A EQUIVALENCE OF THE PTSO OPERATIONAL AND DECLARATIVE SEMANTICS

A.1 Intermediate Operational Semantics

Types.

Annotated persistent memory

\[ M \in \text{AMEM} \triangleq \left\{ f \in \text{LOC} \xrightarrow{\text{fin}} W \cup U \mid \forall x \in \text{dom}(f). \ \text{loc}(f(x)) = x \right\} \]

Annotated persistent sub-buffers

\[ (o, pb) \in \text{APBUFF} \triangleq \left\{ (o, pb) \in \text{Opt} \langle PF \rangle \times \text{LOC} \xrightarrow{\text{fin}} \text{SEQ} \langle W \cup U \rangle \mid \forall x, e. \ e \in pb(x) \Rightarrow \text{loc}(e) = x \right\} \]

Annotated persistent buffers

\[ PB \in \text{APBUFF} \triangleq \text{SEQ} \langle \text{APBUFF} \rangle \setminus \epsilon \]

Annotated volatile buffers

\[ b \in \text{ABUFF} \triangleq \text{SEQ} \langle W \rangle \]

Annotated volatile buffer maps

\[ B \in \text{ABMAP} \triangleq \left\{ B \in \text{TID} \xrightarrow{\text{fin}} \text{ABUFF} \mid \forall w. \ \forall \tau \in \text{dom}(B). \ w \in B(\tau) \Rightarrow \text{tid}(w) = \tau \right\} \]

Annotated labels

\[ \text{ALABELS} \ni \lambda ::= \text{R}(r, w) \quad \text{where } r \in R, w \in W \cup U, \text{loc}(r) = \text{loc}(w), \text{val}_r(r) = \text{val}_w(w) \]

\[ \mid \text{U}(u, w) \quad \text{where } u \in U, w \in W \cup U, \text{loc}(u) = \text{loc}(w), \text{val}_r(u) = \text{val}_w(w) \]

\[ \mid \text{W}(w) \quad \text{where } w \in W \]

\[ \mid \text{F}(f) \quad \text{where } f \in F \]

\[ \mid \text{PF}(pf) \quad \text{where } pf \in PF \]

\[ \mid \text{PS}(ps) \quad \text{where } ps \in PS \]

\[ \mid \text{B}(w) \quad \text{where } w \in W \]

\[ \mid \text{PB}(e) \quad \text{where } e \in W \cup U \cup PF \]

\[ \mid \text{E}(\tau) \quad \text{where } \tau \in \text{TID} \]

\[ \pi \in \text{PATH} \triangleq \text{SEQ} \langle \text{ALABELS} \setminus \{E(\tau) \mid \tau \in \text{TID}\} \rangle \] \quad \text{Event paths}

\[ \pi \in \text{PPath} \triangleq \text{SEQ} \langle \text{ALABELS} \cap \{B(e), PB(e) \mid e \in E\} \rangle \] \quad \text{Propagation paths}

\[ H \in \text{TRACE} \triangleq \text{PPath} \times \text{PATH} \] \quad \text{Traces}

\[ \mathcal{H} \in \text{HIST} \triangleq \text{SEQ} \langle \text{TRACE} \rangle \] \quad \text{Histories}

Let

\[ \text{AMEM} \ni M_0 \quad \text{s.t. } \forall x. \ M_0(x) = \text{init}_x \text{ with } \text{lab}(\text{init}_x) \triangleq W(x, 0) \]

\[ \text{APBUFF} \ni pb_0 \quad \text{s.t. } \forall x. \ pb_0(x) = \epsilon \]

\[ \text{APBUFF} \ni PB_0 \triangleq (\text{NONE}, pb_0) \]

\[ \text{ABUFF} \ni b_0 \triangleq \epsilon \]

\[ \text{ABMAP} \ni B_0 \quad \text{s.t. } \forall \tau. \ B_0(\tau) = b_0 \]

Storage Subsystem

\[ \text{tid}(w) = \tau \]

\[ M, PB, B \xrightarrow{W(w)} M, PB, B[\tau \mapsto w.B(\tau)] \quad \text{(AM-WRITE)} \]
B(τ) = b, w \quad \text{loc}(w) = x \quad PB = (\text{NONE}, pb).PB'' \quad PB' = (\text{NONE}, pb[x \mapsto w.\text{pb}(x)]).PB'' \quad (\text{AM-BPROP})

\frac{M, PB, B \xrightarrow{B(w)} M, PB', B[τ \mapsto b]}{PB = PB''.(a, pb) \quad \text{pb}(x) = \text{S.e} \quad PB' = PB''.(a, pb[x \mapsto S])] \quad (\text{AM-BPROP})

\frac{M, PB, B \xrightarrow{PB(e)} M[x \mapsto e], PB', B}{PB \neq e} \quad (\text{AM-PBPROP})

\frac{M, PB, (\text{SOME}(pf), pb_0), B \xrightarrow{PB(pf)} M, PB, (\text{NONE}, pb_0), B}{PB \neq e} \quad (\text{AM-PBPROP})

\frac{\text{tid}(r) = τ \quad \text{loc}(r) = x \quad B(τ) = b \quad \text{read}(M, PB, b, x) = e}{(\text{AM-READ})}

\frac{M, PB, B \xrightarrow{R(τ, e)} M, PB, B}{\text{tid}(w) = τ \quad \text{loc}(w) = x \quad B(τ) = e \quad \text{PB} = (\text{NONE}, pb).PB' \quad \text{read}(M, PB, e, x) = e} \quad (\text{AM-RMW})

\frac{M, PB, B \xrightarrow{U(u, e)} M, (\text{NONE}, pb[x \mapsto u.\text{pb}(x)]).PB', B}{\text{tid}(f) = τ \quad B(τ) = e} \quad (\text{AM-FENCE})

\frac{M, PB, B \xrightarrow{F(f)} M, PB, B}{\text{tid}(pf)} = τ \quad B(τ) = e \quad PB = (\text{NONE}, pb).PB' \quad (\text{AM-PFENCE})

\frac{M, PB, B \xrightarrow{PS(ps)} M, PB_0, B}{\text{tid}(pf) = τ \quad B(τ) = e} \quad (\text{AM-PSYNC})

\text{where}

\text{read}(\ldots, \ldots, \ldots) : \text{AMEM} \times \text{APBUFF} \times \text{ABUFF} \times \text{LOC} \rightarrow W \cup U

\text{read}(M, PB, b, x) \triangleq \begin{cases} e & \text{if } \text{rd}_b(b, x) = e \\ M(x) & \text{otherwise} \end{cases}

\text{with}

\text{rd}_b(\ldots) : \text{SEQ}(W \cup U) \times \text{LOC} \rightarrow E

\text{rd}_b(e, x) \text{ undefined} \quad \text{rd}_b(e, s, x) \triangleq \begin{cases} e & \text{loc}(e) = x \\ \text{rd}_b(s, x) & \text{otherwise} \end{cases}

\text{rd}_b(\ldots) : \text{APBUFF} \times \text{LOC} \rightarrow W \cup U

\text{rd}_b(\ldots) \text{ undefined}

\text{rd}_b((a, pb).PB, x) \triangleq \begin{cases} e & \text{if } \text{rd}_b(pb(x), x) = e \\ \text{rd}_b(PB, x) & \text{otherwise} \end{cases}

\textbf{Thread Subsystem}

Thread-local steps.

\begin{align*}
& c_1, s \xrightarrow{λ} c_1', s' \quad \text{(AT-SEQ1)} \quad (\text{AT-SEQ2}) \quad \text{skip}; c, s \xrightarrow{E(τ)} c, s \\
& \text{if } e \text{ then } c_1 \text{ else } c_2, s \xrightarrow{E(τ)} c_1, s
\end{align*}
\[
\begin{align*}
\text{if } e \text{ then } c_1 \text{ else } c_2, s & \xrightarrow{\mathcal{E}(\tau)} c_2, s \\
\text{while } e \text{ do } c, s & \xrightarrow{\mathcal{E}(\tau)} \text{if } e \text{ then } (c; \text{while } e \text{ do } c) \text{ else skip}, s \\
\end{align*}
\]

\[
\begin{align*}
s'(e) &= 0 \\
\text{if } e \text{ then } c_1 \text{ else } c_2, s & \xrightarrow{\mathcal{E}(\tau)} c_2, s \\
\end{align*}
\]

\[
\begin{align*}
s' &= s[a \mapsto s(e)] \\
a & \xrightarrow{e} s, \text{skip}, s' \\
r &= (n, \tau, R(x, v)) \\
a & \xrightarrow{R(r, w)} x, s, \text{skip}, s' \\
\end{align*}
\]

\[
\begin{align*}
u & \neq s(e) \\
a & \xrightarrow{\mathcal{CAS}(x, e, e'), s} R(r, w), \text{skip}, s' \\
\end{align*}
\]

\[
\begin{align*}
u & = (n, \tau, U(x, s(e), s(e'))) \\
a & \xrightarrow{U(u, w)} x, s, \text{skip}, s' \\
\end{align*}
\]

\[
\begin{align*}
u & = (n, \tau, U(x, v+s(e))) \\
a & \xrightarrow{U(u, w)} x, s, \text{skip}, s' \\
\end{align*}
\]

\[
\begin{align*}
fence, s & \xrightarrow{\mathcal{F}(f)} \text{skip}, s \\
\end{align*}
\]

\[
\begin{align*}
\text{pfence, s} & \xrightarrow{\mathcal{PF}(pf)} \text{skip}, s \\
\end{align*}
\]

\[
\begin{align*}
\text{psync, s} & \xrightarrow{\mathcal{PS}(ps)} \text{skip}, s \\
\end{align*}
\]

\[
\begin{align*}
P(\tau), S(\tau) & \xrightarrow{\lambda} c, s \\
\end{align*}
\]

\[
\begin{align*}
P, S & \xrightarrow{\lambda} P[\tau \mapsto c], S[\tau \mapsto s] \\
\end{align*}
\]

\[
\begin{align*}
\text{where}
\end{align*}
\]

\[
\begin{align*}
tid(R(r, w)) & \triangleq tid(r) \\
tid(U(u, w)) & \triangleq tid(u) \\
tid(W(w)) & \triangleq tid(w) \\
tid(F(f)) & \triangleq tid(f) \\
tid(PF(pf)) & \triangleq tid(pf) \\
tid(PS(ps)) & \triangleq tid(ps) \\
tid(B(w)) & \triangleq tid(w) \\
tid(PB(e)) & \triangleq tid(e) \\
tid(\mathcal{E}(\tau)) & \triangleq \tau \\
\end{align*}
\]
**Event-Annotted Operational Semantics**

\[ P, S \xrightarrow{\varepsilon} P', S' \quad (\text{A-SilentP}) \]

\[ M, PB, B \xrightarrow{\varepsilon} M', PB', B' \quad (\text{A-SilentM}) \]

\[ M, PB, B \xrightarrow{\lambda} M', PB', B' \quad \lambda \in \{ B(e), PB(e) \} \quad \text{fresh}(\lambda, \pi) \quad \text{fresh}(\lambda, \mathcal{H}) \quad (\text{A-PropM}) \]

\[ P, S \xrightarrow{\lambda} P', S' \quad M, PB, B \xrightarrow{\lambda} M', PB', B' \quad \lambda \neq \varepsilon \quad \text{fresh}(\lambda, \pi) \quad \text{fresh}(\lambda, \mathcal{H}) \quad (\text{A-Step}) \]

\[ M, PB, B \xrightarrow{\pi'} M', PB_0, B_0 \quad (\text{A-Crash}) \]

with

\[ (M, PB, B) \xrightarrow{\text{PB}(e)} (M'', PB'', B'') \quad (M'', PB'', B'') \xrightarrow{\pi} (M', PB', B') \]

\[ (M, PB, B) \xrightarrow{\text{PB}(e), \pi} (M', PB', B') \]

\[ (M, PB, B) \xrightarrow{\text{B}(e)} (M'', PB'', B'') \quad (M'', PB'', B'') \xrightarrow{\pi} (M', PB', B') \]

\[ (M, PB, B) \xrightarrow{\text{B}(e), \pi} (M', PB', B') \]

and

\[ \text{fresh}(\lambda, \pi) \triangleq \lambda \not\in \pi \land \forall e, w, w'. \]

\[ (\lambda = \text{R}(e, w) \Rightarrow \text{R}(e, w') \not\in \pi) \land (\lambda = \text{U}(e, w) \Rightarrow \text{U}(e, w') \not\in \pi) \]

\[ \text{fresh}(\lambda, \mathcal{H}) \triangleq \forall (\pi', \pi) \in \mathcal{H}. \text{fresh}(\lambda, \pi', \pi) \]

**Definition A.1.**

\[ \text{complete}(\pi) \triangleq \forall e. W(e) \in \pi \Rightarrow B(e) \in \pi \]

\[ B(e) \in \pi \Rightarrow \text{PB}(e) \in \pi \]

\[ U(e, -) \in \pi \Rightarrow \text{PB}(e) \in \pi \]

\[ \text{PF}(e) \in \pi \Rightarrow \text{PB}(e) \in \pi \]
\[\text{wfrd}(r, e, \pi, \pi') \triangleq \left\{ \begin{aligned} &\exists \pi_1, \pi_2, \lambda. \quad \pi = \pi_1.\lambda.\pi_2 \\
&\quad \land (\lambda = B(e) \lor \lambda = U(e, -) \lor (\lambda = W(e) \land \text{tid}(e) = \text{tid}(r))) \\
&\quad \land (\lambda = B(e) \lor \lambda = U(e, -)) \Rightarrow \\
&\quad \{ B(e'), U(e', -) \in \pi \mid \text{loc}(e') = \text{loc}(r) \} = \emptyset \\
&\quad \land \{ e'. W(e') \in \pi \land B(e') \notin \pi \\
&\quad \land \text{loc}(e') = \text{loc}(r) \land \text{tid}(e') = \text{tid}(r) \} = \emptyset \\
&\quad \land (\lambda = W(e)) \Rightarrow \\
&\quad B(e) \notin \pi_1 \land \{ W(e') \in \pi_1 \mid \text{loc}(e') = \text{loc}(r) \land \text{tid}(e') = \text{tid}(r) \} = \emptyset \\
&\quad \lor \exists \pi_1, \pi_2, \pi'. \quad \pi = \pi_1.\pi_2.\pi' \\
&\quad \land B(e'), U(e', -) \in \pi, \quad \text{loc}(e') = \text{loc}(r) \land \text{tid}(e') = \text{tid}(r) \\
&\quad \lor \land W(e'') \in \pi, \quad \text{loc}(e'') = \text{loc}(r) \\
&\quad \lor \land \text{tid}(e'') = \text{tid}(r) \\
&\quad \lor \land e = \text{init}_{\text{loc}(e)} \land \\
&\quad \land B(e'), U(e', -) \in \pi, \quad \text{loc}(e') = \text{loc}(r) \land \text{tid}(e') = \text{tid}(r) \\
&\quad \lor \land W(e'') \in \pi, \quad \text{loc}(e'') = \text{loc}(r) \\
&\quad \lor \land \text{tid}(e'') = \text{tid}(r) \\&\end{aligned} \right\} \]
Definition A.2.

\[ \text{wf}(M, PB, B, \mathcal{H}, \pi) \defeq \text{mem}(\mathcal{H}, \pi) = M \land \text{pbuff}(PB_0, \pi) \land \text{bmap}(B, \pi) \land \text{wfp}(\pi, \mathcal{H}) \land \text{wfh}(\mathcal{H}) \]

where

\[ \text{mem}(\mathcal{H}, \pi) = M \defeq \forall x \in \text{Loc}. M(x) = \text{read}(\mathcal{H}, \pi, x) \]

\[ \text{read}(\mathcal{H}, \lambda, \pi, x) \defeq \begin{cases} e & \text{if } \exists \lambda. \lambda = PB(\epsilon) \land \text{loc}(\epsilon) = x \\ \text{read}(\mathcal{H}, \pi, x) & \text{otherwise} \end{cases} \]

\[ \text{read}((-\pi), \mathcal{H}, \epsilon, x) \defeq \text{read}(\mathcal{H}, \pi, x) \]

\[ \text{read}(\epsilon, \epsilon, x) \defeq \text{init}_x \]

\[ \text{pbuff}(PB, \epsilon) \defeq P \]

\[ \begin{cases} \text{pbuff}((\text{None}, pb[x \mapsto e.p}(x))).PB, \pi) & \text{if } \exists \epsilon, x. \\ \lambda \in \{ U(\epsilon, -), B(\epsilon) \} \\ \land \text{loc}(\epsilon) = x \\ \land PB(\epsilon) \not\in \pi \end{cases} \]

\[ \text{pbuff}((\text{None}, pb).PB, \pi, \lambda) \defeq \begin{cases} \text{pbuff}((\text{None}, pb_0).\text{Some}(e), pb).PB, \pi) & \text{if } \exists \epsilon. \lambda = \text{PF}(\epsilon) \\ \land PB(\epsilon) \not\in \pi \\ \text{pbuff}((\text{None}, pb).PB, \pi) & \text{otherwise} \end{cases} \]

\[ \text{bmap}(B, \epsilon) \defeq B \]

\[ \text{bmap}(B, \pi, \lambda) \defeq \begin{cases} \text{bmap}(B[\tau \mapsto e.B(\tau)], \pi) & \text{if } \exists \epsilon, x. \lambda = \text{W}(\epsilon) \land \text{tid}(\epsilon) = \tau \\ \land B(\epsilon) \not\in \pi \\ \text{bmap}(B, \pi) & \text{otherwise} \end{cases} \]

\[ \text{wfh}(\epsilon) \defeq \text{true} \]

\[ \text{wfh}((\pi', \pi).\mathcal{H}) \defeq \text{wfp}(\pi', \pi, \mathcal{H}) \land \text{complete}(\pi'.\pi) \land \text{wfh}(\mathcal{H}) \]

Lemma A.1. For all P, P', S, S', PB, PB', B, B', H, H', \pi, \pi':

- \[ \text{wf}(M_0, PB_0, B_0, \epsilon, \epsilon) \]
- \[ \text{if } P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P', S', M', PB', B', \mathcal{H}', \pi' \text{ and } \text{wf}(M, PB, B, \mathcal{H}, \pi), \text{ then } \text{wf}(M', PB', B', \mathcal{H}', \pi') \]
- \[ \text{if } P, S_0, M_0, PB_0, B_0, \epsilon, \epsilon \Rightarrow^* \text{skip}, S, M, PB, B, \mathcal{H}, \pi, \text{ then } \text{wf}(M, PB, B, \mathcal{H}, \pi) \]

Proof. The proof of the first part follows trivially from the definitions of M_0, PB_0, and B_0. The second part follows straightforwardly by induction on the structure of \Rightarrow. The last part follows from the previous two parts and induction on the length of \Rightarrow^*. \( \square \)

Graph Operational Semantics

Let

\[ \Gamma \in \text{Ghist} \defeq \text{Seq}(\text{Graph} \times \text{Trace}) \text{ Graph histories} \]

\[ P, S \xrightarrow{E(\epsilon)} P', S' \quad (\text{G-SILENTP}) \]

\[ P, S, \Gamma, \pi \Rightarrow P', S', \Gamma, \pi \]

\[ \lambda \in \{ B(\epsilon), PB(\epsilon) \} \quad \text{fresh}(\lambda, \pi) \quad \text{fresh}(\lambda, \Gamma) \quad (\text{G-PROP}) \]

\[
\begin{align*}
P, S \to \lambda \quad &\to P', S' \\
\text{fresh}(\lambda, \pi) \quad &\text{fresh}(\lambda, \Gamma) \\
\frac{P, S, \Gamma, \pi \Rightarrow P', S', \Gamma, \lambda, \pi}{(G\text{-STEP})}
\end{align*}
\]

\[
\begin{align*}
\text{comp}(\pi, \pi') \quad &\quad \text{getG}(\Gamma, \pi, \pi') = G \\
\frac{P, S, \Gamma, \pi \Rightarrow \text{recover}, S_0, (G, (\pi', \pi)) \Gamma, \epsilon}{(G\text{-CRASH})}
\end{align*}
\]

where

\[
\begin{align*}
\text{fresh}(\lambda, \Gamma) \iff \forall (\pi', \pi) \in \Gamma. \text{fresh}(\lambda, \pi', \pi) \\
\text{comp}(\_ , \_ ) \quad &\quad \text{def} \quad \text{PATH} \times \text{PPATH} \to \{\text{true, false}\}
\end{align*}
\]

\[
\begin{align*}
\text{comp}(\pi, \pi') \iff \forall e. W(e) \in \pi \land B(e) \notin \pi \iff B(e) \in \pi'
\end{align*}
\]

\[
\begin{align*}
\text{getG}(\Gamma, \pi, \pi') \triangleq
\begin{cases}
(E^0, E^p, E, \text{po, rf, tso, nvo}) & \text{if wfp}(\pi', \pi, \text{hist}(\Gamma)) \land \text{complete}(\pi', \pi) \\
\text{undefined} & \text{otherwise}
\end{cases}
\end{align*}
\]

with

\[
\begin{align*}
\text{hist}(\epsilon) = \epsilon \\
\text{hist}((G, H), \Gamma) = H.\text{hist}(\Gamma)
\end{align*}
\]

\[
\begin{align*}
E^0 = \begin{cases}
\text{init}_x \mid x \in \text{Loc} & \text{if } \Gamma = \epsilon \\
\max \{G, \text{nvo}\}_{G, E^p \cap (U_x \cup W_x)} \mid x \in \text{Loc} & \text{if } \Gamma = (G, -), \Gamma'
\end{cases}
\end{align*}
\]

\[
\begin{align*}
E^p = E^0 \cup \{e \mid \exists \lambda \in \pi. \text{getPE}(\lambda) = e\} \\
E = E^0 \cup \{e \mid \exists \lambda \in \pi. \text{getE}(\lambda) = e\} \\
\text{rf} = \{(w, e) \mid R(e, w) \in \pi \lor U(e, w) \in \pi\}
\end{align*}
\]

\[
\begin{align*}
\text{po} = E^0 \times (E \setminus E^0) \cup \bigcup_{r \in \Pi_B} \begin{cases}
(e_1, e_2) & e_1 = \text{getE}(\lambda_1) \land e_2 = \text{getE}(\lambda_2) \\
& \land \text{tid}(e_1) = \text{tid}(e_2) = \tau \\
& \land \lambda_1 <_\pi \lambda_2
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\text{tso} \triangleq E^0 \times (E \setminus E^0) \\
\cup \begin{cases}
(e_1, e_2) & \exists \lambda_1, \lambda_2 \in \pi'. \pi. \\
& e_1 = \text{getBE}(\lambda_1) \land e_2 = \text{getBE}(\lambda_2) \land \lambda_1 <_\pi \pi \lambda_2
\end{cases}
\end{align*}
\]

\[
\begin{align*}
\text{nvo} \triangleq E^0 \times (E \setminus E^0) \\
\cup \begin{cases}
(e_1, e_2) & \exists \lambda_1, \lambda_2 \in \pi'. \pi. \\
& e_1 = \text{getPE}(\lambda_1) \land e_2 = \text{getPE}(\lambda_2) \land \lambda_1 <_\pi \pi \lambda_2
\end{cases}
\end{align*}
\]
and
\[
\text{getE}(\cdot) : \text{ALABELS} \rightarrow E
\]
\[
\text{getE}(\lambda) = \begin{cases} 
    e & \text{if } \exists e, w, \lambda \in \{R(e, w), U(e, w), W(e), F(e), PF(e), PS(e)\} \\
    \text{undefined} & \text{otherwise}
\end{cases}
\]

\[
\text{getPE}(\cdot) : \text{ALABELS} \rightarrow E
\]
\[
\text{getPE}(\lambda) = \begin{cases} 
    e & \text{if } \exists e, \lambda \in \{R(e, ,)F(e), PS(e), PB(e)\} \\
    \text{undefined} & \text{otherwise}
\end{cases}
\]

\[
\text{getBE}(\cdot) : \text{ALABELS} \rightarrow E
\]
\[
\text{getBE}(\lambda) = \begin{cases} 
    e & \text{if } \exists e, w, \lambda \in \{R(e, w), U(e, w), F(e), PF(e), PS(e), B(e)\} \\
    \text{undefined} & \text{otherwise}
\end{cases}
\]

### A.2 Soundness of the Intermediate Semantics against PTSO Declarative Semantics

**Theorem 4** (soundness). For all \(P, S, M, \mathcal{H} = (\pi_{n-1}, \pi'_{n-1}), \ldots, (\pi_1, \pi'_1), \pi_n\) and \(\pi'_n = \varepsilon\):

\[
P, S_0, M_0, PB_0, B_0, e, \varepsilon \Rightarrow^* \text{skip} \cdots \| \text{skip}, S, M, PB_0, B_0, \mathcal{H}, \pi_n
\]

then

1. \(P, S_0, \varepsilon, \varepsilon \Rightarrow^* \text{skip} \cdots \| \text{skip}, S, \Gamma, \pi_n\) where

   \[
   \Gamma = \Gamma_n
   \]

   \[
   \Gamma_i = \varepsilon \quad \Gamma_j = (G_j, (\pi_j', \pi_j)) \quad \text{for } j \in \{1 \cdots n-1\}
   \]

   \[
   G_i = \text{getG}(\Gamma_i, \pi_i, \pi'_i) \quad \text{for } i \in \{1 \cdots n\}
   \]

2. \(E = G_1; \cdots; G_n\) is PTSO-valid.

