THIS DOCUMENT IS THE ONLINE-ONLY APPENDIX TO:

Synthesis of Concurrent Systems with Many Similar Processes

PAUL C. ATTIE
Florida International University
and
E. ALLEN EMERSON
The University of Texas at Austin

ACM Transactions on Programming Languages and Systems, Vol. 20, No. 1, January 1998, Pages 51–115.

C. PROOFS

C.1 Process Similarity

The *process index substitution operator* $\theta = \{j_1/i_1, \ldots, j_m/i_m\}$ denotes the simultaneous replacement of process indices $i_1, \ldots, i_m$ by process indices $j_1, \ldots, j_m$ respectively. We require that $i_1, \ldots, i_m$ be pairwise distinct, and $j_1, \ldots, j_m$ be pairwise distinct. θ can be applied to all of the pair syntactic constructs defined in the article, as well as any pair model (e.g., M_{ij}). We define θ in a bottom-up manner as follows.

Definition C.1.1 (*Process Index Substitution Operator*). The process index substitution operator $\theta = \{j_1/i_1 \ldots j_m/i_m\}$ is defined as follows:

(1) θ distributes through propositional logic connectives, $=$, $:=$, $\rightarrow$, $//$, and $\cup$.

(2) For any process index i:

$$\begin{aligned} i\theta &= j_k \text{ if } i = i_k \text{ for some } k \in [1:m] \\ i\theta &= i \text{ otherwise.} \end{aligned}$$

(3) For any $Q_i \in \mathcal{AP}_i$:

$$Q_i\theta \;=\; Q_{i\theta}$$

(4) For any $x_{ij} \in \mathcal{SH}_{ij}$:

$$\begin{aligned} x_{ij}\theta &= x_{i\theta,j\theta} \\ x^i_{ij}\theta &= x^{i\theta}_{i\theta,j\theta} \\ x^j_{ij}\theta &= x^{j\theta}_{i\theta,j\theta} \end{aligned}$$

(5) For any i-state s_i:

$$s_i\theta \;=\; \{< Q_{i\theta}, s_i(Q_i) > \;\mid Q_i \in \mathcal{AP}_i\}$$

i.e., when $i = i_k$, $s_i\theta$ is a j_k-state. It satisfies the atomic propositions in $\mathcal{AP}_{j_k}$, which correspond to the atomic propositions in $\mathcal{AP}_i$ that s_i satisfies (remember that the sets of atomic proposition form a uniform family, see Section 3 on MPCTL*). Note that $s_i\theta = s_i$ if $i \neq i_k$ for all $k \in [1:m]$.

(6) For any pair-process P_i^j:

$$P_i^j\theta \;=\; \{(s_i\theta, B_i^j\theta \rightarrow A_i^j\theta, t_i\theta) \mid (s_i, B_i^j \rightarrow A_i^j, t_i) \in P_i^j\}$$

(7) For any ij-state s_{ij}:

$$\begin{aligned} s_{ij}\theta \;=\; & \{< Q_{i\theta}, s_{ij}(Q_i) > \;\mid Q_i \in \mathcal{AP}_i\} \cup \\ & \{< Q_{j\theta}, s_{ij}(Q_j) > \;\mid Q_j \in \mathcal{AP}_j\} \cup \\ & \{< x_{ij}\theta, h_{ij}\theta > \;\mid s_{ij}(x_{ij}) = h_{ij}, x_{ij} \in \mathcal{SH}_{ij}\} \end{aligned}$$

(8) For transition relation R_{ij}:

$$R_{ij}\theta = \{(s_{ij}\theta, h_{ij}\theta, t_{ij}\theta) \mid (s_{ij}, h_{ij}, t_{ij}) \in R_{ij}\}$$

(9) For the sets of initial ij-states S_{ij}^0 and all ij-states S_{ij}:

$$\begin{aligned} S_{ij}^0\theta &= \{s_{ij}\theta \mid s_{ij} \in S_{ij}^0\} \\ S_{ij}\theta &= \{s_{ij}\theta \mid s_{ij} \in S_{ij}\} \end{aligned}$$

(10) For the pair-structure $M_{ij} = (S_{ij}^0, S_{ij}, R_{ij})$:

$$M_{ij}\theta = (S_{ij}^0\theta, S_{ij}\theta, R_{ij}\theta)$$

PROPOSITION 6.2.1 (PAIR-STRUCTURE SIMILARITY). *Let i, j, i', j' be arbitrary elements of $\{i_1, \ldots, i_K\}$ such that $i\,I\,j$ and $i'\,I\,j'$. Then we have*

$$M_{ij} = M_{i'j'}\{i/i', j/j'\}.$$

PROOF. From the pair-structure definition (5.2.1), we have $M_{i'j'}\{i/i', j/j'\} = (S_{i'j'}^0, S_{i'j'}, R_{i'j'})\{i/i', j/j'\}$. By the process index substitution operator definition (C.1.1), this is equal to $(S_{i'j'}^0\{i/i', j/j'\}, S_{i'j'}\{i/i', j/j'\}, R_{i'j'}\{i/i', j/j'\})$. Now $S_{i'j'}^0\{i/i', j/j'\} = S_{ij}^0$ by the initial-state assumption. $S_{i'j'}$ is the set of all possible $i'j'$-states. Now $\mathcal{AP}_{i'}, \mathcal{AP}_{j'}$ are similar to $\mathcal{AP}_i, \mathcal{AP}_j$ by assumption (see Section 3.4), and since $P_i^j = P_{i'}^{j'}\{i/i', j/j'\}$, we must have $\mathcal{SH}_{ij} = \mathcal{SH}_{i'j'}\{i/i', j/j'\}$, since $\mathcal{SH}_{ij}$ is merely the set of shared variables the occur in P_i^j, P_j^i. Thus, $S_{ij} = S_{i'j'}\{i/i', j/j'\}$, since their domains are related by $\{i/i', j/j'\}$. Finally, by the process similarity assumption and the pair structure definition (5.2.1), we can infer $R_{ij} = R_{i'j'}\{i/i', j/j'\}$, since similar arcs in $P_i^j, P_{i'}^{j'}$ give rise to similar transitions in $R_{ij}, R_{i'j'}$, respectively. □

C.2 State and Path Projection Results

PROPOSITION 6.3.1 (I-STATE PROJECTION). *Let $J \subseteq I$, and let s be an I-state. Then*

$$s \models f \;\text{ iff }\; s{\uparrow}J \models f$$

where f is a formula of $\mathcal{L}(\bigcup_{i \in dom(J)} \mathcal{AP}_i,\ \neg,\ \wedge)$.

PROOF. The proof is by induction on the structure of f.

$f \in \bigcup_{i \in dom(J)} \mathcal{AP}_i$. $s \models f$ iff $s(f) = true$ iff (since $s{\uparrow}J = (\bigcup_{i \in dom(J)} s{\uparrow}i) \cup (\bigcup_{(i,j) \in J} s{\uparrow}\mathcal{SH}_{ij})$) $s{\uparrow}J(f) = true$ iff $s{\uparrow}J \models f$.

$f = g \wedge h$. $s \models (g \wedge h)$ iff ($s \models g$ and $s \models h$) iff (by the inductive hypothesis) ($s{\uparrow}J \models g$ and $s{\uparrow}J \models h$) iff $s{\uparrow}J \models (g \wedge h)$.

$f = \neg g$. $s \models \neg g$ iff not ($s \models g$) iff (by the inductive hypothesis) not ($s{\uparrow}J \models g$) iff $s{\uparrow}J \models \neg g$. □

PROPOSITION 6.3.2 (LOCAL STATE PROJECTION). *Let $i \in dom(J)$, and let s_J be a J-state. Then*

$$s_J \models f_i \text{ iff } s_J{\uparrow}i \models f_i$$

where f_i is a formula of $\mathcal{L}(\mathcal{AP}_i, \neg, \wedge)$.

PROOF. Analogous to that of Proposition 6.3.1. □

LEMMA 6.3.3 (PATH PROJECTION). *Let π be a path in M_I, and let $J \subseteq I$. Then*

$$\pi \models f \text{ iff } \pi{\uparrow}J \models f$$

where f is a formula of $\mathcal{L}(\bigcup_{i \in dom(J)} \mathcal{AP}_i,\ \neg,\ \wedge,\ U)$.

PROOF. The proof is by induction on the structure of f.

$f \in \bigcup_{i \in dom(J)} \mathcal{AP}_i$. Let s, s_J be the initial states of $\pi, \pi{\uparrow}J$ respectively. By the definition of path projection, $s_J = s{\uparrow}J$, and so s_J and s agree on all atomic propositions in $\bigcup_{i \in dom(J)} \mathcal{AP}_i$. Hence $s \models f$ iff $s_J \models f$. Also $\pi \models f$ iff $s \models f$ and $\pi{\uparrow}J \models f$ iff $s_J \models f$ by CTL* semantics (Section 3.1). These three equivalences yield $\pi \models f$ iff $\pi{\uparrow}J \models f$.

$f = g \wedge h$. $\pi \models (g \wedge h)$ iff ($\pi \models g$ and $\pi \models h$) iff, by the inductive hypothesis, ($\pi{\uparrow}J \models g$ and $\pi{\uparrow}J \models h$) iff $\pi{\uparrow}J \models (g \wedge h)$.

$f = \neg g$. $\pi \models \neg g$ iff not($\pi \models g$) iff, by the inductive hypothesis, not($\pi{\uparrow}J \models g$) iff $\pi{\uparrow}J \models \neg g$.

$f = gUh$. Proof by double implication.

Left to right: $\pi \models gUh$ implies $\pi{\uparrow}J \models gUh$.

Assume $\pi \models gUh$. Therefore, for some $n_0 \geq 1$, we have by CTL* semantics (Section 3.1):

$$\pi^{n_0} \models h \text{ and } \bigwedge n\,.\,(1 \leq n < n_0 \Rightarrow \pi^n \models g). \tag{a}$$

By (a) and the induction hypothesis, we get

$$\pi^{n_0}{\uparrow}J \models h \text{ and } \bigwedge n\,.\,(1 \leq n < n_0 \Rightarrow \pi^n{\uparrow}J \models g). \tag{b}$$

Let B^{m_0} denote the J-block of π that contains s^{n_0} (the first state of π^{n_0}). By the path projection definition (5.1.2.1), $\pi^{n_0}{\uparrow}J = (\pi{\uparrow}J)^{m_0}$, since the m_0'th J-block of π corresponds to the m_0'th state of $\pi{\uparrow}J$. So, by (b) we have

$$(\pi{\uparrow}J)^{m_0} \models h. \tag{c}$$

Let m be an arbitrary integer in $[1 : (m_0 - 1)]$. The first state s_J^m of $(\pi{\uparrow}J)^m$ corresponds to the mth J-block of π (by the path projection definition (5.1.2.1)). Let s^l (the first state of π^l) be an arbitrary state of the mth J-block of π. Hence

$(\pi{\uparrow}J)^m = \pi^l{\uparrow}J$. Also, since $m < m_0$, s^l occurs in an earlier J-block of π than s^{n_0}, and therefore $l < n_0$. So by (b) we have

$$\pi^l{\uparrow}J \models g. \tag{d}$$

By (d) and $(\pi{\uparrow}J)^m = \pi^l{\uparrow}J$, we get

$$(\pi{\uparrow}J)^m \models g. \tag{e}$$

But m is an arbitrary integer in $[1 : (m_0 - 1)]$. So, by (c) and (e) we get

$$(\pi{\uparrow}J)^{m_0} \models h \text{ and } \bigwedge m\,.\,(1 \leq m < m_0 \Rightarrow (\pi{\uparrow}J)^m \models g).$$

By CTL* semantics (Section 3.1), this is equivalent to $\pi{\uparrow}J \models gUh$.

Right to left: $\pi{\uparrow}J \models gUh$ implies $\pi \models gUh$.

