl(L(P_1 \,[]\, P_2)) \Rightarrow cl(L_{ext}(T))] \\
\equiv & \quad \text{(as } [cl(L(P)) \equiv cl(L_*(P))] \text{ for any P)} \\
 & [cl(L_*(P_1 \,[]\, P_2)) \Rightarrow cl(L_{ext}(T))] \\
\equiv & \quad \text{(by above argument)} \\
 & [L_*(P_1 \,[]\, P_2) \Rightarrow cl(L_{ext}(T))]
\end{array}
$$

$\square$

We will work with this final form. So,

$$
\begin{array}{ll}
& L_*(P_1 \,[]\, P_2) \\
\Rightarrow & \quad (\text{ by Lemma 2 }) \\
& (\exists W : L_*(Q_1 \,[]\, Q_2)) \\
\Rightarrow & \quad (\text{ as } [L_*(P) \Rightarrow cl(L(P))] \text{ for any } P \;) \\
& (\exists W : cl(L(Q_1 \,[]\, Q_2))) \\
\Rightarrow & \quad (\text{ by hypothesis 4, monotonicity of } cl \;) \\
& (\exists W : cl(L_{ext}(T))) \\
\Rightarrow & \quad (\; W \text{ does not contain any variables of } T \;) \\
& cl(L_{ext}(T))
\end{array}
$$

$\square$

Liveness: $[L(P_1 \,[]\, P_2) \wedge cl(L_{ext}(T)) \Rightarrow L_{ext}(T)]$

$$
\begin{array}{ll}
& L(P_1 \,[]\, P_2) \wedge cl(L_{ext}(T)) \\
\equiv & \quad (\text{ by Theorem 5 }) \\
& L(P_1 \,[]\, P_2) \wedge L_{ext}(CL(T)) \\
\Rightarrow & \quad (\text{ by Theorem 2 }) \\
& L(P_1) \wedge L(P_2) \wedge L_{ext}(CL(T)) \\
\Rightarrow & \quad (\text{ by definition, } [L(P) \Rightarrow L_{ext}(P)] \;) \\
& L_{ext}(P_1) \wedge L_{ext}(P_2) \wedge L_{ext}(CL(T)) \\
\Rightarrow & \quad (\text{ by Hypothesis 5 }) \\
& L_{ext}(P_1) \wedge L_{ext}(P_2) \wedge L_{ext}(T + Q_1 + Q_2) \\
\Rightarrow & \quad (\text{ by Theorem 4 }) \\
& L_{ext}(P_1) \wedge L_{ext}(P_2) \wedge (L_{ext}(T) \vee L_{ext}(Q_1) \vee L_{ext}(Q_2)) \\
\Rightarrow & \quad (\text{ distributing } \wedge \text{ over } \vee\text{; Theorem 3; and weakening }) \\
& L_{ext}(T) \vee L_{ext}(P_2 \,[]\, Q_1) \vee L_{ext}(P_1 \,[]\, Q_2) \\
\Rightarrow & \quad (\text{ applying Hypothesis 3 and Theorem 3 }) \\
& L_{ext}(T) \vee L_{ext}(Q_1 \,[]\, Q_2) \vee L_{ext}(Q_1 \,[]\, Q_2) \\
\Rightarrow & \quad (\text{ by Hypothesis 4 }) \\
& L_{ext}(T)
\end{array}
$$

$\square$

Theorem 7. (Completeness) For processes P_1, P_2, and T, if $P_1 \,[]\, P_2 \models T$, there exist Q_1, Q_2 satisfying the conditions of the proof rule.
Proof. Take $Q_1 = P_1, Q_2 = P_2$. Then, the first and third hypotheses follow trivially, and the second and fourth hypothesis follow by weakening. $\square$

References

[AENT01] N. Amla, E.A. Emerson, K.S. Namjoshi, and R. Trefler. Assume-guarantee based compositional reasoning for synchronous timing diagrams. In *TACAS*, 2001.

[AL95] M. Abadi and L. Lamport. Conjoining specifications. *ACM Trans. on Programming Languages and Systems (TOPLAS)*, May 1995.

[AS85] B. Alpern and F. Schneider. Defining liveness. *Information Processing Letters*, 21(4), 1985.

[NT00] K.S. Namjoshi and R.J. Trefler. On the completeness of compositional reasoning. In *CAV*, volume 1855 of *LNCS*. Springer-Verlag, 2000.