**Proof.** Pick arbitrary \(P, S, M, \mathcal{H} = (\pi_{n-1}, \pi'_{n-1}), \ldots, (\pi_1, \pi'_1), \pi_n\) such that

\[
P, S_0, M_0, PB_0, B_0, e, \varepsilon \Rightarrow^* \text{skip} \cdots \| \text{skip}, S, M, PB_0, B_0, \mathcal{H}, \pi_n
\]

and let \(\pi'_n = \varepsilon\). The proof of the first part follows from Lemma A.1 and by induction on the length of the event-annotated transition \(\Rightarrow^*\).

For the second part, for all \(i \in \{1 \cdots n\}\) let \(E_i = R_i \cup F_i \cup PE_i\) with \(PE_i = W_i \cup U_i \cup PF_i \cup PS_i\). As \(G_i = \text{getG}(\Gamma_i, \pi_i, \pi'_i)\), we know that \(wfp(\pi'_i, \pi_i, \text{hist}(\Gamma_i))\) and \(\text{complete}(\pi'_i, \pi_i)\) hold. It then suffices to show that for all \(i \in \{1 \cdots n\}\) and \(E_i = (E_i^0, E_i^P, E_i, po_i, rf_i, tso_i, nvo_i)\):

\[
E_i^0 \subseteq E_i^P
\]

\[
E_i^P \subseteq E_i
\]

\[
E_i^0 \times (E_i \setminus E_i^0) \subseteq po_i
\]

\[
E_i^0 \times (E_i \setminus E_i^0) \subseteq tso_i
\]

\[
E_i^0 \times (E_i \setminus E_i^0) \subseteq nvo_i
\]

\[
\text{dom}(nvo_i; [E_i^P]) \subseteq E_i^P \text{ and } E_i^P = E_n
\]

\[
E_0^0 = \{ \text{init}_{x} \mid x \in \text{Loc} \} \text{ and } E_{i+1}^0 = \left\{ \max \left( nvo_i[E_i^P \cap (U_i \cup W_i)] \right) \mid x \in \text{Loc} \right\}
\]

\[
R_i \cup F_i \cup PS_i \subseteq E_i^P \text{ and } po_i; [PS_i] \subseteq E_i^P
\]

\[
poi_i \text{ is a strict total order on } E_i
\]
\[ rf_i \subseteq (W_i \cup U_i) \times (R_i \cup U_i) \] and is total and functional on \( R_i \cup U_i \) \hspace{3cm} (10)

\[ tso_i \subseteq E_i \times E_i \] and is total on \( E_i \setminus R_i \) \hspace{3cm} (11)

\[ po_i \setminus (W_i \times R_i) \subseteq tso_i \] \hspace{3cm} (12)

\[ rf_i \subseteq tso_i \cup po_i \] \hspace{3cm} (13)

\[ \forall (w, r) \in rf_i, \forall w' \in W_i \cup U_i, \] \hspace{3cm} (14)

\[ (w', r) \in tso_i \cup po_i \land \text{loc}(w') = \text{loc}(r) \Rightarrow (w, w') \notin tso_i \] \hspace{3cm} (15)

\[ \text{nvo}_i \text{ is a strict total order on } PE_i \] \hspace{3cm} (16)

\[ \forall x \in \text{Loc. } (\text{nvo}_i)_x \subseteq tso_i \] \hspace{3cm} (17)

\[ [PS_i]; tso_i; [PE_i] \cup [PE_i]; tso_i; [PS_i] \subseteq \text{nvo}_i \] \hspace{3cm} (18)

The proofs of parts (1), (3), (4), (5), (7), and (9) follow immediately from the construction of \( G_i \).

\section*{RTE. (2)}

Pick an arbitrary \( e \in E_i^P \). We then know there exist \( \lambda \in \pi_i \) and \( e \) such that \( e = \text{getPE}(\lambda) \) and either \( \lambda = R(e, -) \), or \( \lambda = F(e) \), or \( \lambda = \text{PS}(e) \), or \( \lambda = \text{PB}(e) \). In the first three cases, from the definition of getE(.) we know that \( e = \text{getE}(\lambda) \) and thus from the definition of \( E_i \) we have \( e \in E_i \), as required.

In the last case, from \( \text{wfp}(\pi'_i, \pi_i, \text{hist}(\Gamma_i)) \) we know that there exists \( w \) such that either \( W(e) \in \pi_i \), or \( U(e, w) \in \pi_i \), or \( \text{PF}(e) \in \pi_i \). As such, from the definition of \( E_i \) we have \( e \in E_i \), as required.

\section*{RTE. (6)}

Pick an arbitrary \( e_1, e_2 \) such that \( (e_1, e_2) \in \text{nvo}_i \) and \( e_2 \in E_i^P \). From the definition of \( \text{nvo}_i \) we then know there exist \( \lambda_1, \lambda_2 \in \pi'_i, \pi_i \) such that \( e_1 = \text{getPE}(\lambda_1) \), \( e_2 = \text{getPE}(\lambda_2) \) and \( \lambda_1 <_{\pi'_i, \pi_i} \lambda_2 \). On the other hand, from the definition of \( E_i^P \) and since \( e_2 \in E_i^P \) we know that \( \lambda_2 \in \pi_i \). As such, since \( \lambda_1 <_{\pi'_i, \pi_i} \lambda_2 \) and labels in \( \pi'_i, \pi_i \) are fresh (\( \text{wfp}(\pi'_i, \pi_i, \text{hist}(\Gamma_i)) \) holds), we also know that \( \lambda_1 \in \pi_i \).

Consequently, since \( e_1 = \text{getPE}(\lambda_1) \) and \( \lambda_1 \in \pi_i \), from the definition of \( E_i^P \) we have \( e_1 \in E_i^P \), as required.

To demonstrate that \( E_i^P = E_n \), it suffices to show that \( E_i \subseteq E_n \), as in part (2) we established that \( E_i^P \subseteq E_n \). Pick arbitrary \( e \in E_n \). From the definition of \( E_n \) we then know there exists \( \lambda \in \pi_n \) such that \( \text{getE}(\lambda) = e \). There are then two cases to consider: 1) \( e \notin W_n \cup U_n \cup \text{PF}_n \); or 2) \( e \in W_n \cup U_n \cup \text{PF}_n \).

In case (1) from the definition of getPE(.) we know that \( \text{getPE}(\lambda) = e \) and thus \( e \in E_n^P \), as required.

In case (2) from complete(\( \pi_n \), \( \pi_n \)) we know that there exists \( \lambda' \) such that \( \lambda' = \text{PB}(e) \) and \( \lambda' \in \pi_n, \pi_n \). As \( \pi_n = e \) we know that \( \lambda' \in \pi_n \). As such, from the definition of getPE(.) we know that \( \text{getPE}(\lambda') = e \) and thus \( e \in E_n^P \), as required.

\section*{RTE. (8)}

The proof of the first part follows immediately from the definitions of \( E_i^P \) and getPE(.). For the second part, pick an arbitrary \( (e, ps) \in \text{po}_i; [PS_i] \), i.e. \( (e, ps) \in \text{po}_i \) and \( ps \in PS_i \). From the definition of \( \text{po}_i \) we then know there exist \( \lambda, \lambda' \in \pi_i \) such that \( e = \text{getE}(\lambda), \lambda' = \text{PS}(ps), ps = \text{getE}(\lambda'), \lambda <_{\pi_i} \lambda' \), and \( \text{tid}(e) = \text{tid}(ps) \). There are now two cases to consider: 1) \( e \notin U_i \cup W_i \cup PF_i \); or 2) \( e \in U_i \cup W_i \cup PF_i \).

In case (1) from the definition of getPE(.) we have \( \text{getPE}(\lambda) = e \) and thus the definition of \( E_i^P \) we have \( e \in E_i^P \), as required.

In case (2) from \( \text{wfp}(\pi'_i, \pi_i, \text{hist}(\Gamma_i)) \) we know there exists \( \lambda'' = \text{ PB}(e) \) such that \( \lambda <_{\pi_i} \lambda'' <_{\pi_i} \lambda' \). That is, \( \lambda'' \in \pi_i \). As such, such from the definition of \( E_i^P \) we have \( e \in E_i^P \), as required.
To demonstrate that \( \mathsf{rf}_i \subseteq (W_i \cup U_i) \times (R_i \cup U_i) \), pick an arbitrary \((e_w, e_r) \in \mathsf{rf}_i\). From the definition of \( \mathsf{rf}_i \) we then know there exists \( \lambda \in \pi_i \) such that \( \lambda = R(e_r, e_w) \) or \( \lambda = U(e_r, e_w) \). As such from the type of annotated labels we know \( e_r \in R \cup U \) and \( e_w \in W \cup U \).

To demonstrate that \( \mathsf{rf}_i \) is total on \( R_i \), pick an arbitrary \( r \in R_i \). Form the definition of \( E_i \) we then know there exist \( \lambda \in \pi_i \) and \( e \) such that \( \lambda = R(r, e) \). As such we know \((e, r) \in \mathsf{rf}_i\) and thus \( \mathsf{rf}_i \) is total on \( R_i \). The proof of \( \mathsf{rf}_i \) being total on \( U_i \) is analogous and omitted here.

To show \( \mathsf{rf}_i \) is functional on \( R_i \), pick an arbitrary \( r \in R_i \). Form the definition of \( E_i \) we know there exists \( \lambda \in \pi_i \) and \( e \) such that \( \lambda = R(r, e) \). As such we know \((e, r) \in \mathsf{rf}_i\) and thus \( \mathsf{rf}_i \). Moreover, since \( \pi_i \) contains unique labels \((\mathsf{wfp}(\pi_i, \mathsf{hist}(I_i))\) holds), we know \( \forall e' \neq e. \ R(r, e') \notin \pi_i \) and thus \( e' \neq e. \) \((e', r) \notin \mathsf{rf}_i\). That is, \( \mathsf{rf}_i \) is functional on \( R_i \). The proof of \( \mathsf{rf}_i \) being functional on \( U_i \) is analogous and omitted here.

To demonstrate that \( \mathsf{tsq}_i \subseteq E_i \times E_i \), pick an arbitrary \((e_1, e_2) \in \mathsf{tsq}_i\). From the definition of \( \mathsf{tsq}_i \) we then know there exists \( \lambda_1, \lambda_2 \in \pi_i \) such that \( e_1 = \mathsf{getBE}(\lambda_1) \) and \( e_2 = \mathsf{getBE}(\lambda_2) \). For \( j \in \{1, 2\} \), we then know that either 1) \( \neg \exists e. \ \lambda_j = B(e); \) or 2) \( \lambda_j = B(e_j) \). In case (1) since \( \pi_i \) is \( \mathsf{PP} \), we know that \( \lambda_j \in \pi_i \) and thus from the definition of \( E_i \) we know \( e_j \in E_i \). In case (2) from \( \mathsf{wfp}(\pi_i, \mathsf{hist}(I_i)) \) we know that \( W(e_j) \in \pi_i \). As such, since \( \pi_i \) is \( \mathsf{PP} \), we know that \( W(e_j) \in \pi_i \) and thus from the definition of \( E_i \) we have \( e_j \in E_i \). As such, in both cases we have \((e_1, e_2) \in E_i \times E_i\), as required.

Transitivity and strictness of \( \mathsf{tsq}_i \) follow from the definition of \( \mathsf{tsq}_i \), transitivity and strictness of \( \mathsf{wfp}(\pi_i, \mathsf{hist}(I_i)) \) holds.

To demonstrate that \( \mathsf{tsq}_i \) is total on \( E_i \setminus R_i \), pick arbitrary \( e_1, e_2 \in E_i \setminus R_i \) such that \( e_1 \neq e_2 \). For \( j \in \{1, 2\} \), from the definitions of \( E_i \) we know there exist \( \lambda_j \in \pi_i \) such that either 1) \( e_j \in E_i \setminus (R_i \cup W_i) \) and \( \lambda_j = \mathsf{getE}(\lambda_j); \) or 2) \( e_j \in W_i \) and \( \lambda_j = W(e_j) \). In case (1) we then have \( \lambda_j \in \pi_i \) and \( \mathsf{getBE}(\lambda_j) = e_j \). In case (2) from \( \mathsf{complete}(\pi_i, \pi_i) \) we then know there exists \( \lambda'_j = B(e_j) \in \pi_i \) and \( \mathsf{getBE}(\lambda'_j) = e_j \). As such, in both cases we know there exist \( \lambda_1, \lambda_2 \in \pi_i \) such that \( e_1 = \mathsf{getBE}(\lambda_1) \) and \( e_2 = \mathsf{getBE}(\lambda_2) \). As \( e_1 \neq e_2 \) and \( \pi_i \) contains fresh labels \((\mathsf{wfp}(\pi_i, \mathsf{hist}(I_i)) \) holds), we know that \( \lambda_1 \neq \lambda_2 \) and thus either \( \lambda_1 <_{\pi_i} \lambda_2 \) or \( \lambda_2 <_{\pi_i} \lambda_1 \). As such, from the definition of \( \mathsf{tsq}_i \) we have either \((e_1, e_2) \in \mathsf{tsq}_i\) or \((e_1, e_2) \in \mathsf{tsq}_i\), as required.

Pick an arbitrary \((e_1, e_2) \in \mathsf{po}_i \setminus (W_i \times R_i)\). From the definition of \( \mathsf{po}_i \) we then know there exist \( \tau \) and \( \lambda_1, \lambda_2 \in \pi_i \) such that \( e_1 = \mathsf{getE}(\lambda_1), e_2 = \mathsf{getE}(\lambda_2), \mathsf{tid}(e_1) = \mathsf{tid}(e_2) = \tau \) and \( \lambda_1 <_{\pi_i} \lambda_2 \). That is, \( \lambda_1 <_{\pi_i} \lambda_2 \). There are then three cases to consider: 1) \( e_1 \notin W_i \) or 2) \( e_2 \notin W_i \) and \( e_1 \in W_i \); or 3) \( e_1 \in W_i \).

In case (1) from the definition of \( \mathsf{getBE}(\cdot) \) we know that \( e_1 = \mathsf{getBE}(\lambda_1), e_2 = \mathsf{getBE}(\lambda_2) \). As such, from the definition of \( \mathsf{tsq}_i \) we have \((e_1, e_2) \in \mathsf{tsq}_i\).

In case (2), from the definition of \( \mathsf{getBE}(\cdot) \) we know that \( e_1 = \mathsf{getBE}(\lambda_1) \). On the other hand, from \( \mathsf{wfp}(\pi_i, \mathsf{hist}(I_i)) \) and \( \mathsf{complete}(\pi_i, \pi_i) \) we know there exists \( \lambda = B(e_2) \) such that \( \lambda_2 <_{\pi_i} \lambda \). That is, \( e_2 = \mathsf{getBE}(\lambda) \). Since we also have \( \lambda_1 <_{\pi_i} \lambda_2 \), from the transitivity of \( < \) we have \( \lambda_1 <_{\pi_i} \lambda \). As such, from the definition of \( \mathsf{tsq}_i \) we have \((e_1, e_2) \in \mathsf{tsq}_i\), as required.

In case (3), there are three additional cases to consider: 1) \( \lambda_2 = F(e_2) \) or \( \lambda_2 = P(e_2) \) or \( \lambda_2 = U(e_2, \cdot) \); or ii) \( \lambda_2 = W(e_j) \); or iii) \( \lambda_2 = P\mathcal{S}(e_j) \).

In case (3.i) from the definition of \( \mathsf{getBE}(\cdot) \) we know that \( e_2 = \mathsf{getBE}(\lambda_2) \). On the other hand, from \( \mathsf{wfp}(\pi_i, \mathsf{hist}(I_i)) \) and \( \mathsf{complete}(\pi_i, \pi_i) \) we know there exists \( \lambda = B(e_1) \) such that \( \lambda_1 <_{\pi_i} \lambda_1 \).
In case (3.iii) from $\text{wfp}(\pi'_i, \pi_i, \text{hist}(\Gamma_i))$ and complete($\pi'_i, \pi_i$) we know there exist $\lambda'_1 = B(e_1)$ and $\lambda'_2 = B(e_2)$ such that $\lambda'_1 <_{\pi'_i, \pi_i} \lambda'_2$. That is, $e_1 = \text{getBE}(\lambda'_1)$ and $e_2 = \text{getBE}(\lambda'_2)$. As such, from the definition of $tso_i$ we have $(e_1, e_2) \in tso_i$, as required.

In case (3.iii) from the definition of $\text{BE}(\lambda_i)$ we know that $e_2 = \text{getBE}(\lambda_2)$. On the other hand, from $\text{wfp}(\pi'_i, \pi_i, \text{hist}(\Gamma_i))$ and complete($\pi'_i, \pi_i$) and since $\text{tid}(e_1) = \text{tid}(e_2)$, we know there exist $\lambda'_1 = B(e_1)$ such that $\lambda_1 <_{\pi'_i, \pi_i} \lambda'_1 <_{\pi'_i, \pi_i} \lambda_2$. That is, $e_1 = \text{getBE}(\lambda'_1)$. As such, from the definition of $tso_i$ we have $(e_1, e_2) \in tso_i$, as required.

**RTS. (13)**

Pick arbitrary $(w, r) \in \text{rf}_i$. From the construction of $\text{rf}_i$ we then know there exist $\lambda \in \pi_i$ such that either $\lambda = R(r, w)$ or $\lambda = U(r, w)$. From $\text{wfp}(\pi'_i, \pi_i, \text{hist}(\Gamma_i))$ we then know that either (1) $B(w) <_{\pi_i} r$; or (2) $U(w, -) <_{\pi_i} r$; or (3) $W(w) <_{\pi_i} r$ and $\text{tid}(w) = \text{tid}(r)$; or (4) $w \in E^0_i$. In cases (1-2) from the definition of $tso_i$ we have $(w, r) \in tso_i$, as required. In cases (3-4) from the definition of $po_i$ we have $(w, r) \in po_i$, as required.

**RTS. (14)**

Pick arbitrary $(w, r) \in \text{rf}_i$ and $w' \in U_i \cup W_i$ such that $(w', r) \in tso_i \cup po_i$ and $\text{loc}(w') = \text{loc}(r)$. If $w' = w$, from the strictness of $tso_i$ we immediately know that $(w, w') \notin tso_i$, as required.

Now let us consider the case where $w' \neq w$. From the construction of $\text{rf}_i$ we then know there exist $\lambda \in \pi_i$ such that either $\lambda = R(r, w)$ or $\lambda = U(r, w)$. From $\text{wfp}(\pi'_i, \pi_i, \text{hist}(\Gamma_i))$ we then know that either (i) there exists $\lambda = B(w) <_{\pi_i} \lambda_r$; or (ii) there exists $\lambda = U(w, -) <_{\pi_i} \lambda_r$; or (iii) $w \in E^0_i$. In cases (1.1), (1.2), (2.1), (3.1), (3.2) we have $\lambda' <_{\pi_i} \lambda$. Consequently, in cases (1.1), (1.2), (2.1), (3.1) from the definition of $tso_i$ we have $(w', w) \in tso_i$, i.e. $(w', w) \notin tso_i$, as required. In cases (3.2) and (3.3) from $\text{wfp}(\pi'_i, \pi_i, \text{hist}(\Gamma_i))$ and complete($\pi'_i, \pi_i$) we additionally know there exist $\lambda'' = B(w)$ such that $\lambda <_{\pi'_i, \pi_i} \lambda''$ and thus from the transitivity of $<$ we have $\lambda' <_{\pi'_i, \pi_i} \lambda''$. Consequently, from the definition of $tso_i$ we have $(w', w) \in tso_i$, i.e. $(w', w) \notin tso_i$, as required.

In cases (2.2), (3.3) from the definition of $tso_i$ we have $(w', w) \in tso_i$, i.e. $(w', w) \notin tso_i$, as required. Similarly, in case (1.1) from $\text{wfp}(\pi'_i, \pi_i, \text{hist}(\Gamma_i))$ we know $W(w) \in \pi_i$ and thus from the definition of $tso_i$ we have $(w', w) \in tso_i$, i.e. $(w, w') \notin tso_i$, as required.

Cases (4.1), (4.2) cannot arise as from $\text{wfp}(\pi'_i, \pi_i, \text{hist}(\Gamma_i))$ we arrive at a contradiction. Case (4.3) cannot arise as $w \neq w'$ and from the definition of $E^0_i$ we cannot have two distinct events of the same location in $E^0_i$.

**RTS. (15)**

Transitivity and strictness of $\text{nvo}_i$ follow from the definition of $\text{nvo}_i$, transitivity and strictness of $<_{\pi'_i, \pi_i}$ and the freshness of events in $\pi'_i, \pi_i$ ($\text{wfp}(\pi'_i, \pi_i, \text{hist}(\Gamma_i))$) holds.

To demonstrate that $\text{nvo}_i$ is total on $\text{PE}_i$, pick arbitrary $e_1, e_2 \in \text{PE}_i$ such that $e_1 \neq e_2$. For $j \in \{1, 2\}$, from the definitions of $\text{PE}_i$ we know there exist $\lambda_j \in \pi_i$ such that either (1) $e_j \in U_i$ and $\lambda_j = U(e_j, -)$; or (2) $e_j \in W_i$ and $\lambda_j = W(e_j)$; or (3) $e_j \in \text{PF}_i$ and $\lambda_j = \text{PF}(e_j)$; or (4) $e_j \in \text{PS}_i$ and $\lambda_j = \text{PS}(e_j)$. In cases (1-3) from complete($\pi'_i, \pi_i$) we then know there exists $\lambda'_j = \text{PB}(e_j) \in \pi'_i, \pi_i$ and $\text{getPE}(\lambda'_j) = e_j$. In case (4) we have $\text{getPE}(\lambda_j) = e_j$. As such, in both cases we know there
To demonstrate $[P_S]; tso_i; [P_E] \subseteq \text{nvo}_i$, pick arbitrary $(e_1, e_2) \in [P_S]; tso_i; [P_E]$. From the definition of $tso_i$ we then know that there exist $\lambda_1, \lambda_2 \in \pi'_i \cdot \pi_i$ such that $e_1 = \text{getPE}(\lambda_1)$, $e_2 = \text{getPE}(\lambda_2)$ and $\lambda_1 \prec_{\pi'_i \cdot \pi_i} \lambda_2$. Moreover, since $e_1 \in P_S$ we know that $\text{getPE}(\lambda_1) = e_1$. There are now three cases to consider: 1) $e_2 \notin W_i \cup U_i \cup PF_i$; or 2) $e_2 \in U_i \cup PF_i$; or 3) $e_2 \in W_i$.

In case (1), from the definitions of $\text{getPE}(.)$ and $\text{getBE}(.)$ we know that $\text{getPE}(\lambda_2) = e_2$ and thus from the definition of $\text{nvo}_i$ we have $(e_1, e_2) \in \text{nvo}_i$, as required.

In case (2) from the definition of $\text{getBE}(.)$ we know that either $\lambda_2 = \text{U}(e_2, -)$ or $\lambda_2 = \text{PF}(e_2)$ and thus from $\text{wp}(\pi'_i \cdot \pi_i, \text{hist}(\Gamma_i))$ and $\text{complete}(\pi'_i \cdot \pi_i)$ we know there exists $\lambda = \text{PB}(e_2)$ such that $\lambda_2 \prec_{\pi'_i \cdot \pi_i} \lambda$. Since we also have $\lambda_1 \prec_{\pi'_i \cdot \pi_i} \lambda_2$, from the transitivity of $\prec_{\pi'_i \cdot \pi_i}$ we also have $\lambda_1 \prec_{\pi'_i \cdot \pi_i} \lambda$. Moreover, from the definition of $\text{getPE}(.)$ we have $\text{getPE}(\lambda) = e_2$. Consequently, we have $(e_1, e_2) \in \text{nvo}_i$, as required.

Similarly, in case (3) from the definition of $\text{getBE}(.)$ we know $\lambda_2 = \text{B}(e_2)$ and thus from $\text{wp}(\pi'_i \cdot \pi_i, \text{hist}(\Gamma_i))$ and $\text{complete}(\pi'_i \cdot \pi_i)$ we know there exists $\lambda = \text{PB}(e_2)$ such that $\lambda_2 \prec_{\pi'_i \cdot \pi_i} \lambda$. Since we also have $\lambda_1 \prec_{\pi'_i \cdot \pi_i} \lambda_2$, from the transitivity of $\prec_{\pi'_i \cdot \pi_i}$ we also have $\lambda_1 \prec_{\pi'_i \cdot \pi_i} \lambda$. Moreover, from the definition of $\text{getPE}(.)$ we have $\text{getPE}(\lambda) = e_2$. Consequently, we have $(e_1, e_2) \in \text{nvo}_i$, as required.

To demonstrate $[P_E]; tso_i; [P_S] \subseteq \text{nvo}_i$, pick arbitrary $(e_1, e_2) \in [P_E]; tso_i; [P_S]$. From the definition of $tso_i$ we then know that there exist $\lambda_1, \lambda_2 \in \pi'_i \cdot \pi_i$ such that $e_1 = \text{getBE}(\lambda_1)$, $e_2 = \text{getBE}(\lambda_2)$ and $\lambda_1 \prec_{\pi'_i \cdot \pi_i} \lambda_2$. Moreover, since $e_2 \in P_S$ we know that $\text{getPE}(\lambda_2) = e_2$. There are now four cases to consider: 1) $e_1 \notin W_i \cup U_i \cup PF_i$; or 2) $e_1 \in U_i$; or 3) $e_1 \in W_i$; or 4) $e_1 \in PF_i$.