Assume $\pi{\uparrow}J \models gUh$. Therefore, for some $m_0 \geq 1$, we have by CTL* semantics (Section 3.1)

$$(\pi{\uparrow}J)^{m_0} \models h \text{ and } \bigwedge m\,.\,(1 \leq m < m_0 \Rightarrow (\pi{\uparrow}J)^m \models g). \tag{a}$$

Let s^{n_0} (the first state of π^{n_0}) be the first state of the m_0th J-block of π. By the path projection definition (5.1.2.1), $\pi^{n_0}{\uparrow}J = (\pi{\uparrow}J)^{m_0}$. Hence, by (a) we have

$$\pi^{n_0}{\uparrow}J \models h. \tag{b}$$

Now let m be an arbitrary integer in $[1 : (m_0 - 1)]$. By (a), we get

$$(\pi{\uparrow}J)^m \models g. \tag{c}$$

Furthermore, let s^n (the first state of π^n) be an arbitrary state of the mth J-block of π. By the path projection definition (5.1.2.1), $\pi^n{\uparrow}J = (\pi{\uparrow}J)^m$. Also, since $m < m_0$, s^n occurs in an earlier J-block of π than s^{n_0}, and therefore $n < n_0$. Furthermore, since s^{n_0} is the *first* state of the m_0th J-block of π, we see that n ranges over $[1 : (n_0 - 1)]$, since s^{n_0-1} occurs in an earlier J-block of π than s^{n_0}. Hence, by (c) and $\pi^n{\uparrow}J = (\pi{\uparrow}J)^m$ we have

$$\bigwedge n\,.\,(1 \leq n < n_0 \Rightarrow \pi^n{\uparrow}J \models g). \tag{d}$$

From (b) and (d) we get

$$\pi^{n_0}{\uparrow}J \models h \text{ and } \bigwedge n\,.\,(1 \leq n < n_0 \Rightarrow \pi^n{\uparrow}J \models g). \tag{e}$$

From (e) and the induction hypothesis applied to g, h, we get

$$\pi^{n_0} \models h \text{ and } \bigwedge n\,.\,(1 \leq n < n_0 \Rightarrow \pi^n \models g).$$

By CTL* semantics (Section 3.1), this is equivalent to $\pi \models gUh$. $\square$

C.3 Mapping of I-Structures into J-Structures

LEMMA 6.4.1 (TRANSITION MAPPING). *For all I-states $s, t \in S_I$ and $i \in dom(I)$,*

$$\begin{aligned}
&s \xrightarrow{i} t \in R_I \text{ iff :}\\
&\qquad \textstyle\bigwedge j \in I(i)\,.\,(s{\uparrow}ij \xrightarrow{i} t{\uparrow}ij \in R_{ij}) \textit{ and}\\
&\qquad \textstyle\bigwedge j \in dom(I) - \hat{I}(i)\,.\,(s{\uparrow}j = t{\uparrow}j) \textit{ and}\\
&\qquad \textstyle\bigwedge j, k \in dom(I) - \{i\}, j\,I\,k\,.\,(s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}).
\end{aligned}$$

PROOF. Let s, t be arbitrary I-states, and let i be an arbitrary element of $dom(I)$. By the I-structure definition (5.3.1), $s \xrightarrow{i} t \in R_I$ is equivalent to

$$\begin{array}{ll} (s{\uparrow}i, \bigwedge_{j \in I(i)} B_i^j \rightarrow //_{j \in I(i)} A_i^j, t{\uparrow}i) \text{ is an arc in } P_i^I \text{ and} & \\ \bigwedge j \in I(i) \,.\, (s{\uparrow}ij(B_i^j) = true \text{ and } < s{\uparrow}\mathcal{SH}_{ij} > A_i^j < t{\uparrow}\mathcal{SH}_{ij} >) \text{ and} & \\ \bigwedge j \in dom(I) - \{i\} \,.\, (s{\uparrow}j = t{\uparrow}j) \text{ and} & \\ \bigwedge j, k \in dom(I) - \{i\}, j\, I\, k \,.\, (s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}). & \text{(a)} \end{array}$$

By the MP-synthesis definition (5.1.1), (a) is equivalent to

$$\begin{array}{ll} \bigwedge j \in I(i) \,.\, ((s{\uparrow}i, B_i^j \rightarrow A_i^j, t{\uparrow}i) \text{ is an arc in } P_i^j) \text{ and} & \\ \bigwedge j \in I(i) \,.\, (s{\uparrow}ij(B_i^j) = true \text{ and } < s{\uparrow}\mathcal{SH}_{ij} > A_i^j < t{\uparrow}\mathcal{SH}_{ij} >) \text{ and} & \\ \bigwedge j \in dom(I) - \{i\} \,.\, (s{\uparrow}j = t{\uparrow}j) \text{ and} & \\ \bigwedge j, k \in dom(I) - \{i\}, j\, I\, k \,.\, (s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}). & \text{(b)} \end{array}$$

Now (b), by rewriting the "$\bigwedge j \in dom(I) - \{i\}$..." universal quantification as the conjunction of "for all j in $I(i)$..." and "$\bigwedge j \in dom(I) - \hat{I}(i)$...", is equivalent to

$$\begin{array}{ll} \bigwedge j \in I(i) \,.\, ((s{\uparrow}i, B_i^j \rightarrow A_i^j, t{\uparrow}i) \text{ is an arc in } P_i^j) \text{ and} & \\ \bigwedge j \in I(i) \,.\, (s{\uparrow}ij(B_i^j) = true \text{ and } < s{\uparrow}\mathcal{SH}_{ij} > A_i^j < t{\uparrow}\mathcal{SH}_{ij} >) \text{ and} & \\ \bigwedge j \in I(i) \,.\, (s{\uparrow}j = t{\uparrow}j) \text{ and} & \\ \bigwedge j \in dom(I) - \hat{I}(i) \,.\, (s{\uparrow}j = t{\uparrow}j) \text{ and} & \\ \bigwedge j, k \in dom(I) - \{i\}, j\, I\, k \,.\, (s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}). & \text{(c)} \end{array}$$

By merging all three "$\bigwedge j \in I(i)$..." universal quantifications, (c) is equivalent to

$$\begin{array}{ll} \bigwedge j \in I(i) \,.\, (& \\ \quad (s{\uparrow}i, B_i^j \rightarrow A_i^j, t{\uparrow}i) \text{ is an arc in } P_i^j \text{ and} & \\ \quad s{\uparrow}ij(B_i^j) = true \text{ and} & \\ \quad < s{\uparrow}\mathcal{SH}_{ij} > A_i^j < t{\uparrow}\mathcal{SH}_{ij} > \text{ and} & \\ \quad s{\uparrow}j = t{\uparrow}j \text{) and} & \\ \bigwedge j \in dom(I) - \hat{I}(i) \,.\, (s{\uparrow}j = t{\uparrow}j) \text{ and} & \\ \bigwedge j, k \in dom(I) - \{i\}, j\, I\, k \,.\, (s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}). & \text{(d)} \end{array}$$

By the obvious identities $s{\uparrow}i = (s{\uparrow}ij){\uparrow}i$, $s{\uparrow}\mathcal{SH}_{ij} = (s{\uparrow}ij){\uparrow}\mathcal{SH}_{ij}$, $s{\uparrow}j = (s{\uparrow}ij){\uparrow}j$, $t{\uparrow}i = (t{\uparrow}ij){\uparrow}i$, $t{\uparrow}\mathcal{SH}_{ij} = (t{\uparrow}ij){\uparrow}\mathcal{SH}_{ij}$, $t{\uparrow}j = (t{\uparrow}ij){\uparrow}j$, (d) is equivalent to

$$\begin{array}{ll} \bigwedge j \in I(i) \,.\, (& \\ \quad ((s{\uparrow}ij){\uparrow}i, B_i^j \rightarrow A_i^j, (t{\uparrow}ij){\uparrow}i) \text{ is an arc in } P_i^j \text{ and} & \\ \quad s{\uparrow}ij(B_i^j) = true \text{ and} & \\ \quad < (s{\uparrow}ij){\uparrow}\mathcal{SH}_{ij} > A_i^j < (t{\uparrow}ij){\uparrow}\mathcal{SH}_{ij} > \text{ and} & \\ \quad (s{\uparrow}ij){\uparrow}j = (t{\uparrow}ij){\uparrow}j \text{) and} & \\ \bigwedge j \in dom(I) - \hat{I}(i) \,.\, (s{\uparrow}j = t{\uparrow}j) \text{ and} & \\ \bigwedge j, k \in dom(I) - \{i\}, j\, I\, k \,.\, (s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}). & \text{(e)} \end{array}$$

By the pair-structure definition (5.2.1), (e) is equivalent to

$$\begin{array}{l} \bigwedge j \in I(i) \,.\, (s{\uparrow}ij \xrightarrow{i} t{\uparrow}ij \in R_{ij}) \text{ and} \\ \bigwedge j \in dom(I) - \hat{I}(i) \,.\, (s{\uparrow}j = t{\uparrow}j) \text{ and} \\ \bigwedge j, k \in dom(I) - \{i\}, j\, I\, k \,.\, (s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}). \end{array}$$

The lemma then follows by transitivity of equivalence. □

Corollary 6.4.2 (Transition Mapping). *Let* $J \subseteq I$ *and* $i \in dom(J)$. *If* $s \xrightarrow{i} t \in R_I$, *then* $s{\uparrow}J \xrightarrow{i} t{\uparrow}J \in R_J$.

Proof. Assume $s \xrightarrow{i} t \in R_I$. Then, by the transition-mapping lemma (6.4.1), we have

$$\begin{aligned}
&\textstyle\bigwedge j \in I(i) \,.\, (s{\uparrow}ij \xrightarrow{i} t{\uparrow}ij \in R_{ij}) \text{ and}\\
&\textstyle\bigwedge j \in dom(I) - \hat{I}(i) \,.\, (s{\uparrow}j = t{\uparrow}j) \text{ and}\\
&\textstyle\bigwedge j,k \in dom(I) - \{i\}, j\,I\,k \,.\, (s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}). \qquad \text{(a)}
\end{aligned}$$

From $\bigwedge j \in I(i) \,.\, (s{\uparrow}ij \xrightarrow{i} t{\uparrow}ij \in R_{ij})$ and the pair-structure definition (5.2.1), we get $\bigwedge j \in I(i) \,.\, (s{\uparrow}j = t{\uparrow}j)$. Together with (a), this yields

$$\begin{aligned}
&\textstyle\bigwedge j \in I(i) \,.\, (s{\uparrow}ij \xrightarrow{i} t{\uparrow}ij \in R_{ij}) \text{ and}\\
&\textstyle\bigwedge j \in dom(I) - \{i\} \,.\, (s{\uparrow}j = t{\uparrow}j) \text{ and}\\
&\textstyle\bigwedge j,k \in dom(I) - \{i\}, j\,I\,k \,.\, (s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}). \qquad \text{(b)}
\end{aligned}$$

Since $J \subseteq I$, we have $dom(J) \subseteq dom(I)$. Now $i \in dom(J)$ by assumption, and so $i \in dom(I)$. Thus, by $J \subseteq I$, we have $J(i) \subseteq I(i)$ and $\bigwedge j,k \,.\, (j\,J\,k$ implies $j\,I\,k)$. Thus, from (b) we get

$$\begin{aligned}
&\textstyle\bigwedge j \in J(i) \,.\, (s{\uparrow}ij \xrightarrow{i} t{\uparrow}ij \in R_{ij}) \text{ and}\\
&\textstyle\bigwedge j \in dom(I) - \{i\} \,.\, (s{\uparrow}j = t{\uparrow}j) \text{ and}\\
&\textstyle\bigwedge j,k \in dom(J) - \{i\}, j\,J\,k \,.\, (s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}). \qquad \text{(c)}
\end{aligned}$$

Since $J \subseteq I$, we have $dom(J) \subseteq dom(I)$. Also, $i \in \hat{J}(i)$, and hence $dom(J) - \hat{J}(i) \subseteq dom(I) - \{i\}$. Thus, by (c) we have

$$\begin{aligned}
&\textstyle\bigwedge j \in J(i) \,.\, (s{\uparrow}ij \xrightarrow{i} t{\uparrow}ij \in R_{ij}) \text{ and}\\
&\textstyle\bigwedge j \in dom(J) - \hat{J}(i) \,.\, (s{\uparrow}j = t{\uparrow}j) \text{ and}\\
&\textstyle\bigwedge j,k \in dom(J) - \{i\}, j\,J\,k \,.\, (s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}). \qquad \text{(d)}
\end{aligned}$$

Now $s{\uparrow}ij = (s{\uparrow}J){\uparrow}ij$ when $j \in I(i)$, and $s{\uparrow}j = (s{\uparrow}J){\uparrow}j$ when $j \in dom(J)$, and $s{\uparrow}\mathcal{SH}_{jk} = (s{\uparrow}J){\uparrow}\mathcal{SH}_{jk}$ when $j\,J\,k$. Thus, (d) can be rewritten as

$$\begin{aligned}
&\textstyle\bigwedge j \in J(i) \,.\, ((s{\uparrow}J){\uparrow}ij \xrightarrow{i} (t{\uparrow}J){\uparrow}ij \in R_{ij}) \text{ and}\\
&\textstyle\bigwedge j \in dom(J) - \hat{J}(i) \,.\, ((s{\uparrow}J){\uparrow}j = (t{\uparrow}J){\uparrow}j) \text{ and}\\
&\textstyle\bigwedge j,k \in dom(J) - \{i\}, j\,J\,k \,.\, ((s{\uparrow}J){\uparrow}\mathcal{SH}_{jk} = (t{\uparrow}J){\uparrow}\mathcal{SH}_{jk}). \qquad \text{(e)}
\end{aligned}$$

By the transition-mapping lemma (6.4.1) with $I := J$, and (e), we get $s{\uparrow}J \xrightarrow{i} t{\uparrow}J \in R_J$ as required. □

Lemma 6.4.3 (Path Mapping). *Let* $J \subseteq I$. *If* π *is a path in* M_I, *then* $\pi{\uparrow}J$ *is a path in* M_J.

Proof. Let n be an arbitrary integer greater than zero, and let $u_J^n \xrightarrow{d_n} u_J^{n+1}$ be the nth transition along $\pi{\uparrow}J$. Thus $d_n \in dom(J)$, and u_J^n, u_J^{n+1} are the nth, $(n+1)$st states respectively along $\pi{\uparrow}J$. By the path projection definition (5.1.2.1), $u_J^n = B^n{\uparrow}J$, $u_J^{n+1} = B^{n+1}{\uparrow}J$, where B^n, B^{n+1} are the nth, $(n+1)$st, J-blocks respectively along π. Let s, t, be the last, first states of B^n, B^{n+1}, respectively, so $s{\uparrow}J = B^n{\uparrow}J = u_J^n$, $t{\uparrow}J = B^{n+1}{\uparrow}J = u_J^{n+1}$. Also, by the path projection definition (5.1.2.1), $s \xrightarrow{d_n} t$

is a transition along π, and therefore $s \xrightarrow{d_n} t \in R_I$, since π is a path in M_I. Hence, by the transition-mapping corollary (6.4.2), $s{\uparrow}J \xrightarrow{d_n} t{\uparrow}J \in R_J$, so $u_J^n \xrightarrow{d_n} u_J^{n+1} \in R_J$.

Since every transition of $\pi{\uparrow}J$ is a transition in R_J, it follows that $\pi{\uparrow}J$ is a path in M_J. $\square$

Corollary 6.4.4 (Path Mapping). *Let $J \subseteq I$. If π is an initialized path in M_I then $\pi{\uparrow}J$ is an initialized path in M_J.*

Proof. Let B^0 be the first J-block of π. By the path projection definition (5.1.2.1), the first state of $\pi{\uparrow}J$ is $B^0{\uparrow}J$. Since π is initialized, B^0 contains some initial state $s_I^0 \in S_I^0$. Hence $B^0{\uparrow}J = s_I^0{\uparrow}J$.