In case (1), from the definitions of $\text{getPE}(.)$ and $\text{getBE}(.)$ we know that $\text{getPE}(\lambda_1) = e_1$ and thus from the definition of $\text{nvo}_i$ we have $(e_1, e_2) \in \text{nvo}_i$, as required.

In case (2) from the definition of $\text{getBE}(.)$ we know $\lambda_1 = \text{U}(e_1, -)$ and thus from $\text{wp}(\pi'_i \cdot \pi_i, \text{hist}(\Gamma_i))$ and $\text{complete}(\pi'_i \cdot \pi_i)$ we know there exists $\lambda = \text{PB}(e_1)$ such that $\lambda_1 <_{\pi'_i \cdot \pi_i} \lambda <_{\pi'_i \cdot \pi_i} \lambda_2$. Moreover, from the definition of $\text{getPE}(.)$ we have $\text{getPE}(\lambda) = e_1$. Consequently, we have $(e_1, e_2) \in \text{nvo}_i$, as required.

Similarly, in case (3) from the definition of $\text{getBE}(.)$ we know $\lambda_1 = \text{B}(e_1)$ and thus from $\text{wp}(\pi'_i \cdot \pi_i, \text{hist}(\Gamma_i))$ and $\text{complete}(\pi'_i \cdot \pi_i)$ we know there exists $\lambda = \text{PB}(e_1)$ such that $\lambda_1 <_{\pi'_i \cdot \pi_i} \lambda <_{\pi'_i \cdot \pi_i} \lambda_2$. Moreover, from the definition of $\text{getPE}(.)$ we have $\text{getPE}(\lambda) = e_1$. Consequently, we have $(e_1, e_2) \in \text{nvo}_i$, as required.
Analogously, in case (4) from the definition of getBE(·) we know $\lambda_1 = PF(e_1)$ and thus from wfp($\pi'_1, \pi_i, \mathit{hist}()$) and complete($\pi'_1, \pi_i$) we know there exists $\lambda = PB(e_1)$ such that $\lambda_1 \prec \pi'_1, \pi_i$ \( \lambda \prec \pi'_1, \pi_i, \lambda_2 \). Moreover, from the definition of getPE(·) we have getPE($\lambda$) = $e_1$. Consequently, we have $(e_1, e_2) \in \mathit{nvo}$, as required.

RTS. (18)
To demonstrate that $[PE_j]; \mathit{tso}_j; [PF_j] \subseteq \mathit{nvo}$, pick an arbitrary $(e_1, e_2) \in [PE_j]; \mathit{tso}_j; [PF_j]$. If $e_1 \in PS_i$, then the desired result holds immediately from part (17). On the other hand if $e_1 \notin PS_i$, then from the definition of $\mathit{tso}_i$ we then know that that there exist $\lambda_1, \lambda_2 \in \pi'_1, \pi_i$ such that $e_1 = \mathit{getBE}(\lambda_1)$, $e_2 = \mathit{getBE}(\lambda_2)$, $\lambda_2 = PF(e_2)$, $\lambda_1 < \pi'_1, \pi_i$ \( \lambda_2 \) and either 1) $\lambda_1 = B(e_1)$; 2) $\lambda_1 = U(e_1, -)$; or 3) $\lambda_1 = PB(e_1)$. From wfp($\pi'_1, \pi_i, \mathit{hist}()$) and complete($\pi'_1, \pi_i$) we know there exists $\lambda'_2 = PB(e_2)$ such that $\lambda'_2 \prec \pi'_1, \pi_i, \lambda_2$. As such we have getPE($\lambda'_2$) = $e_2$. As $\lambda_2 = PF(e_2)$ and $\lambda_1 < \pi'_1, \pi_i, \lambda_2$, in all three cases from wfp($\pi'_1, \pi_i, \mathit{hist}()$) and complete($\pi'_1, \pi_i$) we know there exist $\lambda'_2 = PB(e_1)$ such that $\lambda'_1 \prec \pi'_1, \pi_i, \lambda'_2$. That is, getPE($\lambda'_1$) = $e_1$. From the definition of $\mathit{nvo}$ we thus have $(e_1, e_2) \in \mathit{nvo}$, as required.

Similarly, to demonstrate that $[PF_j]; \mathit{tso}_i; [PE_j] \subseteq \mathit{nvo}$, pick an arbitrary $(e_1, e_2) \in [PF_j]; \mathit{tso}_i; [PE_j]$. If $e_2 \in PS_i$, then the desired result holds immediately from part (17). On the other hand if $e_2 \notin PS_i$, then from the definition of $\mathit{tso}_i$ we then know that that there exist $\lambda_1, \lambda_2 \in \pi'_1, \pi_i$ such that $e_1 = \mathit{getBE}(\lambda_1), e_2 = \mathit{getBE}(\lambda_2), \lambda_1 = PF(e_1), \lambda_2 < \pi'_1, \pi_i, \lambda_2$ and either 1) $\lambda_2 = B(e_2)$; 2) $\lambda_2 = U(e_2, -)$; or 3) $\lambda_2 = PF(e_2)$. From wfp($\pi'_1, \pi_i, \mathit{hist}()$) and complete($\pi'_1, \pi_i$) we know there exists $\lambda'_1 = PB(e_1) \in \pi'_1, \pi_i$. As such we have getPE($\lambda'_1$) = $e_1$. As $\lambda_1 = PF(e_1)$ and $\lambda_1 < \pi'_1, \pi_i, \lambda_2$, in all three cases from wfp($\pi'_1, \pi_i, \mathit{hist}()$) and complete($\pi'_1, \pi_i$) we know there exist $\lambda'_1 = PB(e_2)$ such that $\lambda'_1 \prec \pi'_1, \pi_i, \lambda'_2$. That is, getPE($\lambda'_2$) = $e_2$. From the definition of $\mathit{nvo}$ we thus have $(e_1, e_2) \in \mathit{nvo}$, as required.

A.3 Completeness of the Intermediate Semantics against PTSO Declarative Semantics

Definition A.3. Let $E = G_1; \cdots; G_n$ denote a PTSO-valid execution chain. Let $S_1 = e$ and $S_{j+1} = G_j; \cdots; G_1$ for $j \in \{1, \cdots, n\}$. For each execution era $G_i$, the set of traces induced by $G_i$, written $\mathit{traces}(G_i, S_i)$, includes those traces $(\pi', \pi)$ that satisfy the following condition:

\[
(\pi'_1, \pi_i) \cdots (\pi'_1, \pi_i) \in \mathit{traces}(G_i, S_i) \iff \bigwedge_{k=1}^{i} \mathit{getG}(\Gamma_k, \pi_k, \pi'_k) = G_k
\]

where $\Gamma_1 = e$ and $\Gamma_{j+1} = (\pi'_1, \pi_i) \cdots (\pi'_1, \pi_i)$ for $j \in \{1, \cdots, i-1\}$.

Lemma A.2. Let $E = G_1; \cdots; G_n$ denote a PTSO-valid execution chain. Let $S_1 = e$ and $S_{j+1} = G_j; \cdots; G_1$ for $j \in \{1, \cdots, n\}$. For an arbitrary PTSO-valid $G_i$, we demonstrate how to construct a trace $s = (\pi'_1, \pi_1) \cdots (\pi'_1, \pi_i)$ such that $s \in \mathit{traces}(G_i, S_i)$.

For each $k \in \{1, \cdots, i\}$ and $G_k = (E^0, E^p, E, po, rf, \mathit{tso}, \mathit{nvo})$, we construct $(\pi'_k, \pi_k)$ as follows. Let $R = \{r_1 \cdots r_q\}$ denote an enumeration of $G_k.R$ and $\{w_1, \cdots, w_s\}$ denote an enumeration of $G_k.W$. For each $j \in \{1, \cdots, q\}$ and $l \in \{0, \cdots, s-1\}$ where $(w, r_j) \in rf$, we then define

\[
\mathit{tso}_{j+1}^{l+1} = \begin{cases}
\mathit{tso}_j \cup \{(r_j, w_{l+1})\} & \text{if } (r_j, w_{l+1}) \notin \mathit{tso}_j \cup (\mathit{tso}_j)^{-1} \\
\mathit{tso}_j & \text{otherwise}
\end{cases}
\]

where $\text{tso}_0^j = \text{tso}$ and $\text{tso}_{i+1}^j = \text{tso}_i^j$ for $j \in \{1 \cdots q-1\}$. Note that each $\text{tso}_j^j$ is 1) total on writes and respects with $\text{tso}$; and 2) is a strict order on $E$. We next show that:

$$\forall j \in \{1 \cdots q\}. \forall I \in \{0 \cdots s\}. \forall w, r, \forall w' \in W \cup U. \forall w' \in W \cup U. (w, r) \in \text{rf} \land (w', r) \in \text{tso}_j^j \cup \text{po} \land 1\text{oc}(w) = 1\text{oc}(w') \Rightarrow (w, w') \notin \text{tso}_j^j \quad \text{(RFJ)}$$

Let $(w, r_j) \in \text{rf}$. We proceed by double induction on $j$ and $l$.

**Base case $j = 1$ and $l = 0$**

As $G_k$ is PTSO-valid, we know that the desired property holds of $\text{tso}$ and thus of $\text{tso}_0^0 = \text{tso}$ by definition.

**Inductive case $j = 1$ and $l = a+1$ with $0 \leq a < s$**

$$\forall I' \in \{1 \cdots a\}. \forall w, r, \forall w' \in W \cup U. (w, r) \in \text{rf} \land (w', r) \in \text{tso}_1^1 \cup \text{po} \land 1\text{oc}(w) = 1\text{oc}(w') \Rightarrow (w, w') \notin \text{tso}_1^1 \quad \text{(I.H.)}$$

From the definition of $\text{tso}_1^1$, we know that either i) $\text{tso}_1^1 = \text{tso}_0^1$; or ii) $\text{tso}_1^1 = \left( \text{tso}_0^q \cup \{(r_1, w_1)\}\right)^+$. In case (ii) we proceed by contradiction. Let us assume there exists $w_c, w'_c, r_c$ such that $(w_c, r_c) \in \text{rf}, (w'_c, r_c) \notin \text{tso}_1^1 \cup \text{po} \land 1\text{oc}(w_c) = 1\text{oc}(w'_c)$ and $(w_c, w'_c) \in \text{tso}_1^1$. As $(w_c, w'_c) \in \text{tso}_1^1$ is a strict order, we know that $w_c \neq w'_c$. On the other hand, from (I.H.) we then know that $(w'_c, r_c) \notin \text{tso}_1^1 \cup \text{po}$.

However, as $\text{tso}_0^1$ is strict and is total on writes, we know that either a) $(w_1, w'_1) \in \text{tso}_0^1$; or b) $(w'_1, w_1) \notin \text{tso}_1^1$. In case (i) we then have $w_1 \xrightarrow{\text{tso}_0^1} w'_1 \xrightarrow{\text{tso}_0^1} r_1$, contradicting the assumption that $(r_1, w_1) \notin \text{tso}_1^1 \cup \text{tso}_0^q$. In case (ii) we have $(w_1, w'_1) \notin \text{tso}_0^1$, i.e. $(w'_1, w_1) \notin \text{tso}_1^1$, and thus $(w_c, w'_c) \notin \text{tso}_1^1$, contradicting our assumption that $(w_c, w'_c) \in \text{tso}_1^1$.

**Inductive case $j = b+1$ and $l = 0$ with $1 \leq b < q-1$**

$$\forall f' \in \{1 \cdots b\}. \forall I' \in \{1 \cdots s\}. \forall w, r, \forall w' \in W \cup U. (w, r) \in \text{rf} \land (w', r) \in \text{tso}_j^j \Rightarrow (w, w') \notin \text{tso}_j^j \quad \text{(I.H.)}$$

As $\text{tso}_b^j \triangleq \text{tso}_b^s$, the desired result holds immediately from (I.H.).

**Inductive case $j = b+1$ and $l = a+1$ with $1 \leq b < q-1$ and $0 \leq a < s$**

$$\forall I' \in \{1 \cdots a\}. \forall w, r, \forall w' \in W \cup U. (w, r) \in \text{rf} \land (w', r) \in \text{tso}_j^j \Rightarrow (w, w') \notin \text{tso}_j^j \quad \text{(I.H.)}$$

From the definition of $\text{tso}_j^j$, we know that either i) $\text{tso}_j^j = \text{tso}_0^1$; or ii) $\text{tso}_j^j = \left( \text{tso}_0^q \cup \{(r_j, w_j)\}\right)^+$. In case (i) we proceed by contradiction. Let us assume there exists $w_c, w'_c, r_c$ such that $(w_c, r_c) \in \text{rf}, (w'_c, r_c) \notin \text{tso}_j^j \cup \text{po} \land 1\text{oc}(w_c) = 1\text{oc}(w'_c)$ and $(w_c, w'_c) \in \text{tso}_j^j$. As $(w_c, w'_c) \in \text{tso}_j^j$ is a strict order, we know that $w_c \neq w'_c$. On the other hand, from (I.H.) we then know that $(w'_c, r_c) \notin \text{tso}_j^j \cup \text{po}$.

As such, form the definition of $\text{tso}_j^j$ we know that $w'_c \xrightarrow{\text{tso}_j^j} r_j \xrightarrow{\text{tso}_j^j} w_l \xrightarrow{\text{tso}_j^j} r_c$.

However, as \( \text{tso}_f \) is strict and is total on writes, we know that either a) \( (w_l, w'_l) \in \text{tso}_f^q \); or b) \( (w'_c, w_l) \in \text{tso}_f^q \). In case (ii.a) we then have \( w_l \rightarrow w'_l \rightarrow r_j \), contradicting the assumption that \( (r_j, w_l) \notin \text{tso}_f^q \cup (\text{tso}_f^q)^{-1} \). In case (ii.b) we have \( w'_c \rightarrow w_l \rightarrow r_j \), i.e. \( (w'_c, r_j) \in \text{tso}_f^q \). As such, from (I.H.) we have \( (w_c, w'_c) \notin \text{tso}_f^q \), i.e. \( (w'_c, w_c) \in \text{tso}_f^q \) \( \subseteq \text{tso}_f^j \), and thus \( (w_c, w'_c) \notin \text{tso}_f^j \), contradicting our assumption that \( (w_c, w'_c) \in \text{tso}_f \).

Let \( \text{tso}_t \) denote an extension of \( \text{tso}_f^q \) to a strict total order on \( E \). Once again, we demonstrate that:

\[
\forall w, r. \forall w' \in W \cup U. (w, r) \in \text{rf} \land (w', r) \in \text{tso}_t \land 1 \text{oc}(w) = 1 \text{oc}(w') \Rightarrow (w, w') \notin \text{tso}_t \quad \text{(RF)}
\]

Pick arbitrary \( w, w', r \) such that \( (w, r) \in \text{rf} \land 1 \text{oc}(w) = 1 \text{oc}(w') \) and \( (w', r) \in \text{tso}_t \). There are two cases to consider: 1) \( (w', r) \in \text{tso}_f^q \); or 2) \( (w', r) \in \text{tso}_t \setminus \text{tso}_f^q \). In case (1) the result holds from (RF) established above. In case (2), as \( \text{tso}_t \) is a strict order we know that \( (r, w') \notin \text{tso}_f^q \) and thus \( (r, w') \notin \text{tso}_t \). Moreover, as \( (w', r) \in \text{tso}_t \setminus \text{tso}_f^q, \) i.e. \( (w', r) \notin \text{tso}_f^q \). As such, from the definition of \( \text{tso}_f^q \) we know that \( (w, w') \notin \text{tso}_f \), i.e. \( (w', w) \in \text{tso}_f \subseteq \text{tso}_t \). As \( \text{tso}_t \) is a strict order, we have \( (w, w') \notin \text{tso}_t \).

Let \( \{e_1, \ldots, e_n\} \) denote an enumeration of \( G_k. E \setminus E^0 \) that respects \( \text{tso}_f^q \); \( \{w_1, \ldots, w_m\} \) denote an enumeration of \( G_k. W \setminus E^0 \) that respects \( \text{tso}_f \); and \( \{e'_1, \ldots, e'_p\} \) denote an enumeration of \( G_k. (W \cup U \cup \text{PF}) \setminus E^0 \) that respects \( \text{tso}_f \). Since \( G_k \) is PTSO-valid and thus \( \text{dom} (\text{tso}_f \setminus E^0) \subseteq E^0 \), we know there exists \( p \) such that \( 0 \leq p \leq 0 \) and \( \{e'_1, \ldots, e'_p\} \in E^P \setminus E^0 \) and \( \{e'_p+1, \ldots, e'_p\} \in E \setminus (E^P \cup E^0) \).

Let \( \lambda^0 = \lambda_n, \ldots, \lambda_1 \), where \( \lambda_j = \text{genBL}(e_j, G_k) \) for \( j \in \{1 \cdots n\} \) and:

\[
\text{genBL}(e, G) \triangleq \begin{cases} 
B(e) & \text{if } e \in G.W \\
\text{genL}(e, G) & \text{if } e \in G.E \setminus G.W \\
\text{undefined} & \text{otherwise}
\end{cases}
\]

For each \( j \in \{1 \cdots m\} \), let \( N_j = \{ e \mid (w_j, e) \in \text{po} \land e \notin \{w_{j+1}, \ldots, w_m\} \} \); and \( n_j = \min(\text{po}|_{N_j}) \) when such an element exists. For each \( j \in \{1 \cdots m\} \), let \( \pi_j = \text{addw}(\pi^{j-1}, w_j, n_j) \), where:

\[
\text{addw}(\pi, w, n) \triangleq \begin{cases} 
W(w).B(w).s & \text{if } \exists s. \pi = B(s).w \\
W(w).n.s & \text{if } \exists s. \pi = \text{genL}(n, G_k).s \\
e.addw(s, w, n) & \text{if } \exists s. \pi = e.s \\
\text{undefined} & \text{otherwise}
\end{cases}
\]

\[
\text{genL}(e, G) \triangleq \begin{cases} 
R(e, e') & \text{if } e \in G.R \land (e', e) \in G.Rf \\
W(e) & \text{if } e \in G.W \\
U(e, e') & \text{if } e \in G.U \land (e', e) \in G.Rf \\
F(e) & \text{if } e \in G.F \\
PF(e) & \text{if } e \in G.PF \\
PS(e) & \text{if } e \in G.PS \\
\text{undefined} & \text{otherwise}
\end{cases}
\]

Note that for all \( j \in \{1 \cdots m\} \), the \( \text{addw}(\pi^{j-1}, w_j, n_j) \) is always defined as \( B(w_j) \in \pi^{j-1} \).
Let \( \pi \) be a state of the system, and \( e \) be an event. Let \( \pi' = \pi \cup \{ e \} \). Let \( \pi'' = \pi' \cup \{ e \} \). Then \( \pi'' \) can be considered as the result of applying \( e \) to \( \pi' \).

\[
\begin{align*}
\text{addP}(\pi, e, p) & = \\
\begin{cases} 
\text{undefined} & \text{otherwise} \\
\text{if } \exists s. \pi = s_{\text{genPL}}(e, Gi) \\
\text{if } \exists s. \pi = s_{\text{genPL}}(p, Gi) \\
\text{undefined} \\
\text{otherwise} \\
\end{cases}
\end{align*}
\]

\[
\text{genPL}(e, G) \triangleq \\
\begin{cases} 
\text{undefined} & \text{otherwise} \\
\text{if } e \in G.(W \cup U \cup PF) \\
\text{genL}(e, G) & \text{if } e \in G.PS \\
\end{cases}
\]

Note that for all \( j \in \{1, \ldots, p\} \), the \( \text{addP}(\pi_{j-1}, e'_j, p_j) \) is always defined as \( \text{genPL}(e'_j, G_k) \in \pi_{j-1} \). Let \( \pi_k = \pi_{k-1} \) and let \( \pi'_k = \text{genPL}(e_0, G_k) \ldots \text{genPL}(e_{p+1}, G_k) \).

We next demonstrate that \( \text{wfp}(\pi_{k-1}, \pi_k, \text{hist}(\Gamma_k)) \) and complete(\( \pi_{k-1}, \pi_k \)) hold.

**Goal:** \( \text{wfp}(\pi_{k-1}, \pi_k, \text{hist}(\Gamma_k)) \)

Let \( \pi = \pi_{k-1} \). We are then required to show that for all \( \lambda, \pi_1, \pi_2, e, r, e_1, e_2 \):

\[
\text{nodup}(\pi, \pi'') \quad (19)
\]

\[
\pi = \pi_2 \cdot R(r, e).\pi_1 \lor \pi = \pi_2 \cdot U(r, e).\pi_1 \Rightarrow \text{wfrd}(r, e, \pi_1, \pi'') \quad (20)
\]

\[
B(e) \in \pi \Rightarrow W(e) <_\pi B(e) \quad (21)
\]

\[
\text{PB}(e) \in \pi \Rightarrow \\
(B(e) <_\pi \text{PB}(e) \lor U(e, -) <_\pi \text{PB}(e) \lor PF(e) <_\pi \text{PB}(e)) \quad (22)
\]

\[
\text{tid}(e_1) = \text{tid}(e_2) \Rightarrow \\
B(e_1) \in \pi \land W(e_1) <_\pi W(e_2) \iff B(e_1) <_\pi B(e_2) \quad (23)
\]

\[
W(e_1) <_\pi F(e_2) \land \text{tid}(e_1) = \text{tid}(e_2) \Rightarrow B(e_1) <_\pi F(e_2) \quad (24)
\]

\[
W(e_1) <_\pi U(e_2, e) \land \text{tid}(e_1) = \text{tid}(e_2) \Rightarrow B(e_1) <_\pi U(e_2, e) \quad (25)
\]

\[
W(e_1) <_\pi \text{PF}(e_2) \land \text{tid}(e_1) = \text{tid}(e_2) \Rightarrow B(e_1) <_\pi \text{PF}(e_2) \quad (26)
\]

\[
W(e_1) <_\pi \text{PS}(e_2) \land \text{tid}(e_1) = \text{tid}(e_2) \Rightarrow B(e_1) <_\pi \text{PS}(e_2) \quad (27)
\]

\[
\text{loc}(e_1) = \text{loc}(e_2) \land e_1, e_2 \in W \cup U \Rightarrow \\
\text{PB}(e_2) \in \pi \land \left( B(e_1) <_\pi B(e_2) \land (U(e_1, e, -) <_\pi B(e_2) \lor U(e_1, e_2, e) <_\pi B(e_2) \lor U(e_1, e_2, -) <_\pi U(e_2, e)) \right) \iff \text{PB}(e_1) <_\pi \text{PB}(e_2) \quad (28)
\]

\[
e_1, e_2 \in (PE \times PE) \setminus (W \cup U \times W \cup U) \Rightarrow \\
\text{PB}(e_2) \in \pi \land \left( B(e_1) <_\pi \text{PF}(e_2) \land U(e_1, e, -) <_\pi \text{PF}(e_2) \land U(e_1, e_2, e) <_\pi \text{PF}(e_2) \land U(e_1, e_2, -) <_\pi U(e_2, e) \land \text{PF}(e_1) <_\pi \text{PF}(e_2) \right) \iff \text{PB}(e_1) <_\pi \text{PB}(e_2) \quad (29)
\]

\[
\text{PB}(e_2) \in \pi \land \left( B(e_1) <_\pi \text{PS}(e_2) \land U(e_1, e, -) <_\pi \text{PS}(e_2) \land U(e_1, e_2, e) <_\pi \text{PS}(e_2) \land U(e_1, e_2, -) <_\pi U(e_2, e) \land \text{PF}(e_1) <_\pi \text{PF}(e_2) \right) \iff \text{PB}(e_1) <_\pi \text{PB}(e_2) \quad (30)
\]

where $\pi'' = \pi_{k-1} \cdots \pi_1$ and $\pi''' = \pi_{k-1}' \cdots \pi_1'$.

The proof of parts (19), (21), (22) follow immediately from the constructions of $\pi_k'$ and $\pi_k$.

For part (20), pick arbitrary $\pi_1, \pi_2, r, e$ such that $\pi = \pi_2.R(r, e).\pi_1$ or $\pi = \pi_2.U(r, e).\pi_1$. From the construction of $\pi$ we then know that $(e, r) \in rf$. There are now two cases to consider: 1) $e \in E \setminus E^0$; or 2) $e \in E^0$.

In case (1), as $G_k$ is PTSO-valid, we know that $(e, r) \in rf \subseteq tso.U.po$. As such, from the construction of $\pi$ we know that there exists $\pi_3$ such that $\pi_1 = \pi_3.\lambda._-$ and $\lambda = B(e) \lor \lambda = U(e, -) \lor (\lambda = W(e) \land t_{id}(e) = t_{id}(r))$. There are two more cases to consider: i) $\lambda = B(e) \lor \lambda = U(e, -)$; or ii) $\lambda = W(e)$.

In case (i) let us assume there exists $e'$ such that $1oc(e') = 1oc(r)$ and $B(e') \in \pi_3$ or $U(e', -) \in \pi_3$. From the construction of $\pi$ we then have $e' \in W \cup U, (e', r) \in tso$, and $(e', e') \in tso$. This however contradicts our result in (RF) and thus we have $\{B(e'), U(e', -) \in \pi_3 \mid 1oc(e') = 1oc(r)\} = \emptyset$, as required. Similarly, let us assume there exists $e'$ such that $1oc(e') = 1oc(r)$, $\lambda = W(e)$, and $W(e') - \pi_3$ and $B(e') \notin \pi_3$. From the construction of $\pi$ we then have $e' \in W \cup U, (e', r) \in po$ and $(e', e') \in po \cap (W \cup U) \times (W \cup U) \subseteq tso$. This however contradicts our result in (RF) and thus we have $\emptyset = \emptyset$, as required.