Now by the MP-synthesis definition (5.1.1), we have $\bigwedge(i,j) \in I \,.\, (s_I^0{\uparrow}ij \in S_{ij}^0)$. Since $J \subseteq I$, we have $\bigwedge(i,j) \in J \,.\, (s_I^0{\uparrow}ij \in S_{ij}^0)$. Also, by definition of ${\uparrow}J$ (Section 6), $\bigwedge(i,j) \in J \,.\, ((s_I^0{\uparrow}J){\uparrow}ij = s_I^0{\uparrow}ij)$. Hence, we have $\bigwedge(i,j) \in J \,.\, ((s_I^0{\uparrow}J){\uparrow}ij \in S_{ij}^0)$. Now, by the MP-synthesis definition (5.1.1) with J replacing I we have $S_J^0 = \{s_J \mid \bigwedge(i,j) \in J \,.\, (s_J{\uparrow}ij \in S_{ij}^0)\}$. Hence we conclude $s_I^0{\uparrow}J \in S_J^0$. Thus, the first state of $\pi{\uparrow}J$ is an initial state of M_J. By the path-mapping lemma (6.4.3), $\pi{\uparrow}J$ is a path in M_J. It follows that $\pi{\uparrow}J$ is an initialized path in M_J. $\square$

Corollary 6.4.5 (State Mapping). *Let $J \subseteq I$. If t is a reachable state in M_I, then $t{\uparrow}J$ is a reachable state in M_J.*

Proof. Since t is reachable in M_I, there must exist at least one initialized path π in M_I which ends in state t. By the path-mapping corollary (6.4.4), $\pi{\uparrow}J$ is an initialized path in M_J. By the path projection definition (5.1.2.1), $\pi{\uparrow}J$ ends in state $t{\uparrow}J$. Hence $t{\uparrow}J$ is reachable in M_J. $\square$

Corollary 6.4.6 (Relativized State Mapping). *Let $J \subseteq I$. If t is a s-reachable state in M_I, then $t{\uparrow}J$ is a $s{\uparrow}J$-reachable state in M_J.*

Proof. Since t is s-reachable in M_I, there must exist at least one path π in M_I which starts in state s and ends in state t. By the path-mapping lemma (6.4.3), $\pi{\uparrow}J$ is a path in M_J. By the path projection definition (5.1.2.1), $\pi{\uparrow}J$ starts in state $s{\uparrow}J$ and ends in state $t{\uparrow}J$. Hence $t{\uparrow}J$ is $s{\uparrow}J$-reachable in M_J. $\square$

C.4 Deadlock Freedom of the Many-Process System

Proposition 6.5.4.1 (Wait-For-Graph Projection). *Let $J \subseteq I$ and $i\,J\,j$. Furthermore, let s_I be an arbitrary I-state. Then*

(1) $P_i^I \longrightarrow a_i^I \in W_I(s_I)$ iff $P_i^J \longrightarrow a_i^J \in W_J(s_I{\uparrow}J)$, and

(2) $a_i^I \longrightarrow P_j^I \in W_I(s_I)$ iff $a_i^J \longrightarrow P_j^J \in W_J(s_I{\uparrow}J)$.

Proof. By assumption, $i\,J\,j$ and $J \subseteq I$. Hence $i\,I\,j$.

Proof of clause (1). By the wait-for-graph definition (6.5.2.1), $P_i^I \longrightarrow a_i^I \in W_I(s_I)$ iff $s_I{\uparrow}i = a_i^I.start$. Since $i \in dom(J)$, we have $(s_I{\uparrow}J){\uparrow}i = s_I{\uparrow}i$ by definition of ${\uparrow}J$. Thus $s_I{\uparrow}i = a_i^I.start$ iff $(s_I{\uparrow}J){\uparrow}i = a_i^J.start$ (since $a_i^I.start = a_i^J.start = s_i$). Finally, by the wait-for-graph definition (6.5.2.1) and $i\,J\,j$, $(s_I{\uparrow}J){\uparrow}i = a_i^J.start$ iff

$P_i^J \longrightarrow a_i^J \in W_J(s_I {\uparrow} J)$. These three equivalences together yield clause (1) (using transitivity of equivalence).

Proof of clause (2). By the wait-for-graph definition (6.5.2.1), $a_i^I \longrightarrow P_j^I \in W_I(s_I)$ iff $s{\uparrow}ij \not\models a_i^I.guard_j$. Since $i\, J\, j$, we have $(s_I{\uparrow}J){\uparrow}ij = s_I{\uparrow}ij$ by definition of ${\uparrow}J$. Also, $a_i^I.guard_j = a_i^J.guard_j = \bigvee_{\ell \in [1:n]} B_{i,\ell}^j$. Thus $s_I{\uparrow}ij \not\models a_i^I.guard_j$ iff $(s_I{\uparrow}J){\uparrow}ij \not\models a_i^J.guard_j$ Finally, by the wait-for-graph definition (6.5.2.1) and $i\, J\, j$, $(s_I{\uparrow}J){\uparrow}ij \not\models a_i^J.guard_j$ iff $a_i^J \longrightarrow P_j^J \in W_J(s_I{\uparrow}J)$. These three equivalences together yield clause (2), (using transitivity of equivalence, and noting that $s \not\models B$ and $s(B) = false$ have identical meaning). □

Theorem 6.5.4.3 (Supercycle-Free Wait-For-Graph). *If the wait-for-graph assumption WG holds, and $W_I(s_I^0)$ is supercycle-free for every initial state $s_I^0 \in S_I^0$, then for every reachable state t of M_I, $W_I(t)$ is supercycle-free.*

Proof. Let t be an arbitrary reachable state of M_I, and let s be an arbitrary reachable state of M_I such that $s \xrightarrow{k} t$ for some $k \in dom(I)$. We shall establish that

$$\text{if } W_I(t) \text{ is supercyclic, then } W_I(s) \text{ is supercyclic.} \tag{P1}$$

The contrapositive of P1 together with the assumption that $W_I(s_I^0)$ is supercycle-free for all $s_I^0 \in S_I^0$ is sufficient to establish the conclusion of the theorem (by induction on the length of a path from some $s_I^0 \in S_I^0$ to t).

We say that an edge is *k-incident* iff at least one of its vertices is P_k^I or a_k^I. The following (P2) will be useful in proving P1

$$\text{if edge } e \text{ is not } k\text{-incident, then } e \in W_I(t) \text{ iff } e \in W_I(s). \tag{P2}$$

Proof of P2. If e is not k-incident, then, by the wait-for-graph definition (6.5.2.1), either $e = P_h^I \longrightarrow a_h^I$, or $e = a_h^I \longrightarrow P_\ell^I$, for some h, ℓ such that $h \neq k, \ell \neq k$. From $h \neq k, \ell \neq k$ and $s \xrightarrow{k} t \in R_I$, we have $s{\uparrow}h = t{\uparrow}h$ and $s{\uparrow}h\ell = t{\uparrow}h\ell$ by the wait-for-graph definition (6.5.2.1). Since $e \in W_I(t), e \in W_I(s)$ are determined solely by $t{\uparrow}h\ell, s{\uparrow}h\ell$ respectively, (see the wait-for-graph definition (6.5.2.1)), P2 follows. (End proof of P2.)

Let v be a vertex in a supercycle SC. We define $depth_{SC}(v)$ to be the length of the longest backward path in SC which starts in v. If there exists an infinite backward path (i.e., one that traverses a cycle) in SC starting in v, then $depth_{SC}(v) = \omega$ (ω for "infinity"). We now establish that

$$\text{every supercycle } SC \text{ contains at least one cycle.} \tag{P3}$$

Proof of P3. Suppose P3 does not hold, and SC is a supercycle containing no cycles. Therefore, all backward paths in SC are finite, and so by definition of $depth_{SC}$ all vertices of SC have finite depth. Thus, there is at least one vertex v in SC with maximal depth. But, by definition of $depth_{SC}$, v has no successors in SC, which, by the supercycle definition (6.5.3.1), contradicts the assumption that SC is a supercycle. (End proof of P3.)

Our final prerequisite for the proof of P1 is if SC is a supercycle in $W_I(s)$, then the graph SC' obtained from SC by removing all vertices of finite depth from SC (along with incident edges) is also a supercycle in $W_I(s)$. (P4)

Proof of P4. By P3, $SC' \neq \emptyset$. Thus SC' satisfies clause (1) of the supercycle definition (6.5.3.1). Let v be an arbitrary vertex of SC'. Thus $v \in SC$ and $depth_{SC}(v) = \omega$ by definition of SC'. Let w be an arbitrary successor of v in SC. $depth_{SC}(w) = \omega$ by definition of *depth*. Hence $w \in SC'$. Furthermore, w is a successor of v in SC', by definition of SC'. Thus every vertex v of SC' is also a vertex of SC, and the successors of v in SC' are the same as the successors of v in SC Now since SC is a supercycle, every vertex v in SC has enough successors in SC to satisfy clauses (2) and (3) of the supercycle definition (6.5.3.1). It follows that every vertex v in SC' has enough successors in SC' to satisfy clauses (2) and (3) of the supercycle definition (6.5.3.1). (End proof of P4.)

We now present the proof of (P1). We assume the antecedent of P1 and establish the consequent. Let SC be some supercycle in $W_I(t)$. Let SC' be the graph obtained from SC by removing all vertices of finite depth from SC (along with incident edges). We now show that $P_k^I \notin SC'$ and that SC' contains no move vertex of the form a_k^I. There are two cases.

Case 1: $P_k^I \notin SC$. Then obviously $P_k^I \notin SC'$. Now suppose some node of the form a_k^I is in SC'. By definition of SC', we have $a_k^I \in SC$ and $depth_{SC}(a_k^I) = \omega$. Hence, by definition of *depth*, there exists an infinite backward path in SC starting in a_k^I. Thus a_k^I must have a predecessor in SC. By the supercycle definition (6.5.3.1), P_k^I is the only possible predecessor of a_k^I in SC, and hence $P_k^I \in SC$, contrary to the case assumption. We therefore conclude that SC' contains no vertices of the form a_k^I. (End of case 1.)

Case 2: $P_k^I \in SC$. By the supercycle definition (6.5.3.1),

$$\bigwedge a_k^I \in W_I(t) \,.\, (\bigvee \ell \,.\, (a_k^I \longrightarrow P_\ell^I \in W_I(t))). \tag{a}$$

Since there are exactly n moves a_k^I of process P_k^I in $W_I(t)$ ($n = |t_k.moves|$), we can select $\ell_1, \ldots, \ell_n$ (where $\ell_1, \ldots, \ell_n$ are not necessarily pairwise distinct) such that

$$\bigwedge a_k^I \in W_I(t) \,.\, (\bigvee \ell \in \{\ell_1, \ldots, \ell_n\} \,.\, (a_k^I \longrightarrow P_\ell^I \in W_I(t))). \tag{b}$$

Now let $J = \{\{j,k\}, \{k,\ell_1\}, \ldots, \{k,\ell_n\}\}$ where j is an arbitrary element of $I(k)$. Applying the wait-for-graph projection proposition (6.5.4.1) to (b) gives us

$$\bigwedge a_k^J \in W_J(t{\uparrow}J) \,.\, (\bigvee \ell \in \{\ell_1, \ldots, \ell_n\} \,.\, (a_k^J \longrightarrow P_\ell^J \in W_J(t{\uparrow}J))). \tag{c}$$

Now $s \xrightarrow{k} t \in R_I$ by assumption. Hence $s{\uparrow}J \xrightarrow{k} t{\uparrow}J \in R_J$ by the transition-mapping corollary (6.4.2). Also, by the state-mapping corollary (6.4.5) $s{\uparrow}J$ is reachable in M_J, since s is reachable in M_I. Thus we can apply the wait-for-graph assumption to $t{\uparrow}J$ to get

$$\bigwedge a_j^J \,.\, (a_j^J \longrightarrow P_k^J \notin W_J(t{\uparrow}J))$$

or

$$\bigvee a_k^J \in W_J(t{\uparrow}J) \,.\, (\bigwedge \ell \in \{\ell_1, \ldots, \ell_n\} \,.\, (a_k^J \longrightarrow P_\ell^J \notin W_J(t{\uparrow}J))). \tag{d}$$

Now (c) contradicts the second disjunct of (d). Hence

$$\bigwedge a_j^J . (a_j^J \longrightarrow P_k^J \notin W_J(t{\uparrow}J)),$$

and applying the wait-for-graph projection proposition (6.5.4.1) to this gives us

$$\bigwedge a_j^J . (a_j^I \longrightarrow P_k^I \notin W_I(t)).$$

Since j is an arbitrary element of $I(k)$, we conclude that P_k^I has no incoming edges in $W_I(t)$. Thus, by definition of *depth*, $depth_{SC}(P_k^I) = 0$, and so $P_k^I \notin SC'$.

Now suppose some node of the form a_k^I is in SC'. By definition of SC', we have $a_k^I \in SC$ and $depth_{SC}(a_k^I) = \omega$. Hence, by definition of *depth*, there exists an infinite backward path in SC starting in a_k^I. Thus a_k^I must have a predecessor in SC. By the supercycle definition (6.5.3.1), P_k^I is the only possible predecessor of a_k^I in SC, and hence there exists an infinite backward path in SC starting in P_k^I. Thus $depth_{SC}(P_k^I) = \omega$ by definition of *depth*. But we have established $depth_{SC}(P_k^I) = 0$, so we conclude that SC' contains no vertices of the form a_k^I. (End of case 2.)

In both cases, $P_k^I \notin SC'$, and SC' contains no move vertex of the form a_k^I. Thus every edge of SC' is not k-incident. Hence, by P2, every edge of SC' is an edge of $W_I(s)$ (since $SC' \subseteq W_I(t)$). By P4, SC' is a supercycle, so $W_I(s)$ is supercyclic. Thus P1 is established, which establishes the theorem. □

C.5 Liveness Properties

Proposition 6.7.3.4 (Sometimes-Blocking). *Let s_{ij} be a reachable state of M_{ij} and $r_i \in \mathcal{L}(\mathcal{AP}_i, \neg, \wedge)$. If $M_{ij}, s_{ij} \models \neg r_i \wedge AFr_i$, then $s_{ij}{\uparrow}i$ is sometimes-blocking.*

Proof. We assume the antecedent and establish the consequent. Define a P_j^i-path to be a path in M_{ij} composed entirely of P_j^i-transitions. For a pair move $a_i^j = (s_i, \oplus_{\ell \in [1:n]} B_{i,\ell}^j \to A_{i,\ell}^j, t_i)$, let $a_i^j.start, a_i^j.guard$ denote $s_i, \bigvee_{\ell \in [1:n]} B_{i,\ell}^j$ respectively.

By assumption, $s_{ij} \not\models r_i$. Also, by M_{ij}, $s_{ij} \models AFr_i$ and CTL* semantics, every path starting in s_{ij} must lead to a state t_{ij} which fulfills AFr_i, i.e., $t_{ij} \models r_i$. Since $s_{ij}{\uparrow}i \neq t_{ij}{\uparrow}i$ (otherwise we could not have $s_{ij} \not\models r_i$ and $t_{ij} \models r_i$), we conclude, by the pair-structure definition (5.2.1), that every maximal path starting in s_{ij} must eventually contain a P_i^j-transition, because a P_j^i-transition cannot change any atomic proposition in $\mathcal{AP}_i$. Thus, every maximal P_j^i-path starting in s_{ij} (if any) is finite and ends in a state which has no outgoing P_j^i-transitions.

We now demonstrate the existence of a reachable state u_{ij} in M_{ij} such that $u_{ij}{\uparrow}i = s_{ij}{\uparrow}i$ and such that u_{ij} has no outgoing P_j^i-transitions. There are two cases.

Case 1: There are no P_j^i-paths starting in s_{ij}. Therefore s_{ij} has no outgoing P_j^i-transitions, so let u_{ij} be s_{ij}. (End of case 1.)

Case 2: There is at least one P_j^i-path starting in s_{ij}. Hence there is at least one maximal P_j^i-path starting in s_{ij}. By the above discussion, this path is finite and ends in a state with no outgoing P_j^i-transitions. Let u_{ij} be this state. Since there is a P_j^i-path starting in s_{ij} and ending in u_{ij} we conclude that $u_{ij}{\uparrow}i = s_{ij}{\uparrow}i$, since, by the pair-structure definition (5.2.1) a P_j^i-transition cannot change the truth value assigned to any atomic proposition in $\mathcal{AP}_i$. Also, u_{ij} is reachable, since s_{ij} is reachable and there is a path from s_{ij} to u_{ij}. (End of case 2.)

Since u_{ij} has no outgoing P^i_j-transitions, all of the moves in P^i_j (we are assuming compact notation here) are disabled in state u_{ij}, so we have

$$u_{ij} \models \bigwedge a^i_j \in P^i_j \,.\, (\neg \{\!\!\{a^i_j.start\}\!\!\} \vee \neg a^i_j.guard), \tag{a}$$

since a move is disabled iff control is not at its start state or its guard evaluates to false. Since every local state of a process has at least one outgoing arc (Section 2), there exists at least one move a^i_j in P^i_j such that $u_{ij} \models \{\!\!\{a^i_j.start\}\!\!\}$. From this and (a), we have

$$u_{ij} \models \bigvee a^i_j \in P^i_j \,.\, (\{\!\!\{a^i_j.start\}\!\!\} \wedge \neg a^i_j.guard). \tag{b}$$

Finally, since u_{ij} is reachable and $u_{ij}{\uparrow}i = s_{ij}{\uparrow}i$, we obtain from (b)

$$\bigvee s^0_{ij} \in S^0_{ij} \,.\, (\, M_{ij}, s^0_{ij} \models EF(\, \{\!\!\{s_{ij}{\uparrow}i\}\!\!\} \wedge (\bigvee a^i_j \in P^i_j \,.\, (\{\!\!\{a^i_j.start\}\!\!\} \wedge \neg a^i_j.guard))\,)\,).$$

By the sometimes-blocking state definition (6.7.3.1), we conclude that $s_{ij}{\uparrow}i$ is sometimes-blocking. □

LEMMA 6.7.4.1 (PROGRESS). *If*

(1) the liveness assumption LV holds,

(2) for every reachable I-state s, $W_I(s)$ is supercycle-free, and

(3) v is a reachable I-state of M_I such that $\bigwedge k \in I(\ell) \,.\, (M_{k\ell}, v{\uparrow}k\ell \models \neg r_\ell \wedge AFr_\ell)$ for some $\ell \in dom(I)$ and $r_\ell \in \mathcal{L}(\mathcal{AP}_\ell, \neg, \wedge)$, then

$$M_I, v \models_\Phi AFex_\ell.$$

PROOF. We prove $\pi \models Fex_\ell$ for π an arbitrary fullpath in M_I such that v is the first state of π and $\pi \models \Phi$. By the definition of $\models_\Phi$, this establishes $M_I, v \models_\Phi AFex_\ell$. Now $W_I(s)$ is supercycle-free for every reachable I-state s, by assumption. Hence, by the deadlock freedom theorem (6.5.5.2), we have $M_I, S^0_I \models AGEXtrue$. Therefore, π is infinite. Also, by propositional and temporal logic reasoning (and using $nblk_i \equiv \neg blk_i$)

$$\pi \models \bigwedge i \in dom(I) \,.\, (\overset{\infty}{G} nblk_i \vee (\overset{\infty}{F} nblk_i \wedge \overset{\infty}{F} blk_i) \vee \overset{\infty}{G} blk_i).$$

Hence we can partition $dom(I)$ into three sets $\psi_{nblk}, \psi_{ex}, \psi_{blk}$ such that

$$\pi \models \bigwedge i \in \psi_{nblk} \,.\, \overset{\infty}{G} nblk_i$$
$$\pi \models \bigwedge i \in \psi_{ex} \,.\, (\overset{\infty}{F} nblk_i \wedge \overset{\infty}{F} blk_i)$$
$$\pi \models \bigwedge i \in \psi_{blk} \,.\, \overset{\infty}{G} blk_i.$$

So, by CTL* semantics, π has a suffix ρ such that

$$\rho \models \bigwedge i \in \psi_{nblk} \,.\, Gnblk_i \tag{a}$$
$$\rho \models \bigwedge i \in \psi_{ex} \,.\, (\overset{\infty}{F} nblk_i \wedge \overset{\infty}{F} blk_i) \tag{b}$$
$$\rho \models \bigwedge i \in \psi_{blk} \,.\, Gblk_i. \tag{c}$$

We now have three cases, depending on which of $\psi_{nblk}, \psi_{ex}, \psi_{blk}$ contains ℓ.

Case 1: $\ell \in \psi_{nblk}$. Since $\bigwedge k \in I(\ell) \,.\, (M_{k\ell}, v{\uparrow}k\ell \models \neg r_\ell \wedge AFr_\ell)$ by assumption, $v{\uparrow}\ell$ is sometimes-blocking by the sometimes-blocking proposition (6.7.3.4). Thus

$v \models blk_\ell$. Since $\rho \models nblk_\ell$, P_ℓ must have changed state along π, because v is the first state of π, and ρ is a suffix of π (remember that blk_ℓ is a purely propositional formula). By the I-structure definition (5.3.1), P_ℓ must have been executed along π. Hence $\pi \models Fex_\ell$. (End of case 1.)

Case 2: $\ell \in \psi_{ex}$. Since $\pi \models \overset{\infty}{F} nblk_\ell \wedge \overset{\infty}{F} blk_\ell$, P_ℓ must change state infinitely often along π. Hence by the I-structure definition (5.3.1), P_ℓ must be executed infinitely often along π. Thus $\pi \models \overset{\infty}{F} ex_\ell$, which implies $\pi \models Fex_\ell$. (End of case 2.)

Case 3: $\ell \in \psi_{blk}$. First, a few definitions are needed. A subset ψ of $dom(I)$ is I-*connected* if and only if $\bigwedge i, j \in \psi, i \neq j \,.\, (i\, I^+\, j)$ where I^+ is the transitive closure of I. An I-process P_k *borders* ψ if and only if $k \in I(i) - \psi$ for some $i \in \psi$. We let $border(\psi)$ denote the set of all bordering I-processes of ψ. Let η be a maximal I-connected subset of ψ_{blk}. We call η a blk-region. Consider the I-processes that border η. Clearly, no such I-process can be in ψ_{blk}, as η would not be maximal. Hence, every bordering I-process must be in ψ_{nblk} or in ψ_{ex}. It is clear that ψ_{blk} can be partitioned into (I-disconnected) blk-regions. Let θ be a maximal I-connected subset of $dom(I)$ such that $\pi \models \bigwedge i \in \theta \,.\, (\overset{\infty}{F} ex_i)$. We call θ an inf-region. For an I-process P_i such that $\pi \models \overset{\infty}{F} ex_i$, we define $inf(i)$ to be the inf-region which P_i is a member of. For ψ an arbitrary subset of $dom(I)$, if wait-for-graph $W_I(s)$ contains an edge $a_i^I \longrightarrow P_j^I$ such that $i \in \psi$ and $j \notin \psi$, then we say that $W_I(s)$ contains an edge *out of* ψ.

We now establish a series of assertions P2, P3, P4, which together allow us to establish $\pi \models Fex_\ell$.

> Let P_j border some inf-region θ. Then there exists a suffix ρ' of ρ such that, for every state s along ρ',
> $W_I(s)$ contains no edges into P_j. (P2)

Proof of P2. Since P_j borders inf-region θ, there exists $k \in \theta$ such that $j\, I\, k$. Since $j \notin \theta$, we have $\rho \models \overset{\infty}{G} \neg ex_j$, as otherwise θ would not be maximal. Thus there exists a suffix ρ'' of ρ such that

$$\rho'' \models (\overset{\infty}{F} ex_k \wedge G \neg ex_j).$$

Now consider $\rho''{\uparrow}jk$. By the path projection definition (5.1.2.1) and $\rho'' \models (\overset{\infty}{F} ex_k \wedge G \neg ex_j)$, we have $\rho''{\uparrow}jk \models (\overset{\infty}{F} ex_k \wedge G \neg ex_j)$. Thus, by the path-mapping lemma (6.4.3), $\rho''{\uparrow}jk$ is a fullpath in M_{jk}. Also, $\rho''{\uparrow}jk \models Gex_k$, since all transitions of $\rho''{\uparrow}jk$ are either P_k^j-transitions or P_j^k-transitions. Hence

$$M_{jk}, s{\uparrow}jk \models EGex_k, \tag{a}$$

where s is an arbitrary state along ρ''. Note that s is a reachable state in M_I, since it is reachable from v, and v is, by assumption, a reachable state in M_I. Now let i be an arbitrary element of $I(j)$, and let $J = \{\{i, j\}, \{j, k\}\}$. By the state-mapping corollary (6.4.5), $s{\uparrow}J$ is reachable in M_J. Since $(s{\uparrow}J){\uparrow}jk = s{\uparrow}jk$, we have, by (a)

$$M_{jk}, (s{\uparrow}J){\uparrow}jk \models EGex_k. \tag{b}$$

Letting $s_J = s{\uparrow}J$ in LV (Definition 6.7.2.1), we get from (b)

$$\bigwedge a_i^J \,.\, (a_i^J \longrightarrow P_j^J \notin W_J(s{\uparrow}J)), \tag{c}$$

so there is no edge in $W_J(s{\uparrow}J)$ from i to j. Since $i\,I\,j$ and $j\,I\,k$, we have $J \subseteq I$. So, by applying the wait-for-graph projection proposition (6.5.4.1) to (c), we obtain $\bigwedge a_i^I \,.\, (a_i^I \longrightarrow P_j^I \notin W_I(s))$. Since i is an arbitrarily chosen element of $I(j)$, we conclude that $W_I(s)$ contains no edges into P_j^I. As s is an arbitrary state along ρ'', setting ρ' to ρ'' concludes the proof of P2. (End proof of P2.)

> Let ψ be an arbitrary subset of ψ_{blk}. If there exists a suffix ρ' of ρ such that,
>
> for every state s along ρ', $W_I(s)$ contains no edge out of ψ,
>
> then,
>
> $$\rho \models \bigvee i \in \psi \,.\, \overset{\infty}{F} ex_i. \tag{P3}$$

Proof of P3. Assume otherwise. Thus there exists a suffix ρ'' of ρ such that $\rho'' \models \bigwedge i \in \psi \,.\, G\neg ex_i$. Let s be an arbitrary state along ρ''. By assumption, $W_I(s)$ is supercycle-free. Hence $W_I(s){\uparrow}\psi$ is supercycle-free. Thus, by the supercycle proposition (6.5.5.1) with $I := I{\uparrow}\psi$, some move node $a_{i_s}^I$ such that $i_s \in \psi$, must have no outgoing edges in $W_I(s){\uparrow}\psi$. By assumption, $W_I(s)$ contains no edge from a move in ψ to an I-process outside ψ. Thus $a_{i_s}^I$ must have no outgoing edges in $W_I(s)$. Hence, by Observation 6.5.2.2, $s(a_{i_s}^I.guard) = true$. So, by the compact I-structure definition (5.4.3), $a_{i_s}^I$ can be executed, and thus $M_I, s \models en_{i_s}$.