In case (2), as $G_k$ is PTSO-valid, we know either i) $k = 1 \land e = init_{1oc(e)}$; or ii) $k > 0 \land e = max(G_{k-1}.nvo| G_{k-1}.E^0(n|U_{1oc(e)}\cup W_{1oc(e)})$. Let us now assume there exists $e'$ such that $B(e') \in \pi_1$ or $U(e', -) \in \pi_1$, and $1oc(e') = 1oc(r)$. That is, $e' \in W \cup U$. From the construction of $\pi$ we then have $\{e', r \in tso$, and $(e', e') \in tso$. This however contradicts our result in (RF) and thus we have $\emptyset = \emptyset$, as required. Similarly, let us assume there exists $e'$ such that $1oc(e') = 1oc(r)$, $\lambda = W(e)$, and $W(e') \in \pi_1$. That is, $e' \in W \cup U$. From the construction of $\pi$ we then have $e', r \in po$ and $(e', e') \in po \cap (W \cup U) \times (W \cup U) \subseteq tso$. This however contradicts our result in (RF) and thus we have $\emptyset = \emptyset$, in case (i), as $\Gamma_k = e$, we know $\pi'' = e$ and thus we simply have $\{PB(e') \in \pi'' \mid 1oc(e') = 1oc(r)\} = \emptyset$ as required.

In case (ii), we then know either:

a) for all $b \in \{1 \cdots k-1\}, e \in G_b.E^0$ and $G_b.(W \cup U)_{1oc(e)} \setminus E^0 = \emptyset$ and thus $e = init_{1oc(e)}$; or

b) there exists $a \in \{1 \cdots k-1\}$ such that $e \in G_a.E^0 \setminus E^0, \forall e' \in G_a.(W \cup U)_{1oc(e)}$. $(e', e) \in G_a.nvo$ and for all $b \in \{a+1 \cdots k-1\}, e \in G_b.E^0$ and $G_b.(W \cup U)_{1oc(e)} \setminus E^0 = \emptyset$.

In case (a), let us assume there exists $e'$ such that $PB(e') \in \pi''$ and $1oc(e') = 1oc(r) = 1oc(e)$. We then know there exists $b \in \{1 \cdots k-1\}$ such that $e \in G_b.(W \cup U)_{1oc(e)} \setminus E^0$, leading to a contradiction. As such, we have $\{PB(e') \in \pi'' \mid 1oc(e') = 1oc(r)\} = \emptyset$ as required.

In case (b), from the construction of $\pi_1 \cdots \pi_{k-1}$, we know there exists $\pi_3, \pi_4$ such that $\pi_4 = \pi_3.PB(e).\pi_4$, and $\pi'' = \pi_{k-1}' \cdots \pi_a' \cdots \pi_1'$. Let us assume there exists $e'$ such that $PB(e') \in \pi_{k-1}' \cdots \pi_a' \cdots \pi_1'$ and $1oc(e') = 1oc(r) = 1oc(e)$. We then know either there exists $b \in \{k-1 \cdots a+1\}$ such that $e \in G_b.(W \cup U)_{1oc(e)} \setminus E^0$, leading to a contradiction. Similarly, let us assume there exists...
\(\epsilon\) such that \(\text{PB}(\epsilon'^r) \in \pi_3\) and \(1\text{oc}(\epsilon') = 1\text{oc}(r) = 1\text{oc}(\epsilon)\). We then know \((\epsilon, \epsilon') \in G_{a,nvo}\), leading to a contradiction. As such, we have \(\{\text{PB}(\epsilon'^r) \in \pi_{k-1}, \ldots, \pi_{a+1}, \pi_3 \mid 1\text{oc}(\epsilon') = 1\text{oc}(r)\} = \emptyset\), as required.

For part (23), pick arbitrary \(e_1, e_2\) such that \(\text{tid}(e_1) = \text{tid}(e_2)\). For the \(\Rightarrow\) direction assume \(W(e_1) \prec_\pi W(e_2)\). Moreover, from the construction of \(\pi\) we know that for all \(\epsilon\) such that \(\text{tid}(\epsilon) = \text{tid}(e_1)\) we have \((e_1, e) \in \text{po} \iff W(e_1) \prec_\pi \text{genL}(e, G_k)\). As such, we have \((e_1, e_2) \in \text{po}\). As \(G_k\) is PTSO-valid, we then know that \((e_1, e_2) \in \text{tso}\). Consequently, from the construction of \(\pi\) we have \(B(e_1) \prec_\pi B(e_2)\), as required.

For the \(\Leftarrow\) direction, assume \(B(e_1) \prec_\pi B(e_2)\). From the construction of \(\pi\) we have \((e_1, e_2) \in \text{tso}\). As \(G_k\) is PTSO-valid, we then know that \((e_1, e_2) \in \text{tso}\). Consequently, from the construction of \(\pi\) we have \(W(e_1) \prec_\pi W(e_2)\), as required.

For part (24), pick arbitrary \(e_1, e_2\) such that \(\text{tid}(e_1) = \text{tid}(e_2)\) and \(W(e_1) \prec_\pi F(e_2)\). We then know there exists \(j\) such that \(w_j = e_1\). Moreover, from the construction of \(\pi\) we know that for all \(\epsilon\) such that \(\text{tid}(\epsilon) = \text{tid}(e_1)\) we have \((e_1, e) \in \text{po} \iff W(e_1) \prec_\pi \text{genL}(e, G_k)\). As such, by definition we have \((e_1, e_2) \in \text{po}\). As \(G_k\) is PTSO-valid, we then know that \((e_1, e_2) \in \text{tso}\). Consequently, from the construction of \(\pi\) we have \(B(e_1) \prec_\pi F(e_2)\), as required.

The proofs of parts (25), (26) and (27) are analogous and omitted here.

For part (28), pick arbitrary \(e_1, e_2\) such that \(1\text{oc}(e_1) = 1\text{oc}(e_2)\). For the \(\Rightarrow\) direction, assume \(B(e_1) \prec_\pi B(e_2)\) or \(B(e_1) \prec_\pi \text{U}(e_2, -)\) or \(B(e_1, -) \prec_\pi \text{U}(e_2, -)\). From the construction of \(\pi\) we then know that \((e_1, e_2) \in \text{tso}\). As \(G_k\) is PTSO-valid, we then know that \((e_1, e_2) \in \text{nvo}\). Consequently, from the construction of \(\pi\) we have \(B(e_1) \prec_\pi \text{U}(e_2, -)\), as required.

For the \(\Leftarrow\) direction, assume \(\text{PB}(e_1) \prec_\pi \text{PB}(e_2)\). From the construction of \(\pi\) we then know that \((e_1, e_2) \in \text{nvo}\). Consequently, from the construction of \(\pi\) we have \(\text{PB}(e_1) \prec_\pi \text{PB}(e_2)\), as required.

Similarly, for part (29), pick arbitrary \(e_1, e_2 \in (PE \times PE) \setminus (W \cup U \cup W \cup U)\). For the \(\Rightarrow\) direction, assume \(B(e_1) \prec_\pi \text{PF}(e_2)\) or \(B(e_1, -) \prec_\pi \text{PF}(e_2)\) or \(B(e_1) \prec_\pi \text{PF}(e_1, -)\) or \(\text{PF}(e_1) \prec_\pi \text{PF}(e_2)\). From the construction of \(\pi\) we then know that \((e_1, e_2) \in \text{tso}\). Moreover, we know that \((e_1, e_2) \in [W \cup U \cup PF]\); \(\text{tso}; [PF] \cup [PF]; \text{tso}; [W \cup U \cup PF]\) and \(G_k\) is PTSO-valid, we then know that \((e_1, e_2) \in \text{tso}\). Consequently, from the construction of \(\pi\) we have \(B(e_1) \prec_\pi \text{PF}(e_2)\) or \(B(e_1, -) \prec_\pi \text{PF}(e_2)\) or \(B(e_1) \prec_\pi \text{PF}(e_1)\) or \(\text{PF}(e_1) \prec_\pi \text{PF}(e_2)\), as required.

For the \(\Leftarrow\) direction, assume \(\text{PB}(e_1) \prec_\pi \text{PB}(e_2)\). From the construction of \(\pi\) we then know that \((e_1, e_2) \in \text{nvo}\). As \((e_1, e_2) \in [W \cup U \cup PF]; \text{tso}; [PF] \cup [PF]; \text{tso}; [W \cup U \cup PF]\) and \(G_k\) is PTSO-valid, we then know that \((e_1, e_2) \in \text{tso}\). Consequently, from the construction of \(\pi\) we have \(\text{PB}(e_1) \prec_\pi \text{PB}(e_2)\), as required.

**Goal:** complete \((\pi'_k, \pi_k)\)

Follows immediately from the constructions of \(\pi'_k\) and \(\pi_k\).
As \( \text{wpf}(\pi'_k, \pi_k, \text{hist}(\Gamma_k)) \) and complete(\( \pi'_k, \pi_k \)) hold, we know get(\( \Gamma_k, \pi_k, \pi'_k \)) is defined. From the constructions of \( \pi'_k \) and \( \pi_k \), it is now straightforward to demonstrate that get(\( \Gamma_k, \pi_k, \pi'_k \)) = \( G_k \).

**Definition A.4.** Given a \( \Gamma = (G_n, (\pi'_n, \pi_n), \ldots, (G_1, (\pi'_1, \pi_1)) \) and an event path \( \pi \), let

\[
\text{wpf}(\Gamma, \pi) \overset{\text{def}}{=} \text{wpf}((\pi, \mathcal{H}) \land \bigwedge_{i=1}^{n} \text{get}(\Gamma_i, \pi_i, \pi'_i) = G_i \land \text{wpf}(\mathcal{H})
\]

where \( \Gamma_1 = \varepsilon; \Gamma_{i+1} = (G_i, (\pi'_i, \pi_i)) \ldots, (G_1, (\pi'_1, \pi_1)) \) for \( i \in \{1 \ldots n-1\} \); and \( \mathcal{H} = \text{hist}(\Gamma) \).

**Lemma A.3.** Let \( E = G_1; \ldots; G_n \) denote a PTSO-valid execution chain. Let \( S_1 = \varepsilon \) and \( S_{j+1} = G_j \ldots, G_1 \) for \( j \in \{1 \ldots n\} \).

1. For all \( (\pi', \pi) \in \text{traces}(G_i, S_i) \), and all \( \pi, \pi' \):
   \[
   \pi'_i \pi_i = \pi' \pi \Rightarrow \text{wpf}(\Gamma_i, \pi)
   \]
   where \( \Gamma_1 = \varepsilon \) and \( \Gamma_{j+1} = (G_j, (\pi'_j, \pi_j)) \ldots, (G_1, (\pi'_1, \pi_1)) \) for \( i \in \{1 \ldots j-1\} \).

2. For all \( (\pi'_n, \pi_n) \ldots, (\pi'_1, \pi_1) \in \text{traces}(G_n, S_n) \), \( \pi'_n = \varepsilon \).

**Proof.** Pick an arbitrary PTSO-valid execution chain \( E = G_1; \ldots; G_n \). Let \( S_1 = \varepsilon \) and \( S_{j+1} = G_j \ldots, G_1 \) for \( j \in \{1 \ldots n\} \).

**RTS. (1)** We proceed by induction on \( i \).

**Base case \( i = 1 \)**

Pick arbitrary \( (\pi'_1, \pi_1) \in \text{traces}(G_1, S_1) \) and \( \pi, \pi' \) such that \( \pi'_1 \pi_1 = \pi' \pi \). We are then required to show \( \text{wpf}(\Gamma_1, \pi) \), where \( \Gamma_1 = \varepsilon \). It thus suffices to show:

\[
\text{wpf}(\pi, \text{hist}(\Gamma_1)) \land \text{wpf}(\text{hist}(\Gamma_1))
\]

The second conjunct follows trivially from the fact that \( \text{hist}(\Gamma_1) = \varepsilon \) and the definition of \( \text{wpf}(\varepsilon) \).

As \( (\pi'_1, \pi_1) \in \text{traces}(G_1, S_1) \), from the definition of traces(\( , \)) we have get(\( \Gamma_1, \pi_1, \pi'_1 \)). Consequently, from the definition of get(\( \Gamma_1, \pi_1, \pi'_1 \)) we know that \( \text{wpf}(\pi'_1, \pi_1, \text{hist}(\Gamma_1)) \) holds implying the result in the first conjunct.

**Base case \( i = j+1 \)**

\[
\forall (\pi'_j, \pi_j) \ldots, (\pi'_1, \pi_1) \in \text{traces}(G_j, S_j). \forall \pi, \pi'. \pi'_j \pi_j = \pi' \pi \Rightarrow \text{wpf}(\Gamma'_j, \pi)
\]

where \( \Gamma'_j = \varepsilon \) and \( \Gamma'_{j+1} = (G_j, (\pi'_j, \pi_j)) \ldots, (G_1, (\pi'_1, \pi_1)) \) for \( i \in \{1 \ldots j-1\} \).

Pick arbitrary \( (\pi'_j, \pi_j) \ldots, (\pi'_1, \pi_1) \in \text{traces}(G_i, S_i) \) and \( \pi, \pi' \) such that \( \pi'_j \pi_j = \pi' \pi \). We are then required to show \( \text{wpf}(\Gamma_i, \pi) \). It thus suffices to show:

\[
\text{wpf}(\pi, \text{hist}(\Gamma_i)) \land \bigwedge_{k=1}^{j} \text{get}(\Gamma_k, \pi_k, \pi'_k) = G_k \land \text{wpf}(\text{hist}(\Gamma_i))
\]

where \( \Gamma_1 = \varepsilon \) and \( \Gamma_{j+1} = (G_j, (\pi'_j, \pi_j)) \ldots, (G_1, (\pi'_1, \pi_1)) \) for \( i \in \{1 \ldots j-1\} \).

The second conjunct follows from the definition of traces(\( , \)) and the fact that \( (\pi'_j, \pi_j) \ldots, (\pi'_1, \pi_1) \in \text{traces}(G_i, S_i) \). Similarly, as \( (\pi'_j, \pi_j) \ldots, (\pi'_1, \pi_1) \in \text{traces}(G_i, S_i) \), from the definition of traces(\( , \)) we know get(\( \Gamma_i, \pi_i, \pi'_i \)) = \( G_i \) and thus \( \text{wpf}(\pi'_1, \pi_i, \text{hist}(\Gamma_i)) \) holds implying the result in the first conjunct.
For the third conjunct, observe that $\text{hist}(\Gamma_j) = (\pi'_j, \pi_j).\text{hist}(\Gamma_j)$. As $(\pi'_j, \pi_j, \ldots, (\pi'_j, \pi_1) \in \text{traces}(G_i, S_i)$, from the definition of traces(,) we know that $\text{get}(\Gamma_j, (\pi_j, \pi'_j) = G_j$ and thus $\text{wfp}(\pi'_j, \pi_j, \text{hist}(\Gamma_j))$ and complete($\pi'_j, \pi_j$) hold. On the other hand, from (I.H.) we have $\text{wfp}(\text{hist}(\Gamma_j))$. As such, from the definition of wfp(,) we have $\text{wfp}(\Gamma_j)$, as required.

**RTS. (2)** We proceed by contradiction. Assume there exists $(\pi'_n, \pi_n, \ldots, (\pi'_1, \pi_1) \in \text{traces}(G_n, S_n)$ such that $\pi'_n \neq \epsilon$. Let $\Gamma = \epsilon$ and $\Gamma_{j+1} = (G_j, (\pi_j, \pi'_j)) \ldots (G_1, (\pi'_1, \pi_1))$ for $j \in \{1 \ldots i-1\}$. From the definition of traces(,) we then know that $\text{get}(\Gamma_n, \pi_n, \pi'_n) = G_n$, i.e. $\text{wfp}(\pi'_n, \pi_n, \text{hist}(\Gamma_n))$ and complete($\pi'_n, \pi_n$) hold. As $\pi'_n \neq \epsilon$, we then know there exists $\epsilon \in G_n, E$ such that $\text{PB}(\epsilon) \in \pi'_n$, i.e. (from the well-formedness of the path) $\text{PB}(\epsilon) \notin \pi_n$. As such, since $\text{get}(\Gamma_n, \pi_n, \pi'_n) = G_n$, from its definition we know that $\epsilon \notin G_n, E$. This however contradicts the assumption that $G_n$ is PTSO-valid.

**Lemma A.4.** Let $E = G_1; \ldots; G_n$ denote a PTSO-valid execution chain of program P with outcome $O$ and $G_i = (E_i^0, E_i^1, E_i, p_{0i}, r_{fi}, ts_{qi}, tsv_{qi})$ for $i \in \{1 \ldots n\}$. For each $G_i$, let $e_1^i, \ldots, e_m^i$ denote an enumeration of $E_i \setminus E_i^0$ that respects $p_{0i}$. Then there exists $P_1^i \ldots P_n^i, S_i, S_n^i$ such that:

- $P_i^{i-1}, S_i^{i-1} \leftarrow (E_i^0)^* \text{genL}(e_i^0, G_i) \leftarrow (E_i^0)^* P_i^i, S_i^i$, for $i \in \{1 \ldots n\}$ and $j \in \{1 \ldots m\}$
- $P_n = \text{skip} \| \ldots \| \text{skip}$ and $S_n^0 = O$

where $P_1 = P$; $P_0^i = \text{recover}$ for $i \in \{2 \ldots n\}$; and $S_0^i = S_0$ for $i \in \{1 \ldots n\}$.

**Lemma A.5.** Let $E = G_1; \ldots; G_n$ denote a PTSO-valid execution chain of program P with outcome $O$. Let $S_1 = \epsilon$ and $S_{j+1} = G_j \ldots G_1$ for $j \in \{1 \ldots n\}$. Then, for all $i \in \{1 \ldots n\}$, and all $H_i, \ldots, H_1 \in$ traces($G_i, S_i$):

1. If $i < n$ then $P_0^i, S_0, \Gamma_i, \epsilon \Rightarrow^* \text{recover}, S_0, \Gamma_{i+1}, \epsilon$
2. If $i = n$ then $P_0^i, S_0, \Gamma_n, \epsilon \Rightarrow^* \text{skip} \| \ldots \| \text{skip}$, $O, \Gamma_n, \pi_n$

where $P_0 = P$; $P_0^i = \text{recover}$; $\Gamma_1 = \epsilon$ and $\Gamma_{j+1} = (G_j, H_j), \ldots (G_1, H_1)$, for $j \in \{1 \ldots n-1\}$.

**Proof.** Pick an arbitrary program P and a PTSO-valid execution chain $E$ of P with outcome $O$ such that $E = G_1; \ldots; G_n$. Let $S_1 = \epsilon$ and $S_{j+1} = G_j; \ldots G_1$ for $j \in \{1 \ldots n\}$. Let $P_0^i = P$ and $P_0 = \text{recover}$ for $j \in \{2 \ldots n\}$. For all $i \in \{1 \ldots n\}$, pick arbitrary $(\pi'_i, \pi_i) \in$ traces($G_i, S_i$). Let $\Gamma_1 = \epsilon$ and $\Gamma_{j+1} = (G_j, (\pi'_j, \pi_j)), \ldots (G_1, (\pi'_1, \pi_1))$ for $j \in \{1 \ldots n\}$.

PART (1). Pick arbitrary $i < n$. From traces($G_i, S_i$) we know $\pi_i$ respects $G_i.p_0$. That is, $\pi_i$ is of the form: $s_m \text{genL}(e_m, G_i) \ldots s_1 \text{genL}(e_1, G_i).s_0$, where:

i) For each $j \in \{0 \ldots m\}$, $s_j = \lambda_i(k_j), \ldots \lambda_i(j_1)$ and each $\lambda_i(j_r)$ is either of the form $B(\epsilon)$ or of the form $\text{PB}(\epsilon)$, for $r \in \{1 \ldots k_j\}$; and

ii) $e_1 \ldots e_m$ denotes an enumeration of $G_i.E$ that respects $G_i.p_0$ (for all $e, e'$, if $(e, e') \in G_i.p_0$ then $\text{genL}(e, G_i) \prec_{\pi_i} \text{genL}(e', G_i)$).

Moreover, since $(\pi'_i, \pi_i) \in \text{traces}(G_i, S_i)$, from the definition of traces(,) we know that $\text{get}(S_i, \pi_i, \pi'_i) = G_i$. Additionally, from Lemma A.3 we know:

$$\forall \lambda, p, q. \pi'_i . \pi_i = p . \lambda . q \Rightarrow \text{fresh}(\lambda, p, q) \wedge \text{fresh}(\lambda, \pi_i)$$

(31)

From (G-ProP) we thus have $P_0^i, S_0, \Gamma_i, \epsilon \Rightarrow^* P_0^i, S_0, \Gamma_i, s_0$. There are now two cases to consider: 1) $m = 0$; or 2) $m > 0$.

In case (1), we have $\pi_i = s_0$ and thus (since each event in $s_0$ is either of the form $B(\epsilon)$ or of the form $\text{PB}(\epsilon)$) from Lemma A.3 we know $s_0 = \pi_i = \pi'_i = \epsilon$. As such, we have $P_0^i, S_0, \Gamma_i, \epsilon \Rightarrow^*$
P_i^0, S_0, \Gamma_i, \epsilon. Moreover, since \pi'_i = \epsilon then \text{comp}(\pi_i, \pi'_i) holds. As such from (G-CRASH) we have
P_i^0, S_0, \Gamma_i, \epsilon \Rightarrow^* \text{recover}, S_0, \Gamma_i+1, \epsilon, as required.

In case (2) from Lemma A.4 we know there exists P_i^1 \cdots P_i^m, S_i^1, S_i^m such that for j \in \{1 \cdots m\}:
\begin{equation}
P_i^{j-1}, S_i^{j-1} \xrightarrow{(E(\tau))^*} \text{genL}^{(e^j_i, G_i)}(E(\tau))^* P_i^j, S_i^j 
\end{equation}

where S_i^0 = S_0 for i \in \{1 \cdots n\}.

For each j \in \{1 \cdots m\}, from (32) we then know there exist P'_j, P''_j, S'_j, S''_j such that P_i^{j-1}, S_i^{j-1} \xrightarrow{(E(\tau))^*} P'_j, S'_j. Let p_0 = s_0 and p_j = s_j, \text{genL}(e_j, G_i)\cdots s_1, \text{genL}(e_1, G_i), s_0, for j \in \{1 \cdots m\}. As such, from (G-SILENTP), (G-STEP), (G-PROP), and (31) we then have:
\begin{align*}
&\Rightarrow^* P_i^{j-1}, S_i^{j-1}, \Gamma_i, p_{j-1} \\
&\Rightarrow P'_j, S'_j, \Gamma_i, p_{j-1} \\
&\Rightarrow P''_j, S''_j, \Gamma_i, \text{genL}(e_j, G_i). p_{j-1} \\
&\Rightarrow^* P'_j, S'_j, \Gamma_i, \text{genL}(e_j, G_i). p_{j-1} \\
&\Rightarrow P_i^j, S_i^j, \Gamma_i, p_j
\end{align*}

Consequently, we have
\begin{align*}
P_i^0, S_0, \Gamma_i, \epsilon \Rightarrow^* P_i^0, S_i^0, \Gamma_i, p_0 \Rightarrow^* P_i^1, S_i^1, \Gamma_i, p_1 \Rightarrow^* \cdots \Rightarrow^* P_i^m, S_i^m, \Gamma_i, p_m
\end{align*}

That is, we have
P_i^0, S_i^0, \Gamma_i, \epsilon \Rightarrow^* P_i^m, S_i^m, \Gamma_i, \pi_i

On the other hand from Lemma A.3 we know that \text{comp}(\pi_i, \pi'_i) holds. As such, since getG(S_i, \pi_i, \pi'_i)=G_i, from (G-CRASH) we have
\begin{align*}
P_i^m, S_i^m, \Gamma_i, \pi_i \Rightarrow^* \text{recover}, S_0, \Gamma_i+1, \epsilon
\end{align*}

That is, we have P_i^0, S_i^0, \Gamma_i, \epsilon \Rightarrow^* \text{recover}, S_0, \Gamma_i+1, \epsilon, as required.

**PART (2).** From traces(G_n, S_n) we know \pi_n respects G_n.po. That is, \pi_n is of form: s_m, \text{genL}(e_m, G_n) \\
\cdots s_1, \text{genL}(e_1, G_n), \text{where:}
\begin{enumerate}
\item For each j \in \{1 \cdots m\}, s_j = \lambda_{i(j,k_j)} \cdots \lambda_{i(j,1)} and each \lambda_{i(j,r)} is either of the form B(\langle) or of the form PB(\langle), for r \in \{1 \cdots k_j\};
\item \epsilon_1 \cdots \epsilon_m denotes an enumeration of G_n.E that respects G_i.po (for all e, e', if (e, e') \in G_n.po then \text{genL}(e, G_n) \prec_{\pi_n} \text{genL}(e', G_n)).
\end{enumerate}

Moreover, since (\pi'_n, \pi_n) \in \text{traces}(G_n, S_n), from the definition of traces(\ldots) we know that
getG(S_n, \pi_n, \pi'_n)=G_n. Additionally, from Lemma A.3 we know:
\begin{align*}
\pi_n = \epsilon \land \forall \lambda, p, q. \pi_n \pi_n = p \land q \Rightarrow \text{fresh}(\lambda, p, q) \land \text{fresh}(\lambda, \Gamma_n)
\end{align*}

From (G-PROP) we thus have P_n^0, S_0, \Gamma_n, \epsilon \Rightarrow^* P_n^0, S_0, \Gamma_n, s_0. There are now two cases to consider: 1) m = 0; or 2) m > 0.