Now let t be the successor state to s along ρ''. By assumption, $W_I(t)$ contains no edge out of ψ. Also, $W_I(t){\uparrow}\psi = W_I(s){\uparrow}\psi$ since no I-process in ψ is executed along ρ''. Thus $W_I(t)$ contains no edges out of $a_{i_s}^I$. Hence, by Observation 6.5.2.2, $t(a_{i_s}^I.guard) = true$. We can inductively repeat this argument (e.g., for the successor state to t, and then the successor state to that state, etc...) to conclude that $u(a_{i_s}^I.guard) = true$ for every state u which occurs after s along ρ''. Thus $\rho'' \models \overset{\infty}{G} en_{i_s}$.

Now $\rho'' \models \overset{\infty}{G} blk_{i_s}$, since $i_s \in \psi$ and $\psi \subseteq \psi_{blk}$. Since $\rho'' \models G\neg ex_{i_s}$, we have $\rho'' \models \overset{\infty}{G}(blk_{i_s} \wedge en_{i_s}) \wedge G\neg ex_{i_s}$. Hence, by the weak blocking fairness definition (6.7.3.3), $\rho'' \models \neg\Phi$. Hence $\pi \models \neg\Phi$, since ρ'' is a suffix of π. But π was chosen so that $\pi \models \Phi$. Hence the original assumption is false, and P3 is established. (End proof of P3.)

> Let j be an arbitrary element of ψ_{blk}. Then either
>
> P_j is a member of some *inf*-region, or
>
> P_j borders some *inf*-region. (P4)

Proof of P4. Assume otherwise. Since $j \in \psi_{blk}$, we have $j \in \eta$ for some blk-region η. Hence

$$\zeta = \eta - \left(\bigcup\nolimits_{\theta \text{ is an } inf\text{-region}} (\theta \cup border(\theta))\right)$$

is nonempty, since $j \notin \theta \cup border(\theta)$ for any *inf*-region θ by assumption, and thus $j \in \zeta$. By (P2), we have that there exists a suffix ρ' of ρ such that

for any I-process P_k which borders an *inf*-region,
for every state s along ρ',
$W_I(s)$ contains no edges into P_k. (a)

Also, if for a state s and an I-process P_k, $s \models nblk_k$, then $W_I(s)$ contains no edge into P_k (see Observation 6.7.3.2). Thus, by definition of ψ_{nblk} we have

for every state s along ρ',
$W_I(s)$ contains no edge into any I-process P_k in ψ_{nblk}. (b)

Now consider an arbitrary member P_k of $border(\zeta)$. Since every I-process is a member of exactly one of ψ_{blk}, ψ_{ex}, ψ_{nblk}, we have three (sub-)cases.

$k \in \psi_{blk}$. Since $\zeta \subseteq \eta$, and $k \in border(\zeta)$, we have $k \in \eta$, since η is a *blk*-region, (and $k \in \psi_{blk}$ by the case assumption) and all *blk*-regions are maximal, by definition. By assumption, $k \in border(\zeta)$, so $k \notin \zeta$ by definition of *border*. Therefore, $k \in \eta - \zeta$. Since $\zeta = \eta - (\bigcup_{\theta \text{ is an } inf\text{-region}} (\theta \cup border(\theta)))$ by definition, we have $k \in \theta \cup border(\theta)$ for some *inf*-region θ. Since $k \in border(\zeta)$, there exists P_m such that $k\, I\, m$ and $m \in \zeta$. However, if $k \in \theta$, then $m \in border(\theta)$, contrary to the definition of ζ. Thus we conclude that $k \in border(\theta)$.

$k \in \psi_{ex}$. Therefore $\pi \models \overset{\infty}{F} ex_k$ (see case 2). So k is a member of some *inf*-region θ, by definition of *inf*-region. Since $k \in border(\zeta)$, there exists m such that $k\, I\, m$ and $m \in \zeta$. Thus $m \in border(\theta)$, contrary to the definition of ζ. We conclude that k cannot be a member of any *inf*-region, and hence k cannot be a member of ψ_{ex}.

$k \in \psi_{nblk}$. We do not need to infer anything for this case other than $k \in \psi_{nblk}$.

Considering the above three (sub-)cases, we have shown that every member of $border(\zeta)$ either borders some *inf*-region or is a member of ψ_{nblk}. Therefore, in $W_I(s)$, every edge out of ζ must have a target P_k such that P_k borders an *inf*-region or such that P_k is in ψ_{nblk}. Thus, by (a) and (b), we conclude that

for every state s along ρ', $W_I(s)$ contains no edge out of ζ.

Since $\zeta \subseteq \eta$, ζ is a subset of ψ_{blk} by definition of *blk*-region. Thus, by (P3), $\rho' \models \overset{\infty}{F} ex_{k'}$ for some $P_{k'}$ in ζ. Hence $k' \in \theta$ for some *inf*-region θ. But this contradicts $k' \in \zeta$, by definition of ζ. Hence the original assumption is false, and P4 is established. (End proof of P4.)

We can now establish $\pi \models Fex_\ell$. By the case 3 assumption, $\ell \in \psi_{blk}$. By (P4), we have

P_ℓ is a member of some *inf*-region θ or P_ℓ borders some *inf*-region θ.

If P_ℓ is a member of θ, then $\pi \models \overset{\infty}{F} ex_\ell$, and therefore $\pi \models Fex_\ell$. If P_ℓ borders θ, then there exists $j \in I(\ell) \cap \theta$. Hence $\pi \models \overset{\infty}{F} ex_j$, and since π is an infinite fullpath, we conclude by the path projection definition (5.1.2.1) that $\pi{\uparrow}\ell j$ is an infinite path (and therefore is a fullpath) and $\pi{\uparrow}\ell j \models \overset{\infty}{F} ex_j$. Moreover, the first state of $\pi{\uparrow}\ell j$ is $v{\uparrow}\ell j$. Since $\bigwedge k \in I(\ell) \,.\, (M_{k\ell}, v{\uparrow}k\ell \models \neg r_\ell \wedge AFr_\ell)$ by assumption (3) of the lemma, and $j \in I(\ell)$, we have $M_{\ell j}, v{\uparrow}\ell j \models \neg r_\ell \wedge AFr_\ell$. Thus, $M_{\ell j}, v{\uparrow}\ell j \models AFex_\ell$, and so

$\pi{\uparrow}\ell j \models Fex_\ell$ since $\pi{\uparrow}\ell j$ is a fullpath and the first state of $\pi{\uparrow}\ell j$ is $v{\uparrow}\ell j$. Hence, by the path projection definition (5.1.2.1), we conclude $\pi \models Fex_\ell$. (End of case 3.)

Since we have established $\pi \models Fex_\ell$ in all three cases, the lemma is established. $\Box$

C.6 The Generalized Large Model Theorem

Theorem 6.8.1 (Generalized Large Model). *Let $f_{k\ell}$ be an arbitrary formula of $FLCTL_{k\ell}$ (Definition 6.6.1). Let s be an arbitrary reachable I-state, and, for all i, j such that $i\,I\,j$, let $s_{ij} = s{\uparrow}ij$. If the liveness assumption LV holds, and $W_I(u)$ is supercycle-free for every reachable I-state u, then*

$$\bigwedge(i,j) \in I \,.\, (M_{ij}, s_{ij} \models \textstyle\bigwedge_{k\ell} f_{k\ell}) \textit{ implies } M_I, s \models_\Phi \textstyle\bigwedge_{k\ell} f_{k\ell}.$$

Proof. By MPCTL* semantics, $M_{ij}, s_{ij} \models \bigwedge_{k\ell} f_{k\ell}$ is equivalent to $M_{ij}, s_{ij} \models f_{ij}$. Also by MPCTL* semantics, $M_I, s \models_\Phi \bigwedge_{k\ell} f_{k\ell}$ is equivalent to $M_I, s \models_\Phi \bigwedge(i,j) \in I \,.\, (f_{ij})$. This is equivalent, by CTL* semantics, to $\bigwedge(i,j) \in I \,.\, (M_I, s \models_\Phi f_{ij})$. Hence, the generalized large-model theorem is established if we can prove that

$$\bigwedge(i,j) \in I \,.\, (M_{ij}, s_{ij} \models f_{ij}) \text{ implies } \bigwedge(i,j) \in I \,.\, (M_I, s \models_\Phi f_{ij}) \qquad (*)$$

given the assumptions of the generalized large-model theorem. The proof is by induction on the structure of f_{ij}.

The proofs of the cases $f_{ij} = h_{ij}$, $f_{ij} = f'_{ij} \wedge f''_{ij}$, $f_{ij} = AGh_{ij}$, $f_{ij} = AG(p_i \Rightarrow AY_i q_i)$, $f_{ij} = AG(a_i \Rightarrow EX_i b_i)$, are verbatim identical to the proofs for the same cases respectively in the large-model theorem (6.6.2), and are thereby omitted here. Note that only the proof for the remaining case of $f_{ij} = AG(p_i \Rightarrow A[q_i U r_i])$ in Theorem 6.6.2 appealed to the assumptions $I = \{i_1, \ldots, i_K\} \times \{i_1, \ldots, i_K\} - \{(i,i) \mid i \in \{i_1, \ldots, i_K\}\}$ and $M_I, S_I^0 \models AGEXtrue$ in the antecedent of Theorem 6.6.2. We now give the proof for $f_{ij} = AG(p_i \Rightarrow A[q_i U r_i])$ in the generalized case.

$f_{ij} = AG(p_i \Rightarrow A[q_i U r_i])$. We will establish $M_I, t \models_\Phi (p_i \Rightarrow A[q_i U r_i])$ where t is an arbitrary s-reachable state in M_I. If $M_I, t \models \neg p_i$ then $M_I, t \models_\Phi (p_i \Rightarrow A[q_i U r_i])$, and we are done. Otherwise $M_I, t \models p_i$, and we must establish $M_I, t \models_\Phi A[q_i U r_i]$. If $M_I, t \models r_i$ then we are done. Otherwise $M_I, t \models \neg r_i$, so $t{\uparrow}ij \models \neg r_i$ by the I-state projection proposition (6.3.1). Let π be an arbitrary fullpath of M_I starting in t such that $\pi \models \Phi$. The antecedent is

$$M_{ij}, s_{ij} \models AG(p_i \Rightarrow A[q_i U r_i]). \qquad \text{(a)}$$

By assumption, $s{\uparrow}ij = s_{ij}$. Thus by the relativized state-mapping corollary (6.4.6) with $J = \{\{i,j\}\}$, $t{\uparrow}ij$ is a s_{ij}-reachable state in M_{ij}. So, by (a) we have

$$M_{ij}, t{\uparrow}ij \models (p_i \Rightarrow A[q_i U r_i]). \qquad \text{(b)}$$

Since $M_I, t \models p_i$, we conclude $M_{ij}, t{\uparrow}ij \models p_i$ by the I-state projection proposition (6.3.1). Together with (b), this yields

$$M_{ij}, t{\uparrow}ij \models A[q_i U r_i], \qquad \text{(c)}$$

and since $t{\uparrow}ij \models \neg r_i$, we have by (c) and CTL* semantics

$$M_{ij}, t{\uparrow}ij \models \neg r_i \wedge AFr_i. \qquad \text{(d)}$$

By the path-mapping lemma (6.4.3), $\pi{\upharpoonright}ij$ is a path in M_{ij}. Also, $\pi{\upharpoonright}ij$ starts in $t{\upharpoonright}ij$, and therefore, by (c) and CTL* semantics, we have $M_{ij}, \pi{\upharpoonright}ij \models [q_i U_w r_i]$ (we cannot conclude $M_{ij}, \pi{\upharpoonright}ij \models [q_i U r_i]$ because we have not shown that $\pi{\upharpoonright}ij$ is a fullpath). By the path projection lemma (6.3.3) with $J = \{\{i,j\}\}$, we have $M_I, \pi \models [q_i U_w r_i]$. It remains for us to establish $M_I, \pi \models F r_i$.

Since j ranges over $I(i)$, we have, from (d), $\bigwedge j \in I(i) . (M_{ij}, t{\upharpoonright}ij \models \neg r_i \wedge AF r_i)$. By assumption, s is reachable, and t is s-reachable, so t is reachable. Together with the assumptions of the theorem, we have satisfied the antecedent of the progress lemma (6.7.4.1) for $v = t$ and $\ell = i$. Thus, by the progress lemma (6.7.4.1), we have $M_I, t \models_\Phi AFex_i$. Therefore, P_i^I is eventually executed along π. By repeating this argument inductively, we conclude that either P_i^I is executed repeatedly along π until a global state t' (along π) is reached such that $M_I, t' \models r_i$, (and hence $M_I, \pi \models F r_i$), or P_i^I is executed infinitely often along π (since $\neg r_i \wedge AF r_i$ continues to hold, and therefore the progress lemma can be applied repeatedly). In the latter case, we have that π contains an infinite number of P_i^I-transitions, so by definition of path projection, $\pi{\upharpoonright}ij$ contains an infinite number of P_i^j-transitions, so $\pi{\upharpoonright}ij$ is a fullpath. Since $\pi{\upharpoonright}ij$ is a fullpath starting in $t{\upharpoonright}ij$, we have, by (c), $M_{ij}, \pi{\upharpoonright}ij \models [q_i U r_i]$. By the path projection lemma (6.3.3), we have $M_I, \pi \models [q_i U r_i]$, which implies $M_I, \pi \models F r_i$.

Since $M_I, \pi \models F r_i$ in both cases, and $M_I, \pi \models [q_i U_w r_i]$ has been established above, we have $M_I, \pi \models [q_i U r_i]$. Since π is an arbitrary fullpath starting in t such that $\pi \models \Phi$, we conclude $M_I, t \models_\Phi A[q_i U r_i]$. Hence $M_I, t \models_\Phi (p_i \Rightarrow A[q_i U r_i])$. Since t is an arbitrary s-reachable state in M_I, we conclude $M_I, s \models_\Phi AG(p_i \Rightarrow A[q_i U r_i])$. Since i ranges over $\{i_1, \ldots, i_K\}$, and j ranges over $I(i)$, we have $\bigwedge (i,j) \in I . (M_I, s \models_\Phi AG(p_i \Rightarrow A[q_i U r_i]))$. Thus, we have established (*), which concludes the proof of the generalized large-model theorem. □

D. A COMPACT REPRESENTATION FOR SYNCHRONIZATION SKELETONS

In this appendix, we provide a full discussion of the compact representation introduced in Section 5.4. Our discussion here is self-contained and so repeats some of the material in Section 5.4 (in particular, the definition of compact MP-synthesis is repeated, and so retains the number, 5.4.1, that it has in the main text).

Suppose P_i^j (for every $j \in I(i)$) contains two arcs from i-state s_i to i-state s_i', e.g., $a_{i,1}^j = (s_i, B_{i,1}^j \to A_{i,1}^j, s_i')$ and $a_{i,2}^j = (s_i, B_{i,2}^j \to A_{i,2}^j, s_i')$. Then, by the MP-synthesis definition (5.1.1), P_i^I contains $2^{|I(i)|}$ arcs from i-state s_i to i-state s_i', one arc for each element of the cartesian product

$$\{a_{i,1}^{j_1}, a_{i,2}^{j_1}\} \times \cdots \times \{a_{i,1}^{j_n}, a_{i,2}^{j_n}\}$$

where $\{j_1, \ldots, j_n\} = I(i)$. Thus P_i^I is exponentially large in K $(= |dom(I)|)$ in the worst case (since $|I(i)| = K - 1$ when $i\, I\, j$ for every j in $dom(I) - \{i\}$), which defeats the purpose of MP-synthesis. We deal with this by defining a compact representation for processes in which there is at most one arc between any pair of (local) states, thereby avoiding the exponential blowup illustrated above.

Consider a pair-process P_i^j which has two arcs from state s_i to state s_i', labeled with the synchronization commands $B_{i,1}^j \to A_{i,1}^j$, $B_{i,2}^j \to A_{i,2}^j$. In compact notation, we replace these two arcs by a single arc whose label is $B_{i,1}^j \to A_{i,1}^j \oplus B_{i,2}^j \to A_{i,2}^j$.

The symbol $\oplus$ is a binary operator which takes a pair of guarded commands as arguments. It is defined as follows:

$$(B_1 \rightarrow A_1) \oplus (B_2 \rightarrow A_2) \stackrel{\text{df}}{=} B_1 \ \vee \ B_2 \rightarrow \begin{array}{l} \textbf{if } B_1 \rightarrow A_1 \\ \Box \ B_2 \rightarrow A_2 \\ \textbf{fi} \end{array}$$

Roughly, the operational semantics of $B^j_{i,1} \rightarrow A^j_{i,1} \oplus B^j_{i,2} \rightarrow A^j_{i,2}$ is that if one of the guards $B^j_{i,1}, B^j_{i,2}$ evaluates to true, then the corresponding body $A^j_{i,1}, A^j_{i,2}$, respectively, can be executed. If neither $B^j_{i,1}$ nor $B^j_{i,2}$ evaluates to true, then the command "blocks," i.e., waits until one of $B^j_{i,1}, B^j_{i,2}$ evaluates to true. Note that the guarded commands which we use as arc labels are, in general, *partial commands*, i.e., they cannot be executed in every global state. A full treatment of the calculus of partial guarded commands is beyond our scope here. The reader is referred to Nelson [1989]. It is easily seen that $\oplus$ is commutative. To see that it is also associative, we note that

$$(B_1 \rightarrow A_1 \oplus B_2 \rightarrow A_2) \oplus B_3 \rightarrow A_3$$

and

$$B_1 \rightarrow A_1 \oplus (B_2 \rightarrow A_2 \oplus B_3 \rightarrow A_3)$$

have the same semantics, namely, if one of B_1, B_2, B_3 is true, then the corresponding body can be executed, and if none of B_1, B_2, B_3 are true, then the command blocks. Since $\oplus$ is commutative and associative, it can be extended to n arguments using the indexed notation $\oplus_{\ell \in [1:n]}$. Thus, if P^j_i contains n arcs from i-state s_i to i-state s'_i, with labels $B^j_{i,1} \rightarrow A^j_{i,1}, \ldots, B^j_{i,n} \rightarrow A^j_{i,n}$, then these n arcs can be replaced by a single arc whose label is $\oplus_{\ell \in [1:n]} B^j_{i,\ell} \rightarrow A^j_{i,\ell}$.

We call an arc whose label has the form $\oplus_{\ell \in [1:n]} B^j_{i,\ell} \rightarrow A^j_{i,\ell}$ a *(pair) move*. In compact notation, a pair-process has at most one move between any pair of local states. The translation from compact notation back to normal notation is straightforward: simply replace every move by the corresponding set of arcs. The operational semantics is as follows. Assume that the current state is $(s_1, \ldots, s_i, \ldots, s_K, x_1, \ldots, x_m)$, and that P_i contains a move from s_i to s'_i labeled by $\oplus_{\ell \in [1:n]} B^j_{i,\ell} \rightarrow A^j_{i,\ell}$. If a guard $B^j_{i,\ell}$ (for some ℓ in $[1 : n]$) evaluates to true in the current state, then $(s_1, \ldots, s'_i, \ldots, s_K, x'_1, \ldots, x'_m)$ is a permissible next-state where $x'_1, \ldots, x'_m$ is a list of updated shared variables resulting from action $A^j_{i,\ell}$. A computation path is an infinite sequence of states where successive pairs of states are related by the above next-state relation. As before, omission of the guard $B^j_{i,\ell}$ from a guarded command means that $B^j_{i,\ell}$ is interpreted as *true*, and we write the command as $A^j_{i,\ell}$, while omission of the action $A^j_{i,\ell}$ from a guarded command means that the shared variables are unaltered, and we write the command as $B^j_{i,\ell}$. This is formalized by the following definition, which is a consequence of the pair structure definition (5.2.1) and the translation between compact and normal notation given above.

Definition 5.4.2 (*Compact Pair-Structure*). Let $i\,I\,j$. The semantics of $(S^0_{ij}, P^j_i \| P^i_j)$ in compact notation is given by the pair-structure $M_{ij} = (S^0_{ij}, S_{ij}, R_{ij})$ where

(1) S_{ij} is a set of ij-states,

(2) $S^0_{ij} \subseteq S_{ij}$ gives the initial states of $P^j_i \| P^i_j$, and

(3) $R_{ij} \subseteq S_{ij} \times \{i,j\} \times S_{ij}$ is a transition relation giving the transitions of $P^j_i \| P^i_j$. A transition (s_{ij}, h, t_{ij}) by $P^{\bar{h}}_h$ is in R_{ij} if and only if
   (a) $h \in \{i,j\}$,
   (b) s_{ij} and t_{ij} are ij-states, and
   (c) there exists a move $(s_{ij}{\uparrow}h, \oplus_{\ell\in[1:n]} B^{\bar{h}}_{h,\ell} \rightarrow A^{\bar{h}}_{h,\ell}, t_{ij}{\uparrow}h)$ in $P^{\bar{h}}_h$ such that there exists $m \in [1:n]$:
      (i) $s_{ij}(B^{\bar{h}}_{h,m}) = true$,
      (ii) $< s_{ij}{\uparrow}\mathcal{SH}_{ij} > A^{\bar{h}}_{h,m} < t_{ij}{\uparrow}\mathcal{SH}_{ij} >$, and
      (iii) $s_{ij}{\uparrow}\bar{h} = t_{ij}{\uparrow}\bar{h}$.

   Here $\bar{h} = i$ if $h = j$ and $\bar{h} = j$ if $h = i$.

Now an I-process P^I_i is derived by "composing" all the moves of P^j_i (as j varies over $I(i)$) which have the same start and end states. Toward this end, we define the binary composition operator $\otimes$ on guarded commands. We say that a guarded command is *simple* iff it has the form $B \rightarrow A$ where B is a guard and A is a parallel assignment statement. Applied to a pair of simple guarded commands, $\otimes$ returns a simple guarded command whose guard is the conjunction of the guards of the operands and whose assignment statement is the parallel composition of the assignment statements of the operands.

Definition D.1 (*Simple Guarded-Command Composition*). The composition of two simple guarded commands is given by

$$(B_1 \rightarrow A_1) \otimes (B_2 \rightarrow A_2) \;=\; B_1 \wedge B_2 \rightarrow A_1 // A_2.$$

That is, if both guards are true, then both bodies can be executed in parallel. We note that, in general, it is possible that A_1, A_2 have a variable in common on their left-hand sides. Since $\otimes$ is a *syntactic* operator, this presents no problems in the definition of $\otimes$. However, the semantics of $A_1 // A_2$ is problematic in general. But, the only time when we need to assign a semantics to guarded commands containing $\otimes$ is in Definition 5.4.3 below, and we shall see that in this case, the possibility of assigning twice to the same variable within a parallel assignment does not arise. It is clear that $\otimes$, when applied to simple guarded commands, is commutative and associative, since both $\wedge$ and $//$ are. We now define general guarded commands as those that can be built up from simple guarded commands by applying $\oplus$ and $\otimes$.

Definition D.2 (*General Guarded Command*). General guarded commands are inductively defined as follows.

(1) A simple guarded command is a general guarded command

(2) If G_1, G_2 are general guarded commands, then so are $G_1 \oplus G_2$ and $G_1 \otimes G_2$

(3) The only general guarded commands are those that are generated by rules (1) and (2) above

In order to define the operational semantics of a general guarded command, we introduce a *normal form* for general guarded commands.

Definition D.3 (*Normal Form*). A general guarded command is in normal form iff it has the form $\oplus_{\ell \in [1:n]} G_\ell$, where each G_ℓ is a simple guarded command.

We note that the operational semantics of normal forms has been described informally above. The compact pair-structure definition (5.4.2) formalizes the operational semantics of normal forms by defining the pair-structures that are generated by pair-processes expressed in compact notation (whose arc labels are general guarded commands expressed in normal form). We can now define the application of $\otimes$ to general guarded commands in normal form as follows.

Definition D.4 (*Normal Form Composition*). The composition of two general guarded commands in normal form is given by

$$(\oplus_{\ell \in \varphi} G_\ell) \ \otimes \ (\oplus_{k \in \psi} G_k) \quad = \quad \oplus_{\ell \in \varphi, k \in \psi}(G_\ell \otimes G_k).$$

We recall that $G_\ell \otimes G_k$ is given by the simple guarded-command composition definition (D.1), since G_ℓ, G_k are simple guarded commands.

It remains to show that every guarded command can be expressed in normal form.

Proposition D.5 (Normal Form). *Every general guarded command can be expressed in normal form.*

Proof. By structural induction over the definition of a general guarded command G.

Base Case. G is simple. Then $G = \oplus_{\ell \in \varphi} G_\ell$ where φ is a singleton set. This is in normal form.

Induction Step. $G = G_1 \oplus G_2$. By the induction hypothesis, G_1 and G_2 are in normal form. Let $G_1 = \oplus_{\ell \in \varphi} G_\ell$, $G_2 = \oplus_{k \in \psi} G_k$. Hence G_ℓ for all $\ell \in \varphi$ and G_k for all $k \in \psi$ are simple guarded commands, by the normal-form definition (D.3). Since $\oplus$ is associative and commutative over simple guarded commands, we have $G_1 \oplus G_2 = (\oplus_{\ell \in \varphi} G_\ell) \oplus (\oplus_{k \in \psi} G_k) = (\oplus_{m \in \varphi \cup \psi} G_m)$. This last expression is in normal form.

Induction Step. $G = G_1 \otimes G_2$. By the induction hypothesis, G_1 and G_2 are in normal form. Let $G_1 = \oplus_{\ell \in \varphi} G_\ell$, $G_2 = \oplus_{k \in \psi} G_k$. Hence, by the normal-form composition definition (D.4), $G_1 \otimes G_2 = \oplus_{\ell \in \varphi, k \in \psi}(G_\ell \otimes G_k)$. Now G_ℓ for all $\ell \in \varphi$ and G_k for all $k \in \psi$ are simple guarded commands, by the normal-form definition (D.3). Thus, by the simple guarded-command composition definition (D.1), $G_\ell \otimes G_k$ can be rewritten as a simple guarded command. Thus by the normal-form definition (D.3), $G_1 \otimes G_2$ can be expressed in normal form. □

We finally recast the MP-synthesis definition (5.1.1) into compact form as follows.

Definition 5.4.1 (*Compact MP-Synthesis*). A *compact I-process* P_i^I is derived from the compact pair-process P_i^j as follows:

P_i^I contains a move from s_i to t_i with label $\otimes_{j \in I(i)} \oplus_{\ell \in [1:n]} B_{i,\ell}^j \rightarrow A_{i,\ell}^j$
iff
for every j in $I(i)$: P_i^j contains a move from s_i to t_i with label $\oplus_{\ell \in [1:n]} B_{i,\ell}^j \rightarrow A_{i,\ell}^j$.

The *initial state set* S_I^0 of the I-system is derived from the initial state S_{ij}^0 of the pair-system as follows:

$$S_I^0 = \{s \mid \bigwedge (i,j) \in I \,.\, (s{\uparrow}ij \in S_{ij}^0)\}.$$

Thus an I-process in compact notation has at most one move of the form $(s_i, \otimes_{j \in I(i)} \oplus_{\ell \in [1:n]} B_{i,\ell}^j \rightarrow A_{i,\ell}^j, t_i)$ between any pair of i-states s_i, t_i. Note that a move in a pair-process is simply a special case of a move in an I-process when I is a single pair, e.g., if $I = \{\{i,k\}\}$, then $I(i)$ in $\otimes_{j \in I(i)}$ expands to the singleton $\{k\}$ giving a move in the pair-process P_i^k. In the sequel, we shall use a_i^j, a_i^I to denote moves in P_i^j, P_i^I, respectively.