In case (1), we have P_n^0 = \text{skip}|| \cdots ||\text{skip}, S_n = S_0 = O, and \pi_n = s_0 and thus (since each event in s_0 is either of the form B(\langle) or of the form PB(\langle)) from Lemma A.3 we know s_0 = \pi_n = \pi'_n = \epsilon.

As such, we trivially have P_n^0, S_0, \Gamma_n, \epsilon \Rightarrow^* \text{skip}|| \cdots ||\text{skip}, O, \Gamma_n, \epsilon, as required.

In case (2), in similar steps to that of the proof of part (1) we have:
\begin{align*}
P_n^0, S_n^0, \Gamma_n, \epsilon \Rightarrow^* P_n^m, S_n^m, \Gamma_n, \pi_n
\end{align*}

That is, we have P_n^0, S_n^0, \Gamma_n, \epsilon \Rightarrow^* \text{skip}|| \cdots ||\text{skip}, O, \Gamma_n, \pi_n, as required. 

\[\square\]
Corollary 1. Let $E = G_1; \cdots; G_n$ denote a PTSO-valid execution chain of program $P$ with outcome $O$.
Let $S_j = e$ and $S_{j+1} = G_j; \cdots; G_1$ for $j \in \{1, \cdots, n\}$. Then, there exists $H_n; \cdots; H_1 \in \text{traces}(G_n, S_n)$, with $H_n = (\pi_n, e)$ such that

$$P, S_0, e, e \Rightarrow^* \text{skip} \mid \cdots \mid \text{skip}, O, (G_{n-1}, H_{n-1}), \cdots, (G_1, H_1), \pi_n$$

Proof. Follows from Lemma A.2 and Lemma A.5.

Given an execution path $\pi$ and a graph history $\Gamma$, the set of configurations induced by $\Gamma$ and $\pi$,
written $\text{confs}(\Gamma, \pi)$, includes those configurations that satisfy the following condition:

$$\text{confs}(\Gamma, \pi) \triangleq \{(M, PB, B) \mid \text{wf}(M, PB, B, \text{hist}(\Gamma), \pi)\}$$

Lemma A.6. For all $P, P', S, S', \Gamma, \Gamma', \pi, \pi'$:

If

$$\text{wf}(\Gamma, \pi) \land \text{wf}(\Gamma', \pi') \land P, S, \Gamma, \pi \Rightarrow P', S', \Gamma', \pi'$$

then for all $(M, PB, B) \in \text{confs}(\Gamma, \pi)$, there exists $(M', PB', B') \in \text{confs}(\Gamma', \pi')$ such that

$$P, S, M, PB, B, \text{hist}(\Gamma), \pi \Rightarrow P', S', M', PB', B', \text{hist}(\Gamma'), \pi'$$

Proof. Pick arbitrary $P, P', S, S', \Gamma, \Gamma', \pi, \pi'$ such that $\text{wf}(\Gamma, \pi), \text{wf}(\Gamma', \pi')$, and $P, S, \Gamma, \pi \Rightarrow P', S', \Gamma', \pi'$. Pick arbitrary $(M, PB, B) \in \text{confs}(\Gamma, \pi)$. Let $\mathcal{H} = \text{hist}(\Gamma)$. From the definition of $\text{confs}(\ldots)$ we then know that $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds. We then proceed by induction on the structure of $\Rightarrow$.

Case (G-SILENTP)

From (G-SILENTP) we then know that $P, S \overset{E(\tau)}{\Rightarrow} P', S'$, and that $\Gamma' = \Gamma, \pi' = \pi$. As such, from (A-SILENTP) we have $P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P', S', M, PB, B, \mathcal{H}, \pi$. Moreover, as $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, the required result holds immediately.

Case (G-PROP)

From (G-PROP) we then know that there exists $e$ and $\lambda \in \{B(e), PB(e)\}$ such that $\pi' = \lambda.\pi$, fresh($\lambda, \pi$), fresh($\lambda, \Gamma$), $P' = P, S' = S$ and $\Gamma' = \Gamma$. From the definition of fresh($\ldots$) we then know that fresh($\lambda, \mathcal{H}$) holds. There are now three cases to consider. Either 1) $\lambda = B(e)$; or 2) $\lambda = PB(e)$ and $e \in W \cup U$; or 3) $\lambda = PB(e)$ and $e \in PF$.

In case (1), let tid($e$) = $\tau$, loc($e$) = $x$. Since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we know there exist $pb''$, PB such that $PB = (\text{NONE}, pb).PB''$. In what follows, we demonstrate that there exists $b$ such that $B(\tau) = b.e$. From (AM-BPROP) we then have $M, PB, B \overset{B(e)}{\Rightarrow} M, (\text{NONE}, pb[x \mapsto e.PB(x)]).PB'', B[\tau \mapsto b]$. As such, from (A-PROP-M) we have:

$P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M, (\text{NONE}, pb[x \mapsto e.PB(x)]).PB', B[\tau \mapsto b], \mathcal{H}, \lambda.\pi$

That is, there exists $M' = M, PB' = (\text{NONE}, pb[x \mapsto e.pb''(x)]).PB''$ and $B' = B[\tau \mapsto b]$ such that $P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M', PB', B', \mathcal{H}, \pi'$. Moreover, since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we also have $\text{wf}(M', PB', B', \mathcal{H}, \pi')$ and thus from the definition of $\text{confs}(\ldots)$ we have $(M', PB', B') \in \text{confs}(\Gamma, \pi')$, as required. We next demonstrate that there exists $b$ such that $B(\tau) = b.e$.

Since $\text{wf}(\Gamma', \pi')$ holds, we know that $W(e) \in \pi$. Moreover, as fresh($\lambda, \pi$), we know that $\lambda \notin \pi$.

As such, from the definition of $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ we know that $e \in B(\tau)$. Now let us suppose that $e$ is not at the head of $B(\tau)$, i.e. there exists $e' \neq e$ and $b$ such that $e' <_{B(\tau)} e$. Once again, from the
We know there exists \( W(e') \in \pi \), \( B(e') \notin \pi \) (and thus \( B(e') \notin \lambda.\pi \)) and that \( W(e') <_\pi W(e) \). Moreover, since \( ab \in \lambda.\pi \) and \( W(\Gamma, \lambda.\pi) \) holds, from the definition of \( \text{wf}(\ldots) \) and the definition of \( \text{wpf}(\ldots) \) we know that \( B(e') <_{\lambda.\pi} B(e) \). This however leads to a contradiction as \( B(e') \notin \lambda.\pi \). We can thus conclude that there exists \( b \) such that \( B(\tau) = b.e \).

In case (2), let \( PB = PB''(o, pb) \) and let \( 10c(e) = x \). In what follows, we demonstrate that there exists \( s \) such that \( pb(x) = s.e \). From (AM-PBProp) we then have \( M, B \xrightarrow{PB(e)} M[x \mapsto e], PB''(\text{None}, pb[x \mapsto s]), B, \mathcal{H}, \lambda.\pi \). As such, from (A-PropM) we have:

\[
P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M[x \mapsto e], PB''(o, pb[x \mapsto s]), B, \mathcal{H}, \lambda.\pi
\]

That is, there exists \( M' = M[x \mapsto e], PB' = PB''(o, pb[x \mapsto s]) \) and \( B' = B \) such that \( P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M', PB', B', \mathcal{H}, \pi' \). Moreover, since \( \text{wf}(M, PB, B, \mathcal{H}, \pi) \) holds, from its definition we also have \( \text{wf}(M', PB', B', \mathcal{H}, \pi') \) and thus from the definition of \( \text{confs}(\ldots) \) we have \( (M', PB', B') \in \text{confs}(\Gamma, \pi') \), as required. We next demonstrate that there exists \( s \) such that \( pb(x) = s.e \).

Since \( \text{wf}(\Gamma', \pi') \) holds, we know that there exists \( \lambda_e \in \pi \) such that \( \lambda_e = U(e, \_e) \) or \( \lambda_e = B(e) \). Moreover, as \( \text{fresh}(\lambda, \pi) \), we know that \( \lambda \notin \pi \). As such, from the definition of \( \text{wf}(M, PB, B, \mathcal{H}, \pi) \) we know there exists \( (o_e, pb_e) \in PB \) such that \( e \in pb_e(x) \). Now let us suppose that \( e \) is not the next event in \( PB \) to be propagated, i.e. either i) there exists \( (o_e', pb_e') \in PB \) such that \( (o_e', pb_e') <_{PB} (o_e, pb_e) \) and either \( o_e' = \text{Some}(e') \) or there exists \( y \) such that \( e' \in pb_{e'}(y) \); or ii) there exists \( y \) such that \( e' \in pb_{e'}(y) \). Once again, from the definition of \( \text{wf}(M, PB, B, \mathcal{H}, \pi) \) we know that there exists \( \lambda_{e'} \in \pi \) such that \( \lambda_{e'} = B(e') \), or \( \lambda_{e'} = U(e', \_e) \) or \( \lambda_{e'} = \text{PF}(e') \), that \( PB(e') \notin \pi \) (and thus \( PB(e') \notin \lambda.\pi \)) and that \( \lambda_{e'} <_{\pi} \lambda_e \). Moreover, since \( \lambda \in \lambda.\pi \) and \( \text{wf}(\Gamma, \lambda.\pi) \) holds, from the definition of \( \text{wf}(\ldots) \) and the definition of \( \text{wpf}(\ldots) \) we know that \( PB(e') <_{\lambda.\pi} PB(e) \). This however leads to a contradiction as \( PB(e') \notin \lambda.\pi \). We can thus conclude that there exists \( s \) such that \( pb(x) = s.e \).

In case (3), let \( PB = PB''(o, pb) \). In what follows, we demonstrate that \( (o, pb) = (\text{Some}(e), pb_0) \).

From (AM-PBPropF) we then have \( M, PB, B \xrightarrow{PB(e)} M, PB''(\text{None}, pb_0), B \). As such, from (A-PropM) we have:

\[
P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, MPB''(\text{None}, pb_0), B, \mathcal{H}, \lambda.\pi
\]

That is, there exists \( M' = M, PB' = PB''(\text{None}, pb_0) \) and \( B' = B \) such that \( P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M', PB', B', \mathcal{H}, \pi' \). Moreover, since \( \text{wf}(M, PB, B, \mathcal{H}, \pi) \) holds, from its definition we also have \( \text{wf}(M', PB', B', \mathcal{H}, \pi') \) and thus from the definition of \( \text{confs}(\ldots) \) we have \( (M', PB', B') \in \text{confs}(\Gamma, \pi') \), as required. We next demonstrate that \( (o, pb) = (\text{Some}(e), pb_0) \).

Since \( \text{wf}(\Gamma', \pi') \) holds, we know \( \text{PF}(e) \in \pi \). Moreover, as \( \text{fresh}(\lambda, \pi) \), we know that \( \lambda \notin \pi \). As such, from the definition of \( \text{wf}(M, PB, B, \mathcal{H}, \pi) \) we know there exists \( (o_e, pb_e) \in PB \) such that \( o_e = \text{Some}(e) \). Now let us suppose that \( e \) is not the next event in \( PB \) to be propagated, i.e. either i) there exists \( (o_{e'}, pb_{e'}) \in PB \) such that \( (o_{e'}, pb_{e'}) <_{PB} (o_e, pb_e) \) and either \( o_{e'} = \text{Some}(e') \) or there exists \( y \) such that \( e' \in pb_{e'}(y) \); or ii) there exists \( y \) such that \( e' \in pb_{e'}(y) \). Once again, from the definition of \( \text{wf}(M, PB, B, \mathcal{H}, \pi) \) we know that there exists \( \lambda_{e'} \in \pi \) such that \( \lambda_{e'} = B(e') \), or \( \lambda_{e'} = U(e', \_e) \) or \( \lambda_{e'} = \text{PF}(e') \), that \( PB(e') \notin \pi \) (and thus \( PB(e') \notin \lambda.\pi \)) and that \( \lambda_{e'} <_{\pi} \lambda_{e} \). Moreover, since \( \lambda \in \lambda.\pi \) and \( \text{wf}(\Gamma, \lambda.\pi) \) holds, from the definition of \( \text{wf}(\ldots) \) and the definition of \( \text{wpf}(\ldots) \) we know that \( PB(e') <_{\lambda.\pi} PB(e) \). This however leads to a contradiction as \( PB(e') \notin \lambda.\pi \). We can thus conclude that \( (o, pb) = (\text{Some}(e), pb_0) \).

**Case (G-Step)**

We know there exists \( e, r, u \) and \( \lambda \in \{R(r, e), W(e), U(u, e), F(e), PF(e), PS(e)\} \) such that \( \pi' = \lambda.\pi \), \( \text{fresh}(\lambda, \pi) \), \( \text{fresh}(\lambda, \Gamma) \), \( \Gamma' = \Gamma \) and \( P, S \rightarrow P', S' \). From the definition of \( \text{fresh}(\ldots) \) we then know...
that fresh($\lambda, \mathcal{H}$) holds. There are now six cases to consider. Either 1) $\lambda = R(e, w)$; or 2) $\lambda = W(e)$; or 3) $\lambda = U(e, w)$; or 4) $\lambda = F(e)$; or 5) $\lambda = PF(e)$; or 6) $\lambda = PS(e)$.

**Case 1:** $\lambda = R(e, e)$

Let $\text{tid}(r) = \tau$, $\text{loc}(r) = x$ and $B(\tau) = b$. In what follows we demonstrate that $\text{read}(M, PB, b, x) = e$.

From (AM-READ) we then have $M, PB, B \xrightarrow{R(e, e)} M, PB, B$. As such, from (A-STEP) we have:

$$P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M, PB, B, \mathcal{H}, \lambda, \pi$$

That is, there exists $M' = M, PB' = PB$ and $B' = B$ such that $P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M', PB', B', \mathcal{H}, \pi'$. Moreover, since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we also have $\text{wf}(M', PB', B', \mathcal{H}, \pi')$ and thus from the definition of $\text{confs}(\cdot)$ we have $(M', PB', B', \mathcal{H}, \pi') \in \text{confs}(\Gamma, \pi')$, as required. We next demonstrate that $\text{read}(M, PB, b, x) = e$.

From the definition of $\text{wf}(\Gamma, \lambda, \pi)$ we know that $\text{wfrd}(r, e, \pi, \pi_h)$, where $\pi_h = \pi_n, \ldots, \pi_1$, when $\Gamma = (\lambda, (\pi_n, -)), \ldots, (\pi_1, -))$. From the definition of $\text{wfrd}(r, e, \pi, \pi_h)$ there are now four cases to consider:

i) $\exists \pi_1, \pi_2. \pi = \pi_1, W(e), \pi_2 \land \text{tid}(e) = \text{tid}(r) \land B(e) \notin \pi_1$

\[ \land \{ W(e') \in \pi_1 | \text{loc}(e') = \text{loc}(r) \land \text{tid}(e') = \text{tid}(r) \} = \emptyset \]

ii) $\exists \pi_1, \pi_2, \lambda_e. \pi = \pi_1 \lambda_e, \pi_2 \land (\lambda_e = B(e) \lor \lambda_e = U(e, -))$

\[ \land \{ B(e'), U(e', -) \in \pi_1 | \text{loc}(e') = \text{loc}(r) \} = \emptyset \]

\[ \land \{ e' | W(e') \in \pi \land B(e') \notin \pi \land \text{loc}(e') = \text{loc}(r) \land \text{tid}(e') = \text{tid}(r) \} = \emptyset \]

iii) $\exists \pi_1, \pi_2. \pi_h = \pi_1, PB(e), \pi_2$

\[ \land \{ B(e'), U(e', -) \in \pi, | \text{loc}(e') = \text{loc}(r) \land \text{tid}(e') = \text{tid}(r) \} = \emptyset \]

\[ \land \{ W(e'') \in \pi, | \text{loc}(e'') = \text{loc}(r) \land \text{tid}(e'') = \text{tid}(r) \} = \emptyset \]

iv) $e = \text{init}_x \land \{ B(e'), U(e', -) \in \pi, | \text{loc}(e') = \text{loc}(r) \land \text{tid}(e') = \text{tid}(r) \} = \emptyset$

In case (i), since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we know there exists $b'$ such that $b = e, b'$. As such, by definition we have $\text{read}(M, PB, b, x) = e$.

In case (ii), since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we know that for all $e' \in b$, $\text{loc}(e') \neq x$; and that there exists $PB_1, PB_2, (a, pb)$ such that $PB = PB_1, (a, pb), PB_2, PB(x) = e.s$ and for all $(o', pb') \in PB_1, pb'(x) = e$. As such, by definition we have $\text{read}(M, PB, b, x) = e$.

In case (iii), since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we know that for all $e' \in b$, $\text{loc}(e') \neq x$; that for all $(o, pb) \in PB, PB(x) = e$; and that $M(x) = \text{init}_x$. As such, by definition we have $\text{read}(M, PB, b, x) = e$.

In case (iv), since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we know that for all $e' \in b$, $\text{loc}(e') \neq x$; that for all $(o, pb) \in PB, PB(x) = e$; and that $M(x) = \text{init}_x$. As such, by definition we have $\text{read}(M, PB, b, x) = e$.

**Case 2:** $\lambda = W(e)$

Let $\text{tid}(e) = \tau$. From (AM-WRITE) we then have $M, PB, B \xrightarrow{W(e)} M, PB, B[\tau \mapsto e.B(\tau)]$. As such, from (A-STEP) we have:

$$P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M, PB, B[\tau \mapsto e.B(\tau)], \mathcal{H}, \lambda, \pi$$

That is, there exists $M' = M$, $PB' = PB$ and $B' = B[\tau \mapsto e.B(\tau)]$ such that $P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M', PB', B', \mathcal{H}, \pi'$. Moreover, since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we also have $\text{wf}(M', PB', B', \mathcal{H}, \pi')$ and thus from the definition of $\text{confs}(\_ , \_ )$ we have $(M', PB', B') \in \text{confs}(\Gamma, \pi')$, as required.

Case 3: $\lambda = U\langle u, e \rangle$

Let $\text{tid}(u) = \tau$ and $\text{loc}(u) = x$. In what follows we demonstrate that $B(\tau) = e$. Since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we know there exist $pb''$, $PB$ such that $PB = (\text{None}, pb).PB''$. Moreover, in an analogous way to that in case (2) we can demonstrate that $\text{read}(M, PB, b, x) = e$. From (AM-RMW) we then have $M, PB, B \xrightarrow{U\langle u, e \rangle} M, (\text{None}, pb[x \mapsto u.pb(x)]).PB' , B$. As such, from (A-STEP) we have:

$$P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M, (\text{None}, pb[x \mapsto u.pb(x)]).PB'', B, \mathcal{H}, \lambda, \pi$$

That is, there exists $M' = M$, $PB' = (\text{None}, pb[x \mapsto u.pb(x)]).PB''$ and $B' = B$ such that $P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M', PB', B', \mathcal{H}, \pi'$. Moreover, since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we also have $\text{wf}(M', PB', B', \mathcal{H}, \pi')$ and thus from the definition of $\text{confs}(\_ , \_ )$ we have $(M', PB', B') \in \text{confs}(\Gamma, \pi')$, as required. We next demonstrate that $B(\tau) = e$.

Let us suppose that there exists $e'$ such that $e' \in b(\tau)$. We then know that $\text{tid}(e') = \tau$. From the definition of $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ we then know that $W(e') \in \pi$, $B(e') \notin \pi$ and thus $B(e') \notin \lambda, \pi$. That is, we have $W(e') <_{\lambda, \pi} \lambda$. Moreover, since $\text{alb} \in \lambda, \pi$ and $W(\Gamma, \lambda, \pi)$ holds, from the definition of $\text{wf}(\_ , \_ )$ and the definition of $\text{wfp}(\_ , \_ )$ we know that $B(e') <_{\lambda, \pi} F(e)$. This however leads to a contradiction as $B(e') \notin \lambda, \pi$. We can thus conclude that $B(\tau) = e$.

Case 4: $\lambda = F(e)$

Let $\text{tid}(e) = \tau$. In an analogous way to that in case (3) we can demonstrate that $B(\tau) = e$. From (AM-FENCE) we then have $M, PB, B \xrightarrow{F(e)} M, PB, B$. As such, from (A-STEP) we have:

$$P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M, PB, B, \mathcal{H}, \lambda, \pi$$

That is, there exists $M' = M$, $PB' = PB$ and $B' = B$ such that $P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M', PB', B', \mathcal{H}, \pi'$. Moreover, since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we also have $\text{wf}(M', PB', B', \mathcal{H}, \pi')$ and thus from the definition of $\text{confs}(\_ , \_ )$ we have $(M', PB', B') \in \text{confs}(\Gamma, \pi')$, as required.

Case 5: $\lambda = \text{PF}(e)$

Let $\text{tid}(e) = \tau$. In an analogous way to that in case (3) we can demonstrate that $B(\tau) = e$. On the other hand, from $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ and the definition of $\text{puff}(\_ , \_ )$ in particular, we know that there exists $pb$ and $PB''$ such that $PB = (\text{None}, pb).PB''$. As such, from (AM-PFENCE) we have:

$$M, PB, B \xrightarrow{\text{PF}(e)} M, (\text{None}, pb_0).(\text{Some}(e), pb).PB' , B$. As such, from (A-STEP) we have:

$$P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M, (\text{None}, pb_0).(\text{Some}(e), pb).PB', B, \mathcal{H}, \lambda, \pi$$

That is, there exists $M' = M$, $PB' = (\text{None}, pb_0).(\text{Some}(e), pb).PB''$ and $B' = B$ such that $P, S, M, PB, B, \mathcal{H}, \pi \Rightarrow P, S, M', PB', B', \mathcal{H}, \pi'$. Moreover, since $\text{wf}(M, PB, B, \mathcal{H}, \pi)$ holds, from its definition we also have $\text{wf}(M', PB', B', \mathcal{H}, \pi')$ and thus from the definition of $\text{confs}(\_ , \_ )$ we have $(M', PB', B') \in \text{confs}(\Gamma, \pi')$, as required.

Case 6: $\lambda = \text{PS}(e)$

Let $\text{tid}(e) = \tau$. In an analogous way to that in case (3) we can demonstrate that $B(\tau) = \epsilon$. In what follows we demonstrate that $PB = PB_0$. As such, from (AM-PSYNC) we have:

$$M, PB, B \xrightarrow{\text{PS}(e)} M, PB, B$$
As such, from (A-Step) we have:

\[ P, S, M, PB, B, H, \pi \Rightarrow P, S, M, PB, B, H, \lambda. \pi \]

That is, there exists a \( M' = M, PB' = PB \) and \( B' = B \) such that \( P, S, M, PB, B, H, \pi \Rightarrow P, S, M', PB', B', H, \pi' \). Moreover, since \( \text{wf}(M, PB, B, H, \pi) \) holds, from its definition we also have \( \text{wf}(M', PB', B', H, \pi') \) and thus from the definition of \( \text{confs}(\ldots) \) we have \( (M', PB', B') \in \text{confs}(\Gamma, \pi') \), as required. We next demonstrate that \( PB = PB_0 \).

Let us suppose \( PB \neq PB_0 \), i.e. there exist \( e' \) and \( (o_e, pb_e) \in PB \) such that either i) \( o_e = \text{Some}(e') \); or ii) there exists \( y \) such that \( e' \in pb_e(y) \). Once again, from the definition of \( \text{wf}(M, PB, B, H, \pi) \) we know that there exists \( \lambda_{e'} \in \pi \) such that \( \lambda_{e'} = B(e') \), or \( \lambda_{e'} = U(e', -) \) and \( \lambda_{e'} = \text{PF}(e') \), that \( PB(e') \notin \pi \) (and thus \( PB(e') \notin \lambda. \pi \)) and that \( \lambda_{e'} <_{\lambda. \pi} \lambda \). Moreover, since \( \lambda \in \lambda. \pi \) and \( \text{wf}(\Gamma, \lambda. \pi) \) holds, from the definition of \( \text{confs}(\ldots) \) and the definition of \( \text{wf}(\ldots) \) we know that \( PB(e') <_{\lambda. \pi} \text{PS}(e) \). This however leads to a contradiction as \( PB(e') \notin \lambda. \pi \). We can thus conclude that \( PB = PB_0 \).

**Case (G-Crash)**

Let \( \Gamma = (G_0, -) \ldots (G_i, -) \). From (G-Crash) we know there exists \( \pi'' \) and \( G \) such that \( \pi' = \text{recover}, S' = S_0, \Gamma = (G, (\pi', \pi)), \pi' = e, \text{comp}(\pi, \pi'') \) and \( \text{getG}(G_0, \ldots, G_i, \pi, \pi'') = G \), since \( \text{wf}(M, PB, B, H, \pi) \) holds, from its definition we know that for all events \( e \) and all \( (o, pb) \in PB: \)

i) \( e \in B(\text{tid}(e)) \land W(e) \in \pi \land B(e) \notin \pi; \) and that

ii) \( e \in \text{pb}(\text{loc}(e)) \lor o = \text{Some}(e) \land PB(e) \notin \pi \lor B(e) \in \pi \lor U(e, -) \in \pi \lor \text{PF}(e) \in \pi. \)

As such, from the definition of \( \text{confs}(\ldots) \) we know for all events \( e \) and all \( (o, pb) \in PB: \)

i) \( e \in B(\text{tid}(e)) \land B(e) \in \pi''; \)

ii) \( e \in \text{pb}(\text{loc}(e)) \lor o = \text{Some}(e) \land PB(e) \in \pi''. \)

As such, from the definition of \( \Rightarrow_p \), we have \( M, PB, B \Rightarrow_p \cdots \Rightarrow_p PB_0, B_0 \). Consequently, from (A-Step) we have:

\[ P, S, M, PB, B, H, \pi \Rightarrow P', S', M, PB_0, B_0, (\pi'', \pi) : H, \pi' \]

That is, there exists \( M' = M, PB' = PB_0, B' = B_0 \) and \( H' = (\pi'', \pi) : H = \text{hist}(\Gamma') \) such that:

\[ P, S, M, PB, B, H, \pi \Rightarrow P, S, M', PB', B', H', \pi'. \]

Since \( \text{comp}(\pi, \pi'') \) holds, by definition we have complete(\( \pi'', \pi \)). Moreover, since \( \text{wf}(M, PB, B, H, \pi) \) holds and \( \text{wf}(\Gamma', \pi') \) holds, from their definitions we also have \( \text{wf}(M', PB', B', H', \pi') \) and thus from the definition of \( \text{confs}(\ldots) \) we have \( (M', PB', B') \in \text{confs}(\Gamma, \pi') \), as required.

**Theorem 5** (Completeness). Given a program \( P \), for all PTSO-valid execution chains \( E \) of \( P \) with outcome \( O \), there exists \( M, H \) and \( \pi \) such that

\[ P, S_0, M_0, PB_0, B_0, e, e \Rightarrow^* \text{skip} \cdots \Rightarrow^* \text{skip} \| O, M, PB_0, B_0, H, \pi \]

**Proof.** Follows from Corollary 1, Lemma A.3 and Lemma A.6.

**A.4 Equivalence of PTSO Operational and Intermediate Semantics**

Let

\[ R_I \triangleq \left\{ (\tau : l, \lambda) \mid (\exists e. \text{getE}(\lambda) = e \land \text{tid}(e) = \tau \land \text{ab}(e) = l) \lor (\lambda = E(\tau) \land l = e) \right\} \]

**Lemma A.7.** For all \( P, S, P', S' \):

- for all \( \tau, l \), if \( P, S \xrightarrow{r_1} P', S' \), then there exists \( \lambda \) such that: \((\tau, l, \lambda) \in R_I \) and \( P, S \xrightarrow{\lambda} P', S' \);

- for all \( \lambda \), if \( P, S \xrightarrow{\lambda} P', S' \), then there exists \( \tau, l \) such that: \((\tau, l, \lambda) \in R_I \) and \( P, S \xrightarrow{r_1} P', S' \).
Proof. By straightforward induction on the structures of $\frac{\tau}{\rightarrow}$ and $\frac{\lambda}{\Rightarrow}$.

Let

$$R_m \triangleq \left\{ (\langle M, PB, B \rangle), (M, PB, B) \in Mem \times PBuff \times BMap \land (M, PB, B) \in AMem \times APBuff \times ABMap \land \forall x, v. M(x) = v \iff val_w(M(x)) = v \land sim_pb(PB, PB) \land sim_b(B, B) \right\}$$

$$\sim_pb(PB, PB) \triangleq PB = PB = \epsilon \lor \exists ph, pb, PB', PB'. PB = PB', \sim_{pb}(\sim_pb(pb(x), pb(x))) \land \forall x.\ sim_w(pb(x), ph(x)) \land \forall x, v.\ sim_w(pb(x), ph(x)) \simPB x, B', e. B = B'.(x, v) \land B = B'.e \land val_w(e) = e \implies 1$$

$$\sim_b(B, B) \triangleq (B = B = \epsilon) \lor (\exists x, v.\ B', e. B = B'.(x, v) \land B = B'.e \land val_w(e) = e \land \text{loc}(e) = x)$$

Lemma A.8. For all $M, PB, B, M', PB', B'$:

- $((M_0, PB_0, B_0), (M_0, PB_0, B_0)) \in R_m$;
- for all $M', PB', B', \tau, l$, if $((M, PB, B), (M, PB, B)) \in R_m$ and $(M, PB, B) \frac{\tau}{\rightarrow} (M', PB', B')$, then there exist $M', PB', B', \lambda$ such that $((\tau, l), \lambda) \in R_l$, $((M', PB', B'), (M, PB, B)) \in R_m$ and $(M, PB, B) \frac{\lambda}{\Rightarrow} (M', PB', B')$;
- for all $M', PB', B', \lambda$, if $((M, PB, B), (M, PB, B)) \in R_m$ and $(M, PB, B) \frac{\lambda}{\Rightarrow} (M', PB', B')$, then there exist $M', PB', B', \tau, l$ such that $((\tau, l), \lambda) \in R_l$, $((M', PB', B'), (M, PB, B)) \in R_m$ and $(M, PB, B) \frac{\tau}{\rightarrow} (M', PB', B')$.

Proof. The proof of the first part follows immediately from the definitions of $M_0, PB_0, B_0, M_0, PB_0, B_0$. The proofs of the last two parts follow from straightforward induction on the structures of $\frac{\tau}{\rightarrow}$ and $\frac{\lambda}{\Rightarrow}$.

Let

$$R \triangleq \left\{ (\langle P, S, M, PB, B \rangle), (P, S, M, PB, B, H, \pi) \in Prog \land S \in SMAP \land H \in HIST \land \pi \in PATH \land ((M, PB, B), (M, PB, B)) \in R_m \right\}$$

Lemma A.9. For all $P, M, PB, B, M', PB', B', \mathcal{H}, \pi$:

- $((P, S_0, M_0, PB_0, B_0), (P, S_0, M_0, PB_0, B_0, e, \epsilon)) \in R$;
- for all $P', S', M', PB', B'$, if $((P, S, M, PB, B), (P, S, M, PB, B, H, \pi)) \in R$ and $(P, S, M, PB, B) \Rightarrow (P', S', M', PB', B')$, then there exist $M', PB', B', \mathcal{H}', \pi'$ such that $((P', S', M', PB', B'), (P', S', M', PB', B', H', \pi')) \in R$ and $(P, S, M, PB, B, H, \pi) \Rightarrow (P, S', M', PB', B', H', \pi')$;
- for all $P', S', M', PB', B', \mathcal{H}', \pi'$, if $((P, S, M, PB, B), (P, S, M, PB, B, H, \pi)) \in R$ and $(P, S, M, PB, B, H, \pi) \Rightarrow (P', S', M', PB', B', H', \pi')$, then there exist $M', PB', B'$ such that $((P', S', M', PB', B'), (P', S', M', PB', B', H', \pi')) \in R$ and $(P, S, M, PB, B) \Rightarrow (P', S', M', PB', B', H', \pi')$.

Proof. The proof of the first part follows immediately from the definitions of $R$ and Lemma A.8. The proofs of the last two parts follow from straightforward induction on the structures of $\frac{\tau}{\rightarrow}$, $\frac{\lambda}{\Rightarrow}$, Lemma A.7 and Lemma A.8.

Theorem 6 (Intermediate and operational semantics equivalence). For all $P, S$:
• for all $M$, if $P, S_0, M_0, PB_0, B_0 \Rightarrow^* \text{skip}||\cdots||\text{skip}, S, M, PB_0, B_0$, then there exist $M, \mathcal{H}, \pi$ such that $P, S_0, M_0, PB_0, B_0, \epsilon, \epsilon \Rightarrow^* \text{skip}||\cdots||\text{skip}, S, M, PB_0, B_0, \mathcal{H}, \pi$ and $((M, PB_0, B_0), (M, PB_0, B_0)) \in R_m$.

• for all $M, \mathcal{H}, \pi$, if $P, S_0, M_0, PB_0, B_0, \epsilon, \epsilon \Rightarrow^* \text{skip}||\cdots||\text{skip}, S, M, PB_0, B_0, \mathcal{H}, \pi$, then there exists $M$ such that $P, S_0, M_0, PB_0, B_0 \Rightarrow^* \text{skip}||\cdots||\text{skip}, S, M, PB_0, B_0$ and $((M, PB_0, B_0), (M, PB_0, B_0)) \in R_m$.

**Proof.** Follows from Lemma A.9 and straightforward induction on the length of $\Rightarrow^*$.  

\[\square\]
**B  SIMPLE QUEUE LIBRARY**

For an arbitrary program P and a PTSO-valid execution \( E = G_1; \cdots ; G_n \) of a program P with \( G_i = (E^0, E^P, E, \text{po, rf, iso, nvo}) \), observe that when P comprises k threads, the trace of each execution \( \text{era} \) (via \( \text{start}() \) or \( \text{recover}() \)) comprises two stages: i) the trace of the \textit{setup} stage by the master thread \( \tau_0 \) performing initialisation or recovery, prior to the call to \( \text{run}(P); \) followed (in \( \text{po} \) order) by ii) the trace of the constituent program threads \( \tau_1 \cdots \tau_k \), provided that the execution did not crash during the setup stage.

Note that as the execution is PTSO-valid, thanks to the placement of the persistent fence operations (pfence), for each thread \( \tau_j \), we know that the set of persistent events in execution era \( i \), namely \( E^P_i \), contains roughly a \textit{prefix} (in \( \text{po} \) order) of thread \( \tau_j \)’s trace. More concretely, for each constituent thread \( \tau_j \in \{ \tau_1 \cdots \tau_k \} = \text{dom}(P) \), there exist \( P^j_1 \cdots P^j_n \) such that:

1) \( P[\tau_j] = o^0_j; \cdots ; o^1_j; o^1_j+1; \cdots ; o^{p_j-1+j}; \cdots ; o^{p_j-1+n-1+j}; \cdots ; o^{p_j-n} \), comprising \text{enq} \ and \ \text{deq} \ operations; and

2) at the beginning of each execution era \( i \in \{ 1 \cdots n \} \), the program executed by thread \( \tau_j \) (calculated in \( P^j \) and subsequently executed by calling \( \text{run}(P^j) \)) is that of \( \text{sub}(P[\tau_j], P^{j-1+1}) \),

where \( P^0_j = -1 \), for all \( j \); and

3) in each execution era \( i \in \{ 1 \cdots n \} \), the trace \( H(i, j) \) of each constituent thread \( \tau_j \in \text{dom}(P) \) is of the following form:

\[
H(i, j) = \begin{cases} 
H(o^0_j; \cdots ; o^1_j; o^1_j+1; \cdots ; o^{p_j-1+j}; \cdots ; o^{p_j-1+n-1+j}; \cdots ; o^{p_j-n}) 
& \text{po} \\
H(o^0_j; \cdots ; o^1_j; o^1_j+1; \cdots ; o^{p_j-1+j}; \cdots ; o^{p_j-1+n-1+j}; \cdots ; o^{p_j-n}) 
& \text{po} \\
\cdots 
& \text{po} \\
H(o^0_j; \cdots ; o^1_j; o^1_j+1; \cdots ; o^{p_j-1+j}; \cdots ; o^{p_j-1+n-1+j}; \cdots ; o^{p_j-n}) 
& \text{po} \\
\end{cases}
\]

for some \( m^j_1, n^j_1, \cdots , n^j_1, \cdots , m^j_n, n^j_n \), where:

- The first line denotes the execution of the \((P^{j-1+1})^{\text{st}}\) to \( (P^j)^{\text{th}} \) library calls of thread \( \tau_j \), with \( H(o, \tau, p, n) \) defined shortly. Moreover, before crashing and proceeding to the next era, all volatile events (those in \( PE \)) in \( H(o^0_j; \cdots ; o^1_j; o^1_j+1; \cdots ; o^{p_j-1+j}; \cdots ; o^{p_j-1+n-1+j}; \cdots ; o^{p_j-n}) \) have persisted, and a \textit{prefix} (in \( \text{po} \) order) of the volatile events in \( H(o^0_j; \cdots ; o^1_j; o^1_j+1; \cdots ; o^{p_j-1+j}; \cdots ; o^{p_j-1+n-1+j}; \cdots ; o^{p_j-n}) \) have persisted. Note that this prefix may be equal to \( H(o^0_j; \cdots ; o^1_j; o^1_j+1; \cdots ; o^{p_j-1+j}; \cdots ; o^{p_j-1+n-1+j}; \cdots ; o^{p_j-n}) \), in which case all its events have persisted.

- The second line denotes the execution of the subsequent library calls of thread \( \tau_j \) where \( m^j_n \leq P^j_n \), with \textit{none} of their volatile events having persisted.

- The last line denotes the execution of the \((m^j_n)^{\text{th}}\) call of thread \( \tau_j (m^j_n \leq P^j_n) \), during which the program crashed and thus the execution of era \( i \) ended. The \( H'(o, \tau, p, n) \) denotes a (potentially full) prefix of \( H(o, \tau, p, n) \).

The trace \( H(o, \tau, p, n) \) of each library call is defined as follows:

\[
H(\text{deq}(), \tau, p, n) \triangleq \text{inv}=I(t_p, \text{deq}(),) \xrightarrow{\text{po}} H(p(c, p)) \xrightarrow{\text{po}} H(t\text{id}, \tau) \\
\xrightarrow{\text{po}} R(q, \text{lock}, 1) \xrightarrow{\text{po}} q=U(q, \text{lock}, 0, 1) \\
\xrightarrow{\text{po}} r=R(q, \text{head}, h) \xrightarrow{\text{po}} r=R(q, \text{data}[h], n) \\
\xrightarrow{\text{po}} \text{lin}_1=W(\text{map}[\tau], (p, n)) \xrightarrow{\text{po}} S_1 \xrightarrow{\text{po}} S_\text{F} \xrightarrow{\text{po}} S_k \\
\xrightarrow{\text{po}} qu=W(q, \text{lock}, 0) \xrightarrow{\text{po}} \text{ack}=A(t_p, \text{deq})
\]

with
\[
S_1 = \begin{cases} 
\emptyset & \text{if } n = \text{null} \\
R(n, t, \tau') \mapsto R(n, p, p') \mapsto R(\text{map}[\tau'], (t, t)) \mapsto S_3 & \text{otherwise}
\end{cases}
\]

\[
S_3 = \begin{cases} 
\emptyset & \text{if } t > p' \\
U(\text{map}[\tau'], (t', t), (p'+1, \perp)) & \text{if } t \leq p' \text{ and } (t', t') = (t, t) \\
R(\text{map}[\tau'], (t', t')) & \text{otherwise}
\end{cases}
\]

\[
S_2 = \begin{cases} 
\emptyset & \text{if } n = \text{null} \\
\text{lin}_2 = W(q, \text{head}, h+1) \mapsto \text{PF} & \text{otherwise}
\end{cases}
\]

for some $\tau', p', t, t, t', t''$; and

\[
H(\text{enq}(u), \tau, p, n) \triangleq \text{inv} = \text{I}(r, \text{enq}, n) \mapsto R(p, p) \mapsto R(\text{tid}, \tau) \\
\mapsto W(n, v) \mapsto W(n, \text{tid}, \tau) \mapsto W(n, p, p) \\
\mapsto W(\text{map}[\tau'], (p, n)) \mapsto \text{PF} \\
\mapsto R(q, \text{lock}, 1) \mapsto U(q, \text{lock}, 0, 1) \mapsto R(q, \text{head}, h) \\
\mapsto R(q, \text{data}[h], v_0) \mapsto \cdots \mapsto R(q, \text{data}[h+s-1], v_{s-1})
\]

\[
\overset{s \text{ times}}{\mapsto} R(q, \text{data}[h+s], \text{null}) \mapsto \text{lin} = W(q, \text{data}[h+s], n) \\
\mapsto \text{PF} \mapsto W(q, \text{lock}, 0) \mapsto \text{ack} = \text{A}(r, \text{enq}, n)
\]

for some $s \geq 0$, and for all $v \in \{v_0 \cdots v_{s-1}\}$, $v \neq \text{null}$. In the above traces, for brevity we have omitted the thread identifiers ($r_i$) and event identifiers and represent each event with its label only.

We use the $H(\text{enq}(-), \tau, p, n).\text{inv}$.

It is straightforward to demonstrate that $h_{b_i} = (p_{o_i} \cup r_{f_i})^+$ restricted to the lock events in

\[
\bigcup_{r_j \in \text{dom}(P)}^m \{H(o_{j}^{p_{j}^{+}+1} \tau_j, P_{j}^{p_{j}^{+}+1}, n_j^{p_{j}^{+}+1}) \cdots H(o_j^{p_{j}^{+}+1}, \tau_j, P_j^{p_{j}^{+}}, n_j^{p_{j}^{+}})\}
\]

of

\[
\bigcup_{r_j \in \text{dom}(P)}^m \{H(o_{j}^{p_{j}^{+}+1} \tau_j, P_{j}^{p_{j}^{+}+1}, n_j^{p_{j}^{+}+1}) \cdots H(o_j^{p_{j}^{+}+1}, \tau_j, P_j^{p_{j}^{+}}, n_j^{p_{j}^{+}})\}
\]

In particular, we know there exists an enumeration $C_i = H(c_i^{1}, t_1, t_{1}, n_1^{1}) \cdots H(c_i^{t_i}, t_i, n_{i}^{t_i})$ of

\[
\bigcup_{r_j \in \text{dom}(P)}^m \{H(o_{j}^{p_{j}^{+}+1} \tau_j, P_{j}^{p_{j}^{+}+1}, n_j^{p_{j}^{+}+1}) \cdots H(o_j^{p_{j}^{+}+1}, \tau_j, P_j^{p_{j}^{+}}, n_j^{p_{j}^{+}})\}
\]

such that:

\[
\begin{cases} 
\{H(c_i^{k}, \tau_i, p_i^{k}, n_i^{k}), q_1, H(c_i^{k}, \tau_i, p_i^{k}, n_i^{k}), q_2, \} \mapsto & \land k \in \{1 \cdots t_i\} \\
\{H(c_i^{1}, \tau_i, P_i^{1}, n_i^{1}), \tau_l, H(c_i^{1}, \tau_i, P_i^{1}, n_i^{1}), q_l\} \mapsto & \land l \in \{1 \cdots t_i-1\}
\end{cases}
\]

\[
(h_{b_i})_{\text{loc}}\text{imm}
\]

Let $1p(H(o, \tau, p, n)) \triangleq \begin{cases} 
H(o, \tau, p, n).\text{lin} & \text{if } o = \text{enq}(v) \\
H(o, \tau, p, n).\text{lin}_1 & \text{if } o = \text{deq}() \text{ and } H(o, \tau, p, n).S_2 = \emptyset \\
H(o, \tau, p, n).\text{lin}_2 & \text{if } o = \text{deq}() \text{ and } H(o, \tau, p, n).S_2 \neq \emptyset
\end{cases}
\]

For each $r_j \in \text{dom}(P)$ let:

\[
E^p_{(i,j)} = E^p_i \cap \{e \mid \text{tid}(e) = r_j\} \quad \quad E^p_{(i,j)} = E^p_{(i,j)} \cup S_{(i,j)}
\]
Lemma B.1. Given a PTSO-valid execution $E = G_1; \cdots ; G_n$, let for all $i \in \{1 \cdots n\}$, $H_i$ be defined as above with $C_i = H(c_i^1, \tau_i^1, p_i^1, n_i^1)$. \ldots . H(c_i^l, \tau_i^l, p_i^l, n_i^l)$. For all $i \in \{1 \cdots n\}$, and $a,b$, let $O^k_{ab} = H(c_a^b, \tau_a^b, p_a^b, n_a^b).inv.H(c_a^b, \tau_a^b, p_a^b, n_a^b).ack. \ldots . H(c_b^b, \tau_b^b, p_b^b, n_b^b).inv.H(c_b^b, \tau_b^b, p_b^b, n_b^b).ack.$

For all $G_i = (E_i^0, E_i^p, E_i, p_{oi}, r_{fi}, t_{soi}, nvo_i), H_i$, for all $Q_i^k$ and for all $l \in \{0 \cdots l_i\}, k = t_i - l$, $E_i^k = E_i^p \setminus \bigcup_{x=k+1}^{t_i} H(c_i^x, \tau_i ^x, p_i ^x, n_i ^x).E, \text{ and } Q_i^k$:

$$\text{getQ}(Q_i^k, O_i^{k+1}) = Q_i^k \land \text{isQ}(q_i, Q_i^k, nvo_i, E_0^i, E_i^k) \Rightarrow \exists Q_i^k. \text{getQ}(Q_i^k, O_i^{k+1}) = Q_i^k \land \text{isQ}(q_i, Q_i^k, nvo_i, E_0^i, E_i^k)$$

Proof. Pick an arbitrary PTSO-valid execution $E = G_1; \cdots ; G_n$. Let $H_i$ and $C_i$ be as defined as above for all $i \in \{1 \cdots n\}$. Pick an arbitrary $i \in \{1 \cdots n\}, G_i = (E_i^0, E_i^p, E_i, p_{oi}, r_{fi}, t_{soi}, nvo_i)$ and $H_i$. We proceed by induction on $l$. 

Base case $l = 0, k = t_i$

Pick arbitrary $Q_i^0$ and $Q_i^k$ such that $\text{getQ}(Q_i^0, O_i^k) = Q_i^k$ and isQ($q_i, Q_i^k$, nvo, $E_i^0, E_i^k$). As $k = t_i$, we have isQ($q_i, Q_i^k$, nvo, $E_i^0, E_i^k$). As $O_{k+1}$ is $\varepsilon$, we have getQ($Q_i^k, O_{k+1}$) = $Q_i^k$, as required.

Inductive case $0 < l \leq t_i$

\[ \forall Q_i, \forall k > l. \quad \text{getQ}(Q_i^0, O_i^k) = Q_i^k \land \text{isQ}(q_i, Q_i^k, \text{nvo}, E_i^0, E_i^k) \implies \exists Q_i^l. \quad \text{getQ}(Q_i^l, O_{k+1}^l) = Q_i^l \land \text{isQ}(q_i, Q_i^l, \text{nvo}, E_i^0, E_i^k) \quad \text{(I.H.)} \]

Pick arbitrary $Q_i^0$ and $Q_i^k$ such that $\text{getQ}(Q_i^0, O_i^k) = Q_i^k$ and isQ($q_i, Q_i^k$, nvo, $E_i^0, E_i^k$). We are then required to show that there exists $Q_i^l$ such that $\text{getQ}(Q_i^l, O_{k+1}^l) = Q_i^l$ and isQ($q_i, Q_i^l$, nvo, $E_i^0, E_i^k$). We then know:

\[ O_{k+1}^l = H(c_i^k, t_i^k, p_i^k, n_i^k).\text{inv}.H(c_i^{k+1}, t_i^{k+1}, p_i^{k+1}, n_i^{k+1}).\text{ack}.O_{k+2}^l \]

There are now three cases to consider: 1) there exists $m$ such that $c_i^{k+1} = \text{enq}(m)$ and $n_i^{k+1} = m$; or 2) there exists $m \neq \text{null}$ such that $c_i^{k+1} = \text{seq}(Q_i^l)$ and $n_i^{k+1} = m$; or 3) $c_i^{k+1} = \text{seq}(Q_i^l)$ and $n_i^{k+1} = \text{null}$.

In case (1), as $\text{getQ}(Q_i^0, O_i^k) = Q_i^k$, from its definition we have $\text{getQ}(Q_i^0, O_i^{k+1}) = Q_i^k.m$. Let $Q_i^{k+1} = Q_i^k.m$. Given $H(c_i^{k+1}, t_i^{k+1}, p_i^{k+1}, n_i^{k+1})$, since from the PTSO-validity of $G_i$ we have $E_i^0 \times (E_i^0 \setminus E_i^l) \subseteq \text{nvo}$ and as isQ($q_i, Q_i^k, \text{nvo}, E_i^0, E_i^k$) holds, from its definition we have isQ($q_i, Q_i^{k+1}, \text{nvo}, E_i^0, E_i^k$). From (I.H.) we know there exists $Q_i^l$ such that $\text{getQ}(Q_i^{k+1}, O_{k+2}^l) = Q_i^l$ and isQ($q_i, Q_i^l$, nvo, $E_i^0, E_i^k$). As $\text{getQ}(Q_i^{k+1}, O_{k+2}^l) = Q_i^l$, from its definition we also have $\text{getQ}(Q_i^k, O_{k+1}^l) = Q_i^k$, as required.

In case (2), given the trace of $H(c_i^{k+1}, t_i^{k+1}, p_i^{k+1}, n_i^{k+1})$ we know that there exists $w, r$ such that $w = \text{w}(q_i, \text{data}[a], m), r = H(c_i^{k+1}, t_i^{k+1}, p_i^{k+1}, n_i^{k+1}).r$ and $(w, r) \in rf_i$. As $hb_i$ is acyclic and $G_i$ is PTSO-valid, we know either:

- i) $w \in E_i^0$ and for all $j \in \{1 \cdots k\}, H(c_i^j, t_i^j, p_i^j, n_i^j).E \cap (W \cup U)_{q_i, \text{data}[a]} = \emptyset$; or
- ii) exists $j \text{ s.t. } 1 \leq j \leq k$ and $w \in H(c_i^j, t_i^j, p_i^j, n_i^j)$ and for all $j' \in \{j+1 \cdots k\}, H(c_i^{j'}, t_i^{j'}, p_i^{j'}, n_i^{j'}).E \cap (W \cup U)_{q_i, \text{data}[a]} = \emptyset$.