Returning to the example given at the beginning of this section, the pair of arcs $(s_i, B_{i,1}^j \rightarrow A_{i,1}^j, s_i')$, $(s_i, B_{i,2}^j \rightarrow A_{i,2}^j, s_i')$ in P_i^j are replaced by the single move $(s_i, (B_{i,1}^j \rightarrow A_{i,1}^j \oplus B_{i,2}^j \rightarrow A_{i,2}^j),\ s_i')$, and the $2^{|I(i)|}$ arcs in P_i^I are replaced by the single move $(s_i,\ \otimes_{j \in I(i)} (B_{i,1}^j \rightarrow A_{i,1}^j \oplus B_{i,2}^j \rightarrow A_{i,2}^j),\ s_i')$.

The translation of I-processes from normal to compact notation is most easily achieved via the translation given above for pair-processes, i.e., derive the corresponding pair-process (essentially applying MP-synthesis "in reverse"), perform the translation to obtain the compact pair-processes, and then derive the compact I-process using the compact MP-synthesis definition (5.4.1). Translating from compact to normal notation is achieved by applying the definition of the $\otimes$ operator, in effect expanding the general guarded command $\otimes_{j \in I(i)} \oplus_{\ell \in [1:n]} B_\ell \rightarrow A_\ell$ into normal form, and then creating one arc for each simple guarded command in the normal form. For example, if $I = \{\{i,k\},\{k,\ell\},\{\ell,i\}\}$, then

$$\otimes_{j \in I(i)} (T_j \rightarrow x_{ij} := j \oplus N_j \ \vee\ C_j \rightarrow \mathit{skip})$$

expands into normal form as follows. First, we expand the bound variable j over its range $\{k,\ell\}$, thereby replacing the indexed form $\otimes_{j \in I(i)}$ by the infix form $\otimes$, to obtain

$$\begin{array}{l}(T_k \rightarrow x_{ik} := k \oplus N_k \ \vee\ C_k \rightarrow \mathit{skip}) \ \otimes \\ (T_\ell \rightarrow x_{i\ell} := \ell \oplus N_\ell \ \vee\ C_\ell \rightarrow \mathit{skip}).\end{array}$$

Now we apply the normal-form composition definition (D.4) to obtain

$$\begin{array}{l}(T_k \rightarrow x_{ik} := k \otimes T_\ell \rightarrow x_{i\ell} := \ell) \ \oplus \\ (N_k \ \vee\ C_k \rightarrow \mathit{skip} \otimes T_\ell \rightarrow x_{i\ell} := \ell) \ \oplus \\ (T_k \rightarrow x_{ik} := k \otimes N_\ell \ \vee\ C_\ell \rightarrow \mathit{skip}) \ \oplus \\ (N_k \ \vee\ C_k \rightarrow \mathit{skip} \otimes N_\ell \ \vee\ C_\ell \rightarrow \mathit{skip}).\end{array}$$

Finally, we apply the simple guarded-command composition definition (D.1) to all four occurrences of $\otimes$, to obtain

$$T_k \wedge T_\ell \rightarrow x_{ik} := k // x_{i\ell} := \ell \ \oplus$$

$$(N_k \ \vee\ C_k) \wedge T_\ell \rightarrow skip//x_{i\ell} := \ell \ \oplus$$
$$T_k \wedge (N_\ell \ \vee\ C_\ell) \rightarrow x_{ik} := k//skip \ \oplus$$
$$(N_k \ \vee\ C_k) \wedge (N_\ell \ \vee\ C_\ell) \rightarrow skip//skip.$$

It is easy to see that if P_i^I had three neighbors instead of two, that the size of the final result would be eight instead of four. Generalizing, we see that if $|I(i)| = m$, then

$$\otimes_{j \in I(i)} \oplus_{\ell \in [1:n]} B^j_{i,\ell} \rightarrow A^j_{i,\ell}$$

expands into a term with size on the order of n^m. Thus the compact form provides an exponential savings in the size of the representation of the I-processes.

The operational semantics of compact I-processes is given by the following definition, which is a consequence of the I-structure definition (5.3.1) and the translation between compact and normal notation given above.

Definition 5.4.3 (*Compact I-Structure*). The semantics of $(S_I^0, P^I_{i_1} \| \dots \| P^I_{i_K})$ in compact notation is given by the I-structure $M_I = (S_I^0, S_I, R_I)$ where

(1) S_I is a set of I-states,

(2) $S_I^0 \subseteq S_I$ gives the initial states of $P^I_{i_1} \| \dots \| P^I_{i_K}$, and

(3) $R_I \subseteq S_I \times dom(I) \times S_I$ is a transition relation giving the transitions of $P^I_{i_1} \| \dots \| P^I_{i_K}$. A transition (s, i, t) by P_i^I is in R_I if and only if
   - (a) $i \in dom(I)$,
   - (b) s and t are I-states, and
   - (c) there exists a move $(s{\uparrow}i, \otimes_{j \in I(i)} \oplus_{\ell \in [1:n]} B^j_{i,\ell} \rightarrow A^j_{i,\ell}, t{\uparrow}i)$ in P_i^I such that all of the following hold:
     - (i) for all j in $I(i)$, there exists $m \in [1:n]$: $s{\uparrow}ij(B^j_{i,m}) = true$ and $< s{\uparrow}\mathcal{SH}_{ij} > A^j_{i,m} < t{\uparrow}\mathcal{SH}_{ij} >$,
     - (ii) for all j in $dom(I) - \{i\}$: $s{\uparrow}j = t{\uparrow}j$, and
     - (iii) for all j, k in $dom(I) - \{i\}$, $j\, I\, k$: $s{\uparrow}\mathcal{SH}_{jk} = t{\uparrow}\mathcal{SH}_{jk}$.

Given a pair-system $(S^0_{ij}, P^j_i \| P^i_j)$ in normal notation, we can apply Definitions 5.2.1, 5.1.1, and 5.3.1 to generate the pair-structure, the I-system, and the I-structure (for the generated I-system) respectively which correspond to $(S^0_{ij}, P^j_i \| P^i_j)$. Alternatively, if $(S^0_{ij}, P^j_i \| P^i_j)$ is given in compact notation, then we can apply Definitions 5.4.2, 5.4.1, and 5.4.3 to generate the pair-structure, the I-system (in compact notation), and the I-structure (for the generated I-system), respectively which correspond to $(S^0_{ij}, P^j_i \| P^i_j)$. It should be apparent that the pair, K-process structures generated by one set of definitions are identical to those generated by the other set of definitions. Thus, any result established in the sequel using one set of definitions will be equally applicable to the other set of definitions. Hence, in establishing a particular result, we will use whichever set of definitions is more convenient at the time. In particular, although some results will be established using normal notation, the implementation of MP-synthesis will be in compact notation, so as to avoid the exponential size I-processes which the normal notation may produce.

E. CHECKING THE WAIT-FOR-GRAPH ASSUMPTION AND THE LIVENESS ASSUMPTION

E.1 Checking the Wait-for-Graph Assumption

The wait-for-graph assumption WG is mechanically checked as follows. As stated in Section 5 we are initially given a pair-system $(S^0_{k\ell}, P^\ell_k \| P^k_\ell)$. For technical convenience, we use the process indices k, ℓ rather than i, j. Using the pair-structure definition (5.2.1) we generate $M^r_{k\ell}$ (recall that a superscript r denotes reachable states/structures—see Section 6.1) from $(S^0_{k\ell}, P^\ell_k \| P^k_\ell)$. We also translate P^ℓ_k, P^k_ℓ to compact notation as shown in Section 5.4. From $M^r_{k\ell}$, we determine the set S^r_k of reachable k-states in $M^r_{k\ell}$.[14] For every $t_k \in S^r_k$, we determine $|t_k.moves|$, using the compact form of P^ℓ_k. Now the wait-for-graph assumption (Definition 6.5.4.2) specifies that J has the form $\{\{j,k\},\{k,\ell_1\},\ldots,\{k,\ell_n\}\}$ where $n = |t_k.moves|$ and $k \notin \{j, \ell_1, \ldots, \ell_n\}$. Since no constraint is placed on the equality of members of $\{j, \ell_1, \ldots, \ell_n\}$, we must consider all possible cases for the form of J. We therefore define:

$$
\begin{aligned}
\mathcal{J}(t_k) \;=\; \{J \mid\; & J = \{\{j,k\},\{k,\ell_1\},\ldots,\{k,\ell_m\}\} \text{ and} \\
& m \in [1:n] \text{ and} \\
& j, k, \ell_1, \ldots, \ell_m \text{ are pairwise distinct}\} \\
\mathcal{J}'(t_k) \;=\; \{J \mid\; & J = \{\{k,\ell_1\},\ldots,\{k,\ell_m\}\} \text{ and} \\
& m \in [1:n] \text{ and} \\
& k, \ell_1, \ldots, \ell_m \text{ are pairwise distinct}\}
\end{aligned}
$$

$\mathcal{J}(t_k) \cup \mathcal{J}'(t_k)$ is the set of all the distinct forms of J that must be considered when checking WG for the k-state t_k, with $\mathcal{J}(t_k)$ containing all the forms of J in which $j \notin \{\ell_1, \ldots, \ell_m\}$, and $\mathcal{J}'(t_k)$ containing all the forms of J in which $j \in \{\ell_1, \ldots, \ell_m\}$. Note that m is really the number of distinct indices in $\{\ell_1, \ldots, \ell_n\}$, so it ranges over $[1:n]$. For every J in $\mathcal{J}(t_k)$, we generate M^r_J, using the compact MP-synthesis definition (5.4.1) and the compact I-structure definition (5.4.3). Then, for every J-state t_J such that $t_J{\upharpoonright}k = t_k$ and $s_J \xrightarrow{k} t_J \in R_J$ for some reachable J-state s_J of M_J, we evaluate

$$\bigwedge a^J_j \,.\, (a^J_j \longrightarrow P^J_k \notin W_J(t_J))$$

or

$$\bigvee a^J_k \in W_J(t_J) \,.\, (\bigwedge \ell \in \{\ell_1, \ldots, \ell_m\} \,.\, (a^J_k \rightarrow P^J_\ell \notin W_J(t_J))).$$

Using the wait-for-graph definition (6.5.2.1) and CTL* semantics, we can rewrite this as follows:

$$
\begin{aligned}
t_J \models \;& (\;\; \bigwedge a^J_j \,.\, (\{\!\!\{a^J_j.start\}\!\!\} \Rightarrow a^J_j.guard_k) \\
& \vee \\
& \quad \bigvee a^J_k \,.\, (\{\!\!\{a^J_k.start\}\!\!\} \wedge (\bigwedge \ell \in \{\ell_1, \ldots, \ell_m\} \,.\, a^J_k.guard_\ell)) \\
&).
\end{aligned} \tag{a}
$$

Since the formula on the right-hand side of the $\models$ in (a) is purely propositional,

[14] This step, in effect, enforces our assumption (made in Section 6.1) that every i-state s_i of P^j_i is reachable in M_{ij}.

it is straightforward to evaluate (a) using the inductive definition for $\models$ supplied in Section 1. If, for any t_k, J, and t_J, (a) evaluates to false, then WG does not hold. Likewise, for every J in $\mathcal{J}'(t_k)$, we generate M^r_J. Then, we set j equal to an arbitrarily chosen member of $\{\ell_1, \ldots, \ell_m\}$. Since J is "radially symmetric" with respect to the $\ell_1, \ldots, \ell_m$ (i.e., J is a "star" with k as the center and $\ell_1, \ldots, \ell_m$ as the points), the particular choice of member of $\{\ell_1, \ldots, \ell_m\}$ makes no difference to the final outcome of the check. As in the case of $J \in \mathcal{J}(t_k)$, we evaluate (a), and if, for any t_k, J and t_J, (a) evaluates to false, then WG does not hold.

If, during the execution of the procedure outlined above, no t_k, J, and t_J are found for which (a) evaluates to false, then WG holds, and we can thus conclude that all I-systems are deadlock-free (provided that $W_I(s^0_I)$ is supercycle-free for every $s^0_I \in S^0_I$). We summarize the procedure given above in Figure 16, and compute its time complexity. The procedure clearly terminates, since the range of all loop variables is finite. Furthermore, upon termination, WG holds if and only if $WGflag$ is set to "true."

We use the notation $|S|$ to denote the number of bits needed to represent S using a "straightforward" encoding scheme. By definition of $M^r_{k\ell}$, we have that $|M^r_{k\ell}|$ is $O(|S^0_{k\ell}| + |S^r_{k\ell}| + |R^r_{k\ell}|)$. Since the pair-system is assumed to be nonterminating (Section 6.5), $R^r_{k\ell}$ is total, so we have $|S^0_{k\ell}| \le |R^r_{k\ell}|$ and $|S^r_{k\ell}| \le |R^r_{k\ell}|$. Hence $|M^r_{k\ell}|$ is $O|R^r_{k\ell}|$. We consider each step of the procedure and compute its worst-case time complexity. Step 0 can be performed in constant time. Step 1 requires the construction of $R^r_{k\ell}$. We generate $R^r_{k\ell}$ incrementally by starting with $S^0_{k\ell}$, selecting some state $s_{k\ell} \in S^0_{k\ell}$, and "expanding" $s_{k\ell}$ by computing all the transitions with source state $s_{k\ell}$ using the compact pair structure definition (5.4.2). We then mark $s_{k\ell}$ as "expanded" and repeat the process until all reachable states have been marked. Each reachable state (and each reachable transition) need be inspected only once. The "expansion" of each state using Definition 5.2.1 requires access to P^ℓ_k and P^k_ℓ. Hence the time complexity of step 1 is $O(|P^\ell_k| \cdot |R^r_{k\ell}|)$.

From the definition of the compact notation given in Section 5.4, we see that the complexity of step 2 (converting P^ℓ_k, P^k_ℓ to compact notation) is $O(|P^\ell_k|)$.

Step 3 can be performed by a single traversal of $R^r_{k\ell}$, and so its complexity is $O(|R^r_{k\ell}|)$.