As $E_i^0 \subseteq E_i^l$ and the events of $H(c_i^j, t_i^j, p_i^j, n_i^j)$ are persistent (discussed above in the construction of $H_i$), we know that $w \in E_i^l$. Moreover, as the lock events are totally ordered by $\text{Hb}_i$ and $\text{hb}_i \subseteq \text{po} \cup \text{tso}$ (Lemma E.2), given the placement of pFence instructions and the construction of the enumeration $C_i$, we know that for all locations $x$, if $w_1 = \text{w}(x, -) \in H(c_i^j, t_i^j, p_i^j, n_i^j), w_2 = \text{w}(x, -) \in H(c_i^j, t_i^j, p_i^j, n_i^j)$, and $f < g$, then $(w_1, w_2) \in \text{nvo}_i$. As such, in both cases we know that max($\text{nvo}_{E_i^l}(W \cup U)_{q_i, \text{data}[a]} = w$.

Moreover, since isQ($q_i, Q_i^k, \text{nvo}, E_i^0, E_i^k$) holds, we know that $\text{val}_q(\text{max}(\text{nvo}_{E_i^l}(W \cup U)_{q_i, \text{data}[a]})) = Q_i^k$.

We thus have $Q_i^{l+1} = \text{m}$.

Let $Q_i^k = m, Q'$ for some $Q'$ and let $Q_i^{k+1} = Q'$. As $\text{getQ}(Q_i^0, O_i^k)$ holds, from its definition we also have $\text{getQ}(Q_i^0, O_i^{k+1}) = Q_i^{k+1}$. Given the trace $H(c_i^{k+1}, t_i^{k+1}, p_i^{k+1}, n_i^{k+1})$, as isQ($q_i, Q_i^k, \text{nvo}, E_i^0, E_i^k$) holds, from its definition we have isQ($q_i, Q_i^{k+1}, \text{nvo}, E_i^0, E_i^{k+1}$). From (I.H.) we then know there exists $Q_i^l$ such that $\text{getQ}(Q_i^{k+1}, O_{k+2}^l) = Q_i^l$ and isQ($q_i, Q_i^l$, nvo, $E_i^0, E_i^k$). As $\text{getQ}(Q_i^{k+1}, O_{k+2}^l) = Q_i^l$, from its definition we also have $\text{getQ}(Q_i^k, O_{k+1}^l) = Q_i^k$, as required.

Case (3) is analogous to that of case (2) and is omitted here.

\[ \square \]

Corollary 2. Given a PTSO-valid execution $E = G_1; \cdots ; G_n$, let for all $i \in \{1 \cdots n\}$, $H_i$ be defined as above. For all $G_i = (E_i^0, E_i^p, E_i, \text{po}, \text{rf}_i, \text{ts}_i, \text{nvo})$, $H_i$ and for all $Q_i^k$:

isQ($q_i, Q_i^k, \text{nvo}, E_i^0, E_i^k$) \implies
\[ \exists Q^t_1 . \, \text{get}\{Q^0_1, H_1\} = Q^t_1 \land \text{is}\{q, Q^t_1, \text{nvo}_1, E^0_1, E^p_1\} \]

**Proof.** Follows immediately from the previous lemma when \( k = 0 \). \( \square \)

**Lemma B.2.** Given a PTSO-valid execution \( \mathcal{E} = G_1; \ldots; G_n \) if \( H = H_1; \ldots; H_n \) with \( H_i \) defined as above for all \( i \in \{1 \cdots n\} \), then:

\[ \exists Q_n. \, \text{get}\{\varepsilon, H\} = Q \]

**Proof.** Pick an arbitrary PTSO-valid execution \( \mathcal{E} = G_1; \ldots; G_n \), with \( H = H_1; \ldots; H_n \) and \( H_i \) defined as above for all \( i \in \{1 \cdots n\} \). Let \( Q_0 = \varepsilon \). By definition we then have \( \text{is}\{q, Q^0_1, \text{nvo}_1, E^0_1, E^p_1\} \).

On the other hand from Corollary 2 we have:

\[ \exists Q^t_1 . \, \text{get}\{Q^0_1, H_1\} = Q^t_1 \land \exists Q_1 \] \[ \exists Q^t_2 . \, \text{get}\{Q^0_2, H_2\} = Q^t_2 \land \exists Q_2 \] \[ \vdots \]

\[ \exists Q^t_n . \, \text{get}\{Q^0_n, H_n\} = Q^t_n \land \exists Q_n \]

For all \( j \in \{2 \cdots n\} \), let \( Q_j = \text{get}\{Q_{j-1}^0, H_{j-1}\} \). From above we then have:

\[ \exists Q^t_1, \ldots, Q^t_n . \, \text{get}\{Q^0_1, H_1\} = Q^t_1 \land \text{get}\{Q^0_2, H_2\} = Q^t_2 \land \cdots \land \text{get}\{Q^0_{n-1}, H_{n-1}\} = Q^t_{n-1} \land \text{get}\{Q^0_n, H_n\} = Q^t_n \]

From its definition we thus know there exists \( Q^t_n \) such that \( \text{get}\{Q^0_1, H_1; \ldots; H_n\} = Q^t_n \). That is, there exists \( Q \) such that \( \text{get}\{\varepsilon, H\} = Q \), as required.

\( \square \)

**Theorem 7.** For all client programs \( P \) of the queue library (comprising calls to \text{enq} and \text{deq} only) and all PTSO-valid executions \( \mathcal{E} \) of \( \text{start}(P) \), \( \mathcal{E} \) is persistently linearisable.

**Proof.** Pick an arbitrary program \( P \) and a PTSO-valid execution \( \mathcal{E} = G_1; \ldots; G_n \) of \( P \). For each \( i \in \{1 \cdots n\} \), construct \( T_I \) and \( H_I \) as above. It then suffices to show that:

\[ \forall i \in \{1 \cdots n\} . \, \forall a, b \in T_i . (a, b) \in \text{hb}_i \Rightarrow a \prec_{H_i} b \] (34)

\[ \text{fifo}(\varepsilon, H) \text{ holds when } H = H_1; \ldots; H_n \] (35)

**TS.** (34)

Pick arbitrary \( i \in \{1 \cdots n\} \), \( a, b \in T_i \) such that \( (a, b) \in \text{hb}_i \). We then know there exist \( c, r, p, n, c', r', p', n' \) such that \( a \in H(c, r, p, n), b \in H(c', r', p', n') \) and either:

1. \( H(c, r, p, n) \land H(c', r', p', n') \)
2. \( H(c, r, p, n) \land H(c', r', p', n') \)
3. \( H(c, r, p, n) \land H(c', r', p', n') \)
4. \( H(c, r, p, n) \land H(c', r', p', n') \)
5. \( H(c, r, p, n) \land H(c', r', p', n') \)
6. \( H(c, r, p, n) \land H(c', r', p', n') \)

In case (1) the desired result holds immediately. In case (2) we have \( b \overset{\text{po}_i}{\rightarrow} a \overset{\text{hb}_j}{\rightarrow} b \), and since \( \text{po}_i \subseteq \text{hb}_j \) we have \( b \overset{\text{hb}_j}{\rightarrow} a \overset{\text{hb}_i}{\rightarrow} b \). Consequently, from the transitivity of \( \text{hb}_i \) we have \( (b, b) \in \text{hb}_i \), contradicting the acyclicity of \( \text{hb}_i \) in Lemma E.1.

In case (3) from the totality of \( \text{hb}_i \) on lock events (see above), we know that either i) \( H(c, r, p, n).\text{qu}, H(c', r', p', n').\text{ql} \in \text{hb}_i \); or ii) \( H(c', r', p', n').\text{qu}, H(c, r, p, n).\text{ql} \in \text{hb}_i \). In case (3.i) from the construction of \( C_i \) we know that \( a \prec_{H_i} b \), as required.

In case (3.ii), as \((a, b) \in \text{hb}_1\) and \(H(c, \tau, p, n) \neq H(c', \tau', p', n')\), we know there exists \(w, r, d, e, w', r'\) such that either:

a) \(d \notin H(c, \tau, p, n), e \notin H(c', \tau', p', n')\) and \(a \xrightarrow{\text{poj}} d \xrightarrow{\text{hb}_1} e \xrightarrow{\text{poj}} b; or\)

b) \(w \in W \cap H(c, \tau, p, n), e \notin H(c', \tau', p', n')\) and \(a \xrightarrow{\text{poj}} H(c, \tau, p, n).qI \xrightarrow{\text{hb}_1} w \xrightarrow{rf_j} r \xrightarrow{\text{poj}} e \xrightarrow{\text{poj}} b; or\)

c) \(r \in R \cap H(c', \tau', p', n'), d \notin H(c, \tau, p, n)\) and \(a \xrightarrow{\text{poj}} d \xrightarrow{\text{hb}_1} w \xrightarrow{rf_j} r \xrightarrow{\text{poj}} H(c', \tau', p', n').qU \xrightarrow{\text{poj}} b; or\)

In case (4) we then have \(H(c, \tau, p, n).qI \rightarrow H(c, \tau, p, n).qI \rightarrow H(c', \tau', p', n').qU\). In case (5) we then have \(H(c, \tau, p, n).qI \rightarrow H(c, \tau, p, n).qI \rightarrow H(c', \tau', p', n').qU\). In case (6) we then have \(H(c, \tau, p, n).qI \rightarrow H(c, \tau, p, n).qI \rightarrow H(c', \tau', p', n').qU\).

Then have \(H(c, \tau, p, n).qI \rightarrow H(c, \tau, p, n).qI \rightarrow H(c', \tau', p', n').qU\). From the transitivity of \(\text{hb}_1\) we have \(H(c, \tau, p, n).qI \rightarrow H(c', \tau', p', n').qU\), i.e. from the transitivity of \(\text{hb}_1\) we have \(H(c, \tau, p, n).qI \rightarrow H(c', \tau', p', n').qU\).

We next demonstrate that in all four cases (a-d) we have \(H(c, \tau, p, n).qI \rightarrow H(c', \tau', p', n').qU\). We then have \(H(c, \tau, p, n).qI \rightarrow H(c', \tau', p', n').qU\). From the transitivity of \(\text{hb}_1\) we have \(H(c, \tau, p, n).qI \rightarrow H(c, \tau, p, n).qI \rightarrow H(c', \tau', p', n').qU\). As such we have \(H(c, \tau, p, n).qI \rightarrow H(c', \tau', p', n').qU\).

In case (4) we then have \(a \xrightarrow{\text{hb}_1} b \xrightarrow{\text{poj}} H(c', \tau', p', n').aCk\), and thus as \(\text{poj} \subseteq \text{hb}_1\) and \(\text{hb}_1\) is transitively closed, we have \(a \xrightarrow{\text{hb}_1} H(c', \tau', p', n').aCk\). As such, from the proof of part (3) we have \(a <_{H_I} H(c', \tau', p', n').aCk\), and consequently since \(H(c, \tau, p, n) \neq H(c', \tau', p', n')\), from the construction \(H_I\) we have \(a <_{H_I} b\), as required.

In case (5) we then have \(H(c, \tau, p, n).iNv \xrightarrow{\text{poj}} a \xrightarrow{\text{hb}_1} b \xrightarrow{\text{poj}} H(c', \tau', p', n').aCk\), and thus as \(\text{poj} \subseteq \text{hb}_1\) and \(\text{hb}_1\) is transitively closed, we have \(a \xrightarrow{\text{hb}_1} H(c', \tau', p', n').aCk\). As such, from the proof of part (3) we have \(H(c, \tau, p, n).iNv <_{H_I} H(c', \tau', p', n').aCk\), and consequently since \(H(c, \tau, p, n) \neq H(c', \tau', p', n')\), from the construction \(H_I\) we have \(a <_{H_I} b\), as required.

In case (6) we then have \(H(c, \tau, p, n).iNv \xrightarrow{\text{poj}} a \xrightarrow{\text{hb}_1} b\), and thus as \(\text{poj} \subseteq \text{hb}_1\) and \(\text{hb}_1\) is transitively closed, we have \(H(c, \tau, p, n).iNv \rightarrow b\).

We know that when \(P\) comprises \(k\) threads, the trace of each execution

\[ TS(i) \]

From Lemma B.2 we know there exists \(Q\) such that \(\text{getQ}(\epsilon, H) = Q\). From the definition of \(\text{fifo}(\epsilon, H)\), we know \(\text{fifo}(\epsilon, H)\) holds if and only if there exists \(Q\) such that \(\text{getQ}(\epsilon, H) = Q\). As such we have \(\text{fifo}(\epsilon, H)\), as required.
era (via start() or recover()) comprises two stages: i) the trace of the setup stage by the master thread \(\tau_0\) performing initialisation or recovery, prior to the call to run(\(\mathcal{P}\)); followed (in po order) by ii) the trace of each of the constituent program threads \(\tau_1 \cdots \tau_k\), provided that the execution did not crash during the setup stage.

As before, thanks to the placement of the persistent fence operations (pFence), for each thread \(\tau_j\), we know that the set of persistent events in execution era \(i\), namely \(E^p_i\), contains roughly a prefix (in po order) of thread \(\tau_j\)'s trace. More concretely, for each constituent thread \(\tau_j \in \{\tau_1 \cdots \tau_k\} = \text{dom}(\mathcal{P}),\) there exist \(P_i^1 \cdots P_i^n\) such that:

1) \(P[\tau_j] = o_j^0; \cdots ; o_j^i; p_j^{i+1}; \cdots ; o_j^{p-1}; \cdots ; o_j^{p+1-n}; \cdots ; o_j^p\), comprising enq and deq operations; and

2) at the beginning of each execution era \(i \in \{1 \cdots n\}\), the program executed by thread \(\tau_j\) (calculated in \(\mathcal{P}^i\) and subsequently executed by calling run(\(\mathcal{P}^i\))) is that of sub(\(P[\tau_j], P_j^{-1}+1\)), where \(P_j^0 = -1\), for all \(j\); and

3) in each execution era \(i \in \{1 \cdots n\}\), the trace \(H_{(i,j)}\) of each constituent thread \(\tau_j \in \text{dom}(\mathcal{P})\) is of the following form:

\[ H_{(i,j)} = \text{H}(o_j^{p_j^{-1}+1}, \tau_j, P_j^{-1}+n_j, n_j, o_j^{p_j^{-1}+1}) \]

\[ \xrightarrow{p_0} \cdots \xrightarrow{p_0} \text{H}(o_j^{p_j^{-1}+1}, \tau_j, P_j+n_j, n_j, o_j^{p_j^{-1}+1}) \]

\[ \xrightarrow{p_0} \cdots \xrightarrow{p_0} \text{H}(o_j^{m_j^{-1}}, \tau_j, P_j+n_j, n_j, o_j^{m_j^{-1}}) \]

\[ \xrightarrow{p_0} \cdots \xrightarrow{p_0} \text{H}(o_j^{m_j^{-1}}, \tau_j, P_j+n_j, n_j, o_j^{m_j^{-1}}) \]

\[ \xrightarrow{p_0} \cdots \xrightarrow{p_0} \text{H}(o_j^{m_j^{-1}}, \tau_j, P_j+n_j, n_j, o_j^{m_j^{-1}}) \]

for some \(m_j^1, n_j^1, P_j^1, \cdots, n_j^1, P_j^1, \cdots, n_j^1, P_j^1, \cdots, e_j^1, P_j^1, \cdots, e_j^1\) where:

- The first two lines denote the execution of the \((P_j^{-1}+1)^{\text{th}}\) to \((P_j^1)^{\text{th}}\) library calls of thread \(\tau_j\), with \(H(o, \tau, p, n, e)\) defined shortly. Moreover, before crashing and proceeding to the next era, all volatile events (those in PE) in \(H(o_j^{P_j^{-1}+1}, \cdots) \xrightarrow{p_0} \cdots \xrightarrow{p_0} \text{H}(o_j^{P_j^{-1}+1}, \cdots)\) have persisted, and a prefix (in po order) of the volatile events in \(H(o_j^{P_j^{-1}+1}, \tau_j, P_j+n_j, n_j, e_j^{P_j^{-1}})\) have persisted. Note that this prefix may be equal to \(H(o_j^{P_j^{-1}+1}, \tau_j, P_j+n_j, n_j, e_j^{P_j^{-1}})\), in which case all its events have persisted.

- The next two lines denote the execution of the subsequence of library calls of thread \(\tau_j\) where \(m_j^1 \leq P_j^n\), with none of their volatile events having persisted.

- The last line denotes the execution of the \((m_j^1)^{\text{th}}\) call of thread \(\tau_j (m_j^1 \leq P_j^n)\), during which the program crashed and thus the execution of era \(i\) ended. As before, the \(H'(o, \tau, p, n, e)\) denotes a (potentially full) prefix of \(H(o, \tau, p, n, e)\).

The trace \(H(o, \tau, p, n, e)\) of each library call is defined as follows:

\[ H(\text{deq}(), \tau, p, n, h) \xrightarrow{\text{inv}=I(\tau, \text{deq}(), \text{h})} R(p, c, \text{p}) \xrightarrow{\text{po}} R(\text{tid} \tau, \text{p}) \xrightarrow{\text{po}} FE \]

\[ \xrightarrow{\text{po}} \text{r}_h = R(q, \text{head}, h) \xrightarrow{\text{po}} \text{r}=R(q, \text{data} \{h\}, n) \]

\[ \xrightarrow{\text{po}} S_0 \xrightarrow{\text{no}} \text{lin}_1 = \text{w}(\text{map} \{\text{r}\}[p], n) \xrightarrow{\text{po}} S_1 \xrightarrow{\text{po}} \text{PF} \xrightarrow{\text{po}} S_2 \]

\[ \xrightarrow{\text{po}} \text{ack}=\text{A}(\text{lp}, \text{deq} \{\text{n}\}) \]

where \(FE\) denotes the sequence of events, attempting but failing to set the \(\text{rem}\) field of the head node, with

\[ S_0 = \begin{cases} \emptyset & \text{if } n = \text{null} \\ \cup(n.\text{rem}, \text{null}, \tau) & \text{otherwise} \end{cases} \]
\[ S_1 = \begin{cases} 
\emptyset & \text{if } n = \text{null} \\
R(n.t, \tau') \xrightarrow{po} R(n.pc, p') \xrightarrow{po} W(\text{map}[\tau'][p'], \top) & \text{otherwise}
\end{cases} \]

\[ S_2 = \begin{cases} 
\emptyset & \text{if } n = \text{null} \\
\text{lin}_2=W(q.\text{head}, h+1) \xrightarrow{po} \text{PF} & \text{otherwise}
\end{cases} \]

for some \( \tau', p' \); and

\[ H(\text{enq}(v), \tau, p, n, e) \triangleq \begin{cases} 
inv=I(tp, \text{enq}. n) \xrightarrow{po} R(p.c, p) \xrightarrow{po} R(\text{tid}, \tau) \\
\xrightarrow{po} W(n.\text{val}, v) \xrightarrow{po} W(n.\text{tid}, \tau) \xrightarrow{po} W(n.p.c, p) \xrightarrow{po} W(n.\text{rem}, \text{null}) \\
\xrightarrow{po} W(\text{map}[\tau'][p], n) \xrightarrow{po} \text{PF} \xrightarrow{po} R(q.\text{head}, h) \\
\xrightarrow{po} R(q.\text{data}[h], \text{v}_0) \xrightarrow{po} A_0 \cdots R(q.\text{data}[h+s-1], \text{v}_{s-1}) \xrightarrow{po} A_{s-1}
\end{cases} \]

\( \xRightarrow{s \text{ times}} \)

\[ \xrightarrow{po} R(q.\text{data}[h+s], \text{null}) \xrightarrow{po} \text{lin}=U(q.\text{data}[h+s], \text{null}, n) \xrightarrow{po} \text{PF} \xrightarrow{po} \text{ack}=A(tp, \text{enq}.()) \]

for some \( s \geq 0 \) such that \( h+s = e \), and for all \( k \in \{0 \cdots s-1\} \), \( \text{v}_k \neq \text{null} \) and \( \text{A}_k = \emptyset \); or \( \text{v}_k = \text{null} \) and \( \text{A}_k = R[q.\text{data}[h+k], \text{v}_k'] \) with \( \text{v}_k' \neq \text{null} \). In the above traces, for brevity we have omitted the thread identifiers (\( \tau_j \)) and event identifiers and represent each event with its label only.

We use the \( H(\text{enq}(\cdot), \tau, p, n, e) \) prefix to extract its specific events, e.g. \( H(\text{enq}(\cdot), \tau, p, n, e).\text{inv} \).

Let us write \( q.\text{tail} \) to denote the index of the last entry in the queue. Observe that each \( \text{enq} \) operation leaves the \( q.\text{head} \) value unchanged while increasing \( q.\text{tail} \) by 1. Similarly, each \( \text{deq} \) operation leaves \( q.\text{tail} \) unchanged while increasing \( q.\text{head} \) by one.

Note that in each \( H(\text{enq}(\cdot), \tau, p, n, e) \), the \( e-1 \) denotes the value of \( q.\text{tail} \) immediately before the insertion of node \( n \) by \( H(\text{enq}(\cdot), \tau, p, n, e) \), i.e. the \( e \) denotes the value of \( q.\text{tail} \) immediately after the insertion of node \( n \) by \( H(\text{enq}(\cdot), \tau, p, n, e) \).

Similarly, in each \( H(\text{deq}(), \tau, p, n, h) \), the \( h \) denotes the value of \( q.\text{head} \) immediately before the removal of node \( n \) by \( H(\text{deq}(), \tau, p, n, h) \).

Let:

\[ 1p(H(o, \tau, p, n, e)) \triangleq \begin{cases} 
H(o, \tau, p, n, e).\text{lin} & \text{if } o=\text{enq}(v) \\
H(o, \tau, p, n, e).\text{lin}_1 & \text{if } o=\text{deq}() \text{ and } H(o, \tau, p, n, e).S_2=\emptyset \\
H(o, \tau, p, n, e).\text{lin}_2 & \text{if } o=\text{deq}() \text{ and } H(o, \tau, p, n, e).S_2\neq\emptyset
\end{cases} \]

For each \( \tau_j \in \text{dom}(P) \) let:

\[ E^P_{(i,j)} = E^P_i \cap \{ e \mid \text{tid}(e) = \tau_j \} \quad E'_{(i,j)} = E^P_{(i,j)} \cup S_{(i,j)} \]

where

\[ S_{(i,j)} \triangleq \left\{ \exists a, o, p, n, \text{inv}, e. \\
\text{inv} = I(i.m.a) \text{ max } \left( \text{nvo}, S_{(i,j)}^{(\tau_j)} \right) \\
\wedge \text{inv} \in H(o, \tau, j, p, n, e) \wedge \forall r'. \text{\text{A}}(i.m, r') \notin E_{(i,j)}^{P} \\
\wedge 1p(H(o, \tau, j, p, n, e)) \in E_{(i,j)}^{P} \\
\wedge (m=\text{deq} \Rightarrow r=r) \wedge (m=\text{enq} \Rightarrow r=()) \right\} \]

Let \( E'_i = \bigcup_{\tau_j \in \text{dom}(P)} E'_{(i,j)} \). From the definition of each \( E'_{(i,j)} \) and \( E^P_{(i,j)} \) we then know that \( E^P_{i} \subseteq E'_{i} \) and \( E'_i \in \text{comp}(E^P_i) \). Let \( T_i = \text{trunc}(E'_i) \).

Let \( C_i \) denote an enumeration of \( \bigcup_{\tau_j \in \text{dom}(P)} \{ H(o, j, \tau_j, p, \text{deq}(), n, e) \} \) that respects memory order (in \( \text{tsos} \)) of linearisation points. That is, for all \( H(o, \tau, j, p, n, e), H(o', \tau', j', p', n', e') \), if \( 1p(H(o, \tau, j, p, n, e)) \rightarrow 1p(H(o', \tau', j', p', n', e')) \), then \( H(o, \tau, j, p, n, e) < C_i H(o', \tau', j', p', n', e') \).
When $C_i$ is enumerated as $C_i = H(c_i', r_i, p_i', n_i', e_i') \cdots H(c_i^j, r_i^j, p_i^j, n_i^j, e_i^j)$, let us define

$$H_i = H(c_i', r_i, p_i', n_i', e_i'). \text{inv} \cdot H(c_i^j, r_i^j, p_i^j, n_i^j, e_i^j). \text{ack} \cdots H(c_i^j, r_i^j, p_i^j, n_i^j, e_i^j). \text{inv} \cdot H(c_i^j, r_i^j, p_i^j, n_i^j, e_i^j). \text{ack}$$

**Lemma C.1.** Given a PTSO-valid execution $E = G_1; \cdots ; G_n$, let for all $i \in \{1 \cdots n\}$, $C_i$ be as defined above. Then, for all $H(o, r, p, n, e), H(o', r', p', n', e')$, $a, b, c, d$ such that $a \in H(o, r, p, n, e)$ and $b \in H(o', r', p', n', e')$, $C_i|_a = H(o, r, p, n, e)$, $C_i|_d = H(o, r, p, n, e')$ and $(a, b) \in hb_i$, then either 1) $c = d$ and $(a, b) \in po_i$ or 2) $c < d$.

**Proof.** Pick an arbitrary PTSO-valid execution $E = G_1; \cdots ; G_n$, with $C_i$ defined as above for all $i \in \{1 \cdots n\}$. Let $hb_i^0 = po_i \cup rf_i$ and $hb_i^{j+1} = hb_i^0 \cup hb_i^j$ for all $j \in \mathbb{N}$. It is then straightforward to demonstrate that $hb_i = \bigcup_{j \in \mathbb{N}} hb_i^j$. As such, it suffices to show that for all $j \in \mathbb{N}$, $H(o, r, p, n, e), H(o', r', p', n', e')$, $a, b, c, d$:

$$a \in H(o, r, p, n, e) \land b \in H(o', r', p', n', e') \land (a, b) \in hb_i^j \land$$

$$C_i|_a = H(o, r, p, n, e) \land C_i|_d = H(o', r', p', n', e')$$

$$\Rightarrow (c = d \land (a, b) \in po_i) \lor c < d$$

We thus proceed by induction on $j$.

**Base case $j = 0$**

Pick arbitrary $H(o, r, p, n, e), H(o', r', p', n', e')$, $a, b, c, d$ such that $a \in H(o, r, p, n, e)$ and $b \in H(o', r', p', n', e')$, $C_i|_a = H(o, r, p, n, e)$, $C_i|_d = H(o', r', p', n', e')$ and $(a, b) \in hb_i^0$.

There are now 5 cases to consider: 1) $c = d$; or 2) $c \neq d$, $a = \text{enq}(v)$ and $o' = \text{enq}(v')$ for some $v, v'$; or 3) $c \neq d$, $o = \text{enq}(v)$ and $o' = \text{deq}(v)$ for some $v$; or 4) $c \neq d$, $o = \text{deq}(v)$ and $o' = \text{enq}(v')$ for some $v'$; or 5) $c \neq d$, $o = \text{deq}(v)$ and $o' = \text{deq}(v)$.

In case 1) we then know that either $(a, b) \in po_i$ or $(a, b) \in po_i$. In the former case the desired result holds immediately. In the latter case we then have $a \rightarrow b \rightarrow a$, i.e. $(a, a) \in hb_i$, contradicting the assumption that $hb_i$ is acyclic (Lemma E.1).

In case (2), there are two more cases to consider: i) $(a, b) \in po_i$, or ii) $(a, b) \in rf_i$. In case (2,i), we then know that $1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e'))$. As both linearisation points are in $W \cup U$, from the PTSO-validity of $G_i$ we also know that $1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e'))$.

As such, from the definition of $C_i$ we know that $c < d$, as required.

In case (2.ii) we know that either $a = r'$ or $b = r'$. In case (2.ii,a) we then have $(a, b) \in po_i$ (since otherwise we would have a cyclic $hb_i$) and thus from the proof of part (2.i) we have $c < d$ as required. In case (2.ii,b) we then know that $a = 1p(H(o, r, p, n, e))$, i.e. $\text{loc}(a) = \text{q.val}(e)$. Moreover, from the PTSO-validity of $G_i$ and since $(a, b) \in rf_i$ we know that $(a, b) \in ts_0$. On the other hand, from the shape of $\text{enq}$ traces we know that $(b, 1p(H(o', r', p', n', e')))) \in po_i$ and thus from the PTSO-validity of $G_i$ we have $(b, 1p(H(o', r', p', n', e')))) \in ts_0$. We thus have $a \rightarrow b \rightarrow 1p(H(o', r', p', n', e'))$ and thus from the transitivity of $ts_0$ we have $a = 1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e'))$. As such, from the definition of $C_i$ we know that $c < d$, as required.

In case (3) there are two more cases to consider: i) $(a, b) \in po_i$, or ii) $(a, b) \in rf_i$. In case (3,i), we then know that $1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e'))$. As both linearisation points are in $W \cup U$, from the PTSO-validity of $G_i$ we also know that $1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e'))$.

As such, from the definition of $C_i$ we know that $c < d$, as required.
In case (3.ii) we know that either a) \( r = r' \) or b) \( r \neq r' \). In case (3.ii.a) we then have \((a, b) \in po_i\) (since otherwise we would have a cyclic \( hb_j \)) and thus from the proof of part (3.i) we have \( c < d \) as required.

In case (3.ii.b) we then know that either 1) \( a = 1p(H(o, r, p, n, e)) \), \( b = H(o', r', p', n', e') \), i.e. \( e = e' \); or 2) \( 1oc(a) = n.t \) or \( 1oc(a) = n.p.c \). In case (3.ii.b.1) from the PTSO-validity of \( G_i \) and since \((a, b) \in rf_i\) we know that \((a, b) \in tso_i\). On the other hand, from the shape of seq traces we know that \((b, 1p(H(o', r', p', n', e'))) \in po_i\). Thus from PTSO-validity of \( G_i \) we have \((b, 1p(H(o', r', p', n', e'))) \in tso_i\). We thus have \( a \rightarrow b \rightarrow 1p(H(o', r', p', n', e')) \) and thus from the transitivity of \( tso_i \) we have \( a = 1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e')).\) As such, from the definition of \( C_i \) we know that \( c < d \), as required.

In case (3.ii.b.2) from the shape of the traces we also know \((1p(H(o, r, p, n, e)), H(o', r', p', n', e'), r)) \in rf_i\) and thus from the proof of part (3.ii.b.1) we have \( c < d \), as required.

In case (4) there are two more cases to consider: i) \((a, b) \in po_i\), or ii) \((a, b) \in rf_i\). In case (4.i), we then know that \(1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e'))\). As both linearisation points are in \( W \cup U \), from the PTSO-validity of \( G_i \) we also know that \(1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e')).\) As such, from the definition of \( C_i \) we know that \( c < d \), as required.

In case (4.ii) we know that either a) \( r = r' \) or b) \( r \neq r' \). In case (4.ii.a) we then have \((a, b) \in po_i\) (since otherwise we would have a cyclic \( hb_j \)) and thus from the proof of part (4.i) we have \( c < d \) as required.

In case (4.ii.b) we then know that \( a = 1p(H(o, r, p, n, e)) \). From the PTSO-validity of \( G_i \) and since \((a, b) \in rf_i\) we know that \((a, b) \in tso_i\). On the other hand, from the shape of seq traces we know that \((b, 1p(H(o', r', p', n', e'))) \in po_i\) and thus from the PTSO-validity of \( G_i \) we have \((b, 1p(H(o', r', p', n', e'))) \in tso_i\). We thus have \( a \rightarrow b \rightarrow 1p(H(o', r', p', n', e')) \) and thus from the transitivity of \( tso_i \) we have \( a = 1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e')).\) As such, from the definition of \( C_i \) we know that \( c < d \), as required.

In case (5) there are two more cases to consider: i) \((a, b) \in po_i\), or ii) \((a, b) \in rf_i\). In case (5.i), we then know that \(1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e'))\). As both linearisation points are in \( W \cup U \), from the PTSO-validity of \( G_i \) we also know that \(1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e')).\) As such, from the definition of \( C_i \) we know that \( c < d \), as required.

In case (5.ii) we know that either a) \( r = r' \) or b) \( r \neq r' \). In case (5.ii.a) we then have \((a, b) \in po_i\) (since otherwise we would have a cyclic \( hb_j \)) and thus from the proof of part (5.i) we have \( c < d \) as required.

In case (5.ii.b) we then know that \( a = 1p(H(o, r, p, n, e)) \) From the PTSO-validity of \( G_i \) and since \((a, b) \in rf_i\) we know that \((a, b) \in tso_i\). On the other hand, from the shape of seq traces we know that \((b, 1p(H(o', r', p', n', e'))) \in po_i\) and thus from the PTSO-validity of \( G_i \) we have \((b, 1p(H(o', r', p', n', e'))) \in tso_i\). We thus have \( a \rightarrow b \rightarrow 1p(H(o', r', p', n', e')) \) and thus from the transitivity of \( tso_i \) we have \( a = 1p(H(o, r, p, n, e)) \rightarrow 1p(H(o', r', p', n', e')).\) As such, from the definition of \( C_i \) we know that \( c < d \), as required.

Inductive case \( j = m+1 \)

\[ \forall j' \in \mathbb{N}. \forall H(o, r, p, n, e), H(o', r', p', n', e'), a, b, c, d. \]
\[ j' \leq m \land a \in H(o, r, p, n, e) \land b \in H(o', r', p', n', e') \land (a, b) \in hb_j^/ \]
\[ \land C_j^/[c] = H(o, r, p, n, e) \land C_j^/[d] = H(o', r', p', n', e') \]
\[ \Rightarrow (c = d \land (a, b) \in po_i) \lor c < d \] (L.H.)
Pick arbitrary $H(o, \tau, p, n, e), H(o', \tau', p', n', e'), a, b, c, d$ such that $a \in H(o, \tau, p, n, e)$ and $b \in H(o', \tau', p', n', e')$. Let $C^k|_c = H(o, \tau, p, n, e)$, $C^k|_d = H(o', \tau', p', n', e')$ and $(a, b) \in hb_i$. From the definition of $hb_i$, we then know there exists $f$ such that $(a, f) \in hb_0$ and $(f, b) \in hb_m$. We thus know there exists $H(o'', \tau'', p'', n'', e'')$ and $g$ such that $f \in H(o'', \tau'', p'', n'', e'')$ and $C^k|_g = H(o'', \tau'', p'', n'', e'')$. From the proof of the base case we then know that $(c = g \wedge (a, f) \in po_i) \vee c < g$.

Similarly, from (I.H.) we know $(g = d \wedge (f, b) \in po_i) \vee g < d$. There are then four cases to consider:

1. $(c = g \wedge (a, f) \in po_i)$ and $(g = d \wedge (f, b) \in po_i)$; or 2) $(c = g \wedge (a, f) \in po_i)$ and $g < d$; or 3) $c < g$ and $(g = d \wedge (f, b) \in po_i)$; or 4) $c < g$ and $g < d$.

In case (1) from the transitivity of $\wedge$ and $po_i$ we have $c = d \wedge (a, b) \in po_i$, as required. In case (2) since $c = g$ and $g < d$ we have $c < d$, as required. In case (3) since $c < g$ and $g = d$ we have $c < d$, as required. In case (4) from the transitivity of $< \wedge$ we have $c < d$, as required.

\[\square\]

Lemma C.2. Given a PTSO-valid execution $E = G_1; \cdot \cdot \cdot ; G_n$, let for all $i \in \{1 \cdot \cdot \cdot n\}$, $H_i$ be defined as above with $C_1 = H(c_1^1, i_1^1, p_1^i, n_1^i, e_1^i), \cdot \cdot \cdot , H(c_i^i, i_i^i, p_i^i, n_i^i, e_i^i)$. For all $i \in \{1 \cdot \cdot \cdot n\}$, and $a, b$, let $O_i^k = H(c_a^i, i_a^i, p_a^i, n_a^i, e_a^i).inv.H(c_b^i, i_b^i, p_b^i, n_b^i, e_b^i).ack. \cdot \cdot \cdot , H(c_b^i, i_b^i, p_b^i, n_b^i, e_b^i).ack.H(c_b^i, t_i^i, p_b^i, n_b^i, e_b^i).ack.H(c_b^{i+1}, t_i^{i+1}, p_b^{i+1}, n_b^{i+1}, e_b^{i+1}).ack$.

For all $G_i = (E_i^n, E_i^n, E_i^0, pt, tf, ts, nvo_i), H_i$, for all $O_i^k$ and for all $l \in \{0 \cdot \cdot \cdot t_i\}$, $k = t_i + l, E_i^k = E_i^l \setminus \bigcup_{x = k+1}^{t_i} H(c_x^i, i_x^i, p_x^i, n_x^i, e_x^i).E$, and $Q_i^k$:

\[
\begin{align*}
\text{get}(Q_i^k, O_i^k) &= Q_i^k \land \text{isQ}(q, Q_i^k, nvo_i, E_i^0, E_i^k) \Rightarrow \\
\exists Q_i^k. \text{get}(Q_i^k, O_i^k) &= Q_i^l \land \text{isQ}(q, Q_i^l, nvo_i, E_i^0, E_i^l) 
\end{align*}
\]

Proof. Pick an arbitrary PTSO-valid execution $E = G_1; \cdot \cdot \cdot ; G_n$. Let $H_i$ and $C_i$ be as defined as above for all $i \in \{1 \cdot \cdot \cdot n\}$. Pick an arbitrary $i \in \{1 \cdot \cdot \cdot n\}$. $G_i = (E_i^n, E_i^n, E_i^0, pt, tf, ts, nvo_i) and H_i$. We proceed by induction on $l$.

Base case $l = 0, k = t_i$

Pick arbitrary $Q_i^0$ and $Q_i^k$ such that $\text{get}(Q_i^0, O_i^k) = Q_i^k$ and $\text{isQ}(q, Q_i^k, nvo_i, E_i^0, E_i^k)$. As $k = t_i$, we have $\text{isQ}(q, Q_i^k, nvo_i, E_i^0, E_i^k)$. As $O_i^{t_i} = e$, we have $\text{get}(Q_i^k, O_i^{t_i+1}) = Q_i^k$, as required.

Inductive case $0 < l \leq t_i$

\[
\forall Q. \forall k' > k. \text{get}(Q_i^0, O_i^k') = Q \land \text{isQ}(q, Q_i^0, nvo_i, E_i^0, E_i^k') \Rightarrow \\
\exists Q_i^k. \text{get}(Q_i^k, O_i^{k'+1}) = Q_i^k \land \text{isQ}(q, Q_i^k, nvo_i, E_i^0, E_i^l) (\text{I.H.})
\]

Pick arbitrary $Q_i^k$ and $Q_i^{k+1}$ such that $\text{get}(Q_i^k, O_i^k) = Q_i^k$ and $\text{isQ}(q, Q_i^k, nvo_i, E_i^0, E_i^k)$. We are then required to show that there exists $Q_i^l$ such that $\text{get}(Q_i^k, O_i^{l+1}) = Q_i^l$ and $\text{isQ}(q, Q_i^l, nvo_i, E_i^0, E_i^l)$. We then know:

\[
O_i^{t_i} = H(c_i^{k+1}, i_i^{k+1}, p_i^{k+1}, n_i^{k+1}, e_i^{k+1}).inv.H(c_i^{k+1}, i_i^{k+1}, p_i^{k+1}, n_i^{k+1}, e_i^{k+1}).ack.O_i^{t_i+2}
\]

There are now three cases to consider: 1) there exists $m$ such that $c_i^{k+1} = \text{enq}(m)$ and $n_i^{k+1} = m$; or 2) there exists $m \neq \text{null}$ such that $c_i^{k+1} = \text{deq}()$ and $n_i^{k+1} = m$; or 3) $c_i^{k+1} = \text{deq}()$ and $n_i^{k+1} = \text{null}$.

In case (1), as $\text{get}(Q_i^0, O_i^k) = Q_i^k$, from its definition we have $\text{get}(Q_i^0, O_i^{k+1}) = Q_i^{k+1}$. Let $Q_i^{k+1} = Q_i^k.m$. Given the trace $H(c_i^{k+1}, i_i^{k+1}, p_i^{k+1}, n_i^{k+1}, e_i^{k+1})$, since from the PTSO-validity of $G_i$ we have $E_i^n \times (E_i^n \setminus E_i^k) \subseteq nvo_i$ and as $\text{isQ}(q, Q_i^{k+1}, nvo_i, E_i^0, E_i^{k+1})$ holds, from its definition we have $\text{isQ}(q, Q_i^{k+1}, nvo_i, E_i^0, E_i^{k+1})$. From (I.H.) we know there exists $Q_i^l$ such that $\text{get}(Q_i^{k+1}, O_i^{k+2}) = Q_i^l$ and $\text{isQ}(q, Q_i^l, nvo_i, E_i^0, E_i^l)$. As $\text{get}(Q_i^{k+1}, O_i^{k+2}) = Q_i^l$, by definition we also have $\text{get}(Q_i^0, O_i^{k+1})$
Lemma C.3. Given a PTSO-valid execution \( E = G_1; \cdots ; G_n \) with \( H = H_1 \cdots . H_n \) defined as above for all \( i \in \{1 \cdots n\} \), then:

\[ \exists Q \text{. getQ}(\epsilon, H) = Q \]

Proof. Pick an arbitrary PTSO-valid execution \( E = G_1; \cdots ; G_n \) with \( H = H_1 \cdots . H_n \) and \( H_i \) defined as above for all \( i \in \{1 \cdots n\} \). Let \( Q_0 = \epsilon \). By definition we then have \( \text{isQ}(q, Q_0, nvo, E_1, E_1^0) \).

On the other hand from Corollary 3 we have:

\[ \exists Q^i_1 \text{. getQ}(Q^0_1, H_1) = Q^i_1 \wedge \text{isQ}(q, Q^i_1, nvo, E_1, E_1^0) \]

\[ \forall Q^0_2 \text{, isQ}(q, Q^0_2, nvo_2, E_2, E_2^0) \Rightarrow \exists Q^i_2 \text{. getQ}(Q^0_2, H_2) = Q^i_2 \wedge \text{isQ}(q, Q^i_2, nvo_2, E_2, E_2^0) \]

\[ \cdots \]

\[ \forall Q^0_n \text{, isQ}(q, Q^0_n, nvo_n, E_n, E_n^0) \Rightarrow \exists Q^i_n \text{. getQ}(Q^0_n, H_n) = Q^i_n \wedge \text{isQ}(q, Q^i_n, nvo_n, E_n, E_n^0) \]

For all $j \in \{2 \cdots n\}$, let $Q_j^0 = \text{getQ}(Q_{j-1}^0, H_{j-1})$. From above we then have:

$$
\exists Q_1^t, \cdots, Q_n^t. \\
\text{getQ}(Q_1^0, H_1) = Q_1^t \land \text{getQ}(Q_2^0, H_2) = Q_2^t \land \cdots \land \text{getQ}(Q_{n-1}^0, H_{n}) = Q_{n}^t
$$

From its definition we thus know there exists $Q_n^t$ such that $\text{getQ}(Q_1^0, H_1, \cdots, H_{n}) = Q_n^t$. That is, there exists $Q$ such that $\text{getQ}(\epsilon, H) = Q$, as required. 

\[\square\]

**Theorem 8.** For all client programs $P$ of the queue library (comprising calls to \text{enq} and \text{deq} only) and all PTSO-valid executions $E$ of \text{start}(P), E is persistently linearisable.

**Proof.** Pick an arbitrary program $P$ and a PTSO-valid execution $E = G_1; \cdots; G_n$ of $P$. For each $i \in \{1 \cdots n\}$, construct $T_i$ and $H_i$ as above. It then suffices to show that:

\[\forall i \in \{1 \cdots n\}. \forall a, b \in T_i. (a, b) \in \text{hb}_i \implies a \prec_{H_i} b \quad (36)\]

\[\text{fifo}(\epsilon, H) \text{ holds when } H = H_1, \cdots, H_n \quad (37)\]

**TS.** $(36)$

Pick arbitrary $i \in \{1 \cdots n\}, a, b \in T_i$ such that $(a, b) \in \text{hb}_i$. We then know there exist $c, \tau, p, n, e, c', \tau', p', n', e'$ such that $a \in H(c, \tau, p, n, e), b \in H(c', \tau', p', n', e')$ and either:

1) $H(c, \tau, p, n, e) = H(c', \tau', p', n', e'), a = H(c, \tau, p, n, e).\text{inv} \text{ and } b = H(c, \tau, p, n, e).\text{ack}$; or

2) $H(c, \tau, p, n, e) = H(c', \tau', p', n', e'), a = H(c, \tau, p, n, e).\text{ack} \text{ and } b = H(c, \tau, p, n, e).\text{inv}$; or

3) $H(c, \tau, p, n, e) \neq H(c', \tau', p', n', e')$.

In case (1) the desired result holds immediately. In case (2) we have $b \xrightarrow{po_i} a \xrightarrow{hb_i} b$, and since $\text{po}_i \subseteq \text{hb}_i$, we have $b \xrightarrow{hb_i} a \xrightarrow{hb_i} b$. Consequently, from the transitivity of $\text{hb}_i$, we have $(b, b) \in \text{hb}_i$, contradicting the acyclicity of $\text{hb}_i$ in Lemma E.1. In case (3) from Lemma C.1 and the definition of $H_i$ we have $a \prec_{H_i} b$, as required.

**TS.** $(37)$

From Lemma C.3 we know there exists $Q$ such that $\text{getQ}(\epsilon, H) = Q$. From the definition of $\text{fifo}(\epsilon, H)$ we know $\text{fifo}(\epsilon, H)$ holds if and only if there exists $Q$ such that $\text{getQ}(\epsilon, H) = Q$. As such we have $\text{fifo}(\epsilon, H)$, as required. 

\[\square\]

D  NON-BLOCKING MICHAEL-SCOTT QUEUE LIBRARY

As before, for an arbitrary program $P$ and a PTSO-valid execution $E = G_1; \cdots; G_n$ of $P$ with $G_i = (E_i^0, E_i^P, \text{po}, \text{rf}, \text{tso}, \text{nvo})$, observe that when $P$ comprises $k$ threads, the trace of each execution era (via \text{start}() or \text{recover}()) comprises two stages: i) the trace of the \text{setup} stage by the master thread $\tau_0$ performing initialisation or recovery, prior to the call to $\text{run}(P)$; followed (in \text{po} order) by ii) the trace of each of the constituent program threads $\tau_1 \cdots \tau_k$, provided that the execution did not crash during the setup stage.

As before, thanks to the placement of the persistent fence operations (\text{pfence}), for each thread $\tau_j$, we know that the set of persistent events in execution era $i$, namely $E_i^P$, contains roughly a prefix (in \text{po} order) of thread $\tau_j$’s trace. More concretely, for each constituent thread $\tau_j \in \{\tau_1 \cdots \tau_k\} = \text{dom}(P)$, there exist $P_{\tau_j}^1 \cdots P_{\tau_j}^n$ such that:

1) $P[\tau_j] = o_{\tau_j}^1; \cdots; o_{\tau_j}^{p_j}; o_{\tau_j}^{p_j+1}; \cdots o_{\tau_j}^{p_j^2}; \cdots; o_{\tau_j}^{p_j^{p_j-1}+1}; \cdots; o_{\tau_j}^{p_j^n}$, comprising \text{enq} and \text{deq} operations; and

Fig. 8. A non-blocking persistent Michael-Scott queue implementation with persistence code in blue

2) at the beginning of each execution era \( i \in \{1 \cdots n\} \), the program executed by thread \( \tau_j \) (calculated in \( P' \) and subsequently executed by calling \( \text{run}(P') \)) is that of \( \text{sub}(P[\tau_j],P_{j-1}+1) \), where \( P^0_j = -1 \), for all \( j \); and
3) in each execution era \(i \in \{1 \cdots n\}\), the trace \(H(i,j)\) of each constituent thread \(\tau_j \in \text{dom}(P)\) is of the following form:

\[
H(i,j) \triangleq H(o_j^{p_{j-1}+1}, \tau_j, n_j^{p_{j-1}+1}, e_j^{p_{j-1}+1})
\]

\[
\xrightarrow{p_o} \cdots \xrightarrow{p_o} H(o_j^{p_{j-1}+1}, \tau_j, n_j^{p_{j-1}+1}, e_j^{p_{j-1}+1})
\]

\[
\xrightarrow{p_o} H(o_j^{p_{j}+1}, \tau_j, n_j^{p_{j}+1}, e_j^{p_{j}+1})
\]

\[
\xrightarrow{p_o} \cdots \xrightarrow{p_o} H(o_j^{m_{j-1}+1}, \tau_j, n_j^{m_{j-1}+1}, e_j^{m_{j-1}+1})
\]

\[
\xrightarrow{p_o} H'(o_j^{m_{j}}, \tau_j, n_j^{m_{j}}, e_j^{m_{j}})
\]

for some \(m_j^{p_{j-1}+1}, \cdots, n_j^{p_{j-1}+1}, \cdots, e_j^{p_{j}}\).

- The first two lines denote the execution of the \((P_j^{i-1}+1)\)th to \((P_j^{i})\)th library calls of thread \(\tau_j\), with \(H(o, \tau, p, n, e)\) defined shortly. Moreover, before crashing and proceeding to the next era, all volatile events (those in \(PE\)) in \(H(o_j^{p_{j-1}+1}, \cdots) \xrightarrow{p_o} \cdots \xrightarrow{p_o} H(o_j^{p_{j-1}}, \cdots)\) have persisted, and a prefix (in \(p_o\) order) of the volatile events in \(H(o_j^{p_{j}}, \tau_j, n_j^{p}, e_j^{p})\) has persisted. Note that this prefix may be equal to \(H(o_j^{p_{j}}, \tau_j, n_j^{p}, e_j^{p})\), in which case all its events have persisted.

- The next two lines denote the execution of the subsequent library calls of thread \(\tau_j\) where \(m_j^{p} \leq P_n\), with none of their volatile events having persisted.

- The last line denotes the execution of the \((m_j^{p})\)th call of thread \(\tau_j (m_j^{p} \leq P_n)\), during which the program crashed and the execution of era \(i\) ended. As before, the \(H'(o, \tau, p, n, e)\) denotes a (potentially full) prefix of \(H(o, \tau, p, n, e)\).

The trace \(H(o, \tau, p, n, e)\) of each library call is defined as follows:

\[
H(\text{eq}(\cdot), \tau, p, n, h) \triangleq \text{inv=I}(\text{eq}, \cdot) \xrightarrow{p_o} R(p, c) \xrightarrow{p_o} R(\text{tid}, \tau) \xrightarrow{p_o} FE
\]

\[
\xrightarrow{p_o} \text{h=R}(\text{q.head}, h) \xrightarrow{p_o} \text{r=R}(\text{q.data}[h], n)
\]

\[
\xrightarrow{p_o} \text{lin}_{1}=\text{w}(\text{map}[r][p], n) \xrightarrow{p_o} S_{1} \xrightarrow{p_o} \text{PF} \xrightarrow{p_o} S_{2}
\]

\[
\xrightarrow{p_o} \text{ack=A}(\text{eq}, \cdot)
\]

where \(FE\) denotes the sequence of events, attempting but failing to set the \(\text{rem}\) field of the head node, with

\[
S_{1} = \begin{cases} 