The complexity of step 4.1 is $O(n)$, as we must count all the moves in $t_k.moves$. The complexity of step 4.2 is simply $O(\sum_{J \in \mathcal{J}(t_k)} |J|)$. Since $|J| = m + 1$, this can be rewritten as $O(\sum_{1 \le m \le n} m)$, i.e., $O(n^2)$.

We evaluate the complexity of step 4.3 starting with the most deeply nested iterations and proceeding outward. Now (a) can be rewritten as

$$(t_J{\uparrow}jk \models \bigwedge a^J_j \,.\, (\{\!\!\{a^J_j.start\}\!\!\} \Rightarrow a^J_j.guard_k))$$

or

$$\bigvee a^J_k \,.\, (\,(t_J{\uparrow}j \models \{\!\!\{a^J_k.start\}\!\!\}) \text{ and } (\bigwedge \ell \in \{\ell_1, \ldots, \ell_m\} \,.\, (t_J{\uparrow}j\ell \models a^J_k.guard_\ell))\,).$$

Hence the complexity of evaluating (a) is $O(sz2 \cdot |P^\ell_k| + m \cdot sz2 \cdot |P^\ell_k|)$, where $sz2$ is the number of bits needed to represent a single $k\ell$-state. This is just $O(m \cdot sz2 \cdot |P^\ell_k|)$. Since step 4.3.2 requires the inspection of all states in S^r_J (but *not* all transitions in R^r_J), the complexity of step 4.3.2 is $O(m \cdot |S^r_J| \cdot |P^\ell_k|)$. Step 4.3.1 is carried out in an analogous manner to step 1. Each reachable state (and each reachable transition) in R^r_J is inspected only once. The "expansion" of each state (using the compact

0. $WGflag := true$;
1. generate $M^r_{k\ell}$ from $(S^0_{k\ell}, P^\ell_k \| P^k_\ell)$;
2. translate P^ℓ_k, P^k_ℓ into compact notation;
3. $S^r_k := \{t_k \mid \bigvee s_{k\ell} \in S^r_{k\ell} . (s_{k\ell}{\upharpoonright}k = t_k)\}$;
4. for all t_k in S^r_k
   - 4.1 $n := |t_k.moves|$;
   - 4.2 $\mathcal{J}(t_k) := \{J \mid J = \{\{j,k\},\{k,\ell_1\},\ldots,\{k,\ell_m\}\}$ and $m \in [1:n]$ and $j,k,\ell_1,\ldots,\ell_m$ are pairwise distinct $\}$;
   - 4.3 for all J in $\mathcal{J}(t_k)$
     - 4.3.1 generate M^r_J;
     - 4.3.2 for all t_J such that $t_J{\upharpoonright}k = t_k$ and $s_J \xrightarrow{k} t_J \in R^r_J$ for some s_J
       if (a) evaluates to false, then $WGflag := false$;
   - 4.4 $\mathcal{J}'(t_k) := \{J \mid J = \{\{k,\ell_1\},\ldots,\{k,\ell_m\}\}$ and $m \in [1:n]$ and $k,\ell_1,\ldots,\ell_m$ are pairwise distinct $\}$;
   - 4.5 for all J in $\mathcal{J}'(t_k)$
     - 4.5.1 generate M^r_J;
     - 4.5.2 set j equal to an arbitrarily selected member of $\{\ell_1,\ldots,\ell_m\}$;
     - 4.5.3 for all t_J such that $t_J{\upharpoonright}k = t_k$ and $s_J \xrightarrow{k} t_J \in R^r_J$ for some s_J
       if (a) evaluates to false, then $WGflag := false$

Fig. 16. Procedure to check the wait-for-graph assumption.

I-structure definition (5.4.3) with $I := J$ now) requires access to P^J_k and P^J_ℓ (for $\ell \in \{j,\ell_1,\ldots,\ell_m\}$). Thus the complexity of step 4.3.1 is $O((|P^J_k| + m \cdot |P^J_\ell|) \cdot |R^r_J|)$. In compact notation, P^J_k is derived by applying the operator "$\otimes_{\ell \in J(k)}$" to the label of every move in P^ℓ_k. Hence $|P^J_k|$ is $O(|J| \cdot |P^\ell_k|)$, which is $O(m \cdot |P^\ell_k|)$ since $|J|$ is $O(m)$. $|P^J_\ell|$ (for $\ell \in \{j,\ell_1,\ldots,\ell_m\}$) is $O(|P^k_\ell|)$, since P^J_ℓ has only one J-neighbor, namely P^J_k. Hence the complexity of step 4.3.1 can be rewritten as $O(m \cdot |P^\ell_k| \cdot |R^r_J|)$. So the complexity of steps 4.3.1 and 4.3.2 combined is $O(m \cdot |P^\ell_k| \cdot |R^r_J| + m \cdot |P^\ell_k| \cdot |S^r_J|)$. Since $|S^r_J| \le |R^r_J|$, the first summand is not less than the second, so this is $O(m \cdot |P^\ell_k| \cdot |R^r_J|)$.

Now R^r_J can be regarded as a "product" of the $m+1$ transition sets $R_{k\ell}$ for $\ell \in \{j,\ell_1,\ldots,\ell_n\}$, as given by the transition-mapping lemma (6.4.1). Thus, $|R^r_J|$ is $O(|R^r_{k\ell}|^{m+1}$. However, this counts the process P^J_k $m+1$ times instead of once, so we can improve this upper bound to $O(|R^r_{k\ell}|^{m+1}/|P^\ell_k|^m)$. We rewrite this as $O(|R^r_{k\ell}| \cdot \alpha^m)$, where $\alpha = |R^r_{k\ell}|/|P^\ell_k|$. So the complexity of steps 4.3.1 and 4.3.2 combined is rewritten as $O(m \cdot |P^\ell_k| \cdot |R^r_{k\ell}| \cdot \alpha^m)$.

Step 4.3 iterates steps 4.3.1 and 4.3.2 over all J in $\mathcal{J}(t_k)$, so its complexity is $O(\sum_{J \in \mathcal{J}(t_k)} m \cdot |P^\ell_k| \cdot |R^r_{k\ell}| \cdot \alpha^m)$. Now J determines m, since $m =$ (the number of pairs in J) -1. So as J varies from $\{\{j,k\},\{k,\ell_1\}\}$ to $\{\{j,k\},\{k,\ell_1\},\ldots,\{k,\ell_n\}\}$, m will vary from 1 to n. Thus the complexity of step 4.3 is $O(|P^\ell_k| \cdot |R^r_{k\ell}| \cdot \sum_{1 \le m \le n} m \cdot \alpha^m)$, since $|P^\ell_k|$ and $|R^r_{k\ell}|$ are independent of J. $\sum_{1 \le m \le n} m \cdot \alpha^m$ is bounded from above by $n \cdot \sum_{1 \le m \le n} \alpha^m$, and $\sum_{1 \le m \le n} \alpha^m = (\alpha^{n+1} - 1)/(\alpha - 1)$, which is approximately $O(\alpha^n)$ when $\alpha \gg 1$, as is usually the case. Hence the complexity of step 4.3 is $O(n \cdot |P^\ell_k| \cdot |R^r_{k\ell}| \cdot \alpha^n)$. Finally, the complexity of steps 4.4 and 4.5 is easily seen to be at most equal to the complexity of steps 4.2, 4.3 respectively, since the only difference is that J is smaller by one pair. Thus, the overall complexity of the body of step 4 (i.e., steps 4.1–4.5) is $O(n + n^2 + (n \cdot |P^\ell_k| \cdot |R^r_{k\ell}| \cdot \alpha^n))$. Now n is bounded by the maximum branching within a single pair-process, so $|P^\ell_k| \ge n$. Hence the above is $O(n \cdot |P^\ell_k| \cdot |R^r_{k\ell}| \cdot \alpha^n)$.

The body of step 4 is repeated for every k-state t_k in S_k^r. Let b denote the maximum value of n $(= |t_k.moves|)$ as t_k ranges over S_k^r. Thus the complexity of step 4 is $O(b \cdot |P_k^\ell| \cdot |R_{k\ell}^r| \cdot \alpha^b \cdot |S_k^r|)$. The complexities of steps 1, 2, and 3 computed above are $O(|P_k^\ell| \cdot |R_{k\ell}^r|)$, $O(|P_k^\ell|)$, and $O(|R_{k\ell}^r|)$, respectively. These are all subsumed by the complexity of step 4, which therefore gives the overall complexity of the procedure for checking WG. Now $|S_k^r|$ is $O(|P_k^\ell|)$, since S_k^r is a subset of the k-states of P_k^ℓ. Thus, the overall time complexity can be rewritten as $O(b \cdot |P_k^\ell|^2 \cdot |R_{k\ell}^r| \cdot \alpha^b)$. Replacing α by $|R_{k\ell}^r|/|P_k^\ell|$, we obtain $O(b \cdot |R_{k\ell}^r|^{b+1}/|P_k^\ell|^{b-2})$ for the worst-case time complexity of the procedure for checking the wait-for-graph assumption.

Now each $|R_{k\ell}^r|$ is $O(|P_k^\ell|^2 \cdot 2^{|\mathcal{SH}_{k\ell}|})$, since a pair-system contains two pair-processes (which contribute $|P_k^\ell|^2$ to the size of $R_{k\ell}^r$), and a set of pairwise shared variables (which contributes $2^{|\mathcal{SH}_{k\ell}|}$ to the size of $R_{k\ell}^r$). Hence, the overall time complexity can also be written as $O(b \cdot (|P_k^\ell|^{2b+2} \cdot 2^{(b+1)|\mathcal{SH}_{k\ell}|})/|P_k^\ell|^{b-2})$, that is, $O(b \cdot |P_k^\ell|^{b+4} \cdot 2^{(b+1)|\mathcal{SH}_{k\ell}|})$.

Also, the space complexity of the procedure is $O(|R_J^r|)$, since R_J^r is the largest of a fixed set of data structures that are used. This is $O(|R_{k\ell}^r|^{b+1}/|P_k^\ell|^b)$, or, alternatively, $O(|P_k^\ell|^{b+2} \cdot 2^{(b+1)|\mathcal{SH}_{k\ell}|})$.

E.2 Checking the Liveness Assumption

The liveness assumption (6.7.2.1) is mechanically checked as follows. We introduce a "new"[15] atomic proposition Q to M_J. We set Q to *true* in every state s_J of M_J such that

$$M_{jk}, s_J{\uparrow}jk \models EGex_k$$

and to *false* in all other states of M_J.

Now $\bigwedge a_i^J . (a_i^J \longrightarrow P_j^J \not\in W_J(s_J))$ is equivalent to $s_J \models noblock(i,j)$, where $noblock(i,j) \stackrel{\text{df}}{=} \bigwedge a_i^J . (\{\!\!\{a_i^J.start\}\!\!\} \Rightarrow a_i^J.guard_j)$. Thus, the liveness assumption can be rewritten as

$$\begin{array}{l} \text{for every reachable state } s_J \text{ in } M_J, \\ \qquad M_{jk}, s_J{\uparrow}jk \models EGex_k \;\; \text{implies} \;\; M_J, s_J \models noblock(i,j). \end{array}$$

By definition of Q, we have $M_{jk}, s_J{\uparrow}jk \models EGex_k$ iff $M_J, s_J \models Q$. Hence the above is equivalent to

$$\begin{array}{l} \text{for every reachable state } s_J \text{ in } M_J, \\ \qquad M_J, s_J \models Q \;\; \text{implies} \;\; M_J, s_J \models noblock(i,j). \end{array}$$

By CTL semantics, this is equivalent to

$$\begin{array}{l} \text{for every reachable state } s_J \text{ in } M_J, \\ \qquad M_J, s_J \models (Q \Rightarrow noblock(i,j)). \end{array}$$

Finally, we translate the quantification over all reachable states into CTL by prefacing the formula with the AG modality, and evaluating it in all initial states:

[15] i.e., $Q \not\in \mathcal{AP}_i \cup \mathcal{AP}_j \cup \mathcal{AP}_k$.

$$M_J, S_J^0 \models AG(Q \Rightarrow noblock(i,j)).$$

To determine the truth assignment to Q, we model-check M_{jk}^r (the reachable part of M_{jk}) for the formula $AGEGex_k$. We employ the CTL model-checking algorithm of Clarke et al. [1986], which marks all states in M_{jk} with all subformulae of $AGEGex_k$ that are true in the state. In particular, all states of M_{jk} that satisfy $EGex_k$ will be so marked. We can then use this marking to assign the appropriate truth value to Q in each reachable state s_J of M_J. Finally, we model-check M_J with respect to $AG(Q \Rightarrow noblock(i,j))$ to determine whether the liveness assumption holds or not, again using the model-checking algorithm of Clarke et al. [1986].

The algorithm of Clarke et al. [1986] has time complexity linear in both the structure and the formula being checked. Here we invoke it twice, once for M_{jk}^r with respect to $AGEGex_k$ and once for M_J^r with respect to $AG(Q \Rightarrow noblock(i,j))$. Since $|AGEGex_k|$ is constant, and $|M_{jk}^r| < |M_J^r|$ is easily seen to hold, the time complexity of model-checking M_J^r dominates. Since $noblock(i,j)$ is quantified over all the moves of P_i^J, its size is $O(|P_i^J|)$. Hence, the liveness assumption can be model-checked in time $O(|M_J^r| \cdot |P_i^J|)$. Now $|M_J^r|$ is $O(|P_k^\ell|^3 \cdot 2^{2|\mathcal{SH}_{k\ell}|})$, since the J-system contains three processes and two sets of pairwise shared variables. Also $|P_i^J|$ is $O(|P_\ell^k|)$, since $|J|$ is fixed. Hence the overall complexity can be rewritten as $O(|P_k^\ell|^4 \cdot 2^{2|\mathcal{SH}_{k\ell}|})$. Note that the check must be made for the case $i \neq k$ and also for the case $i = k$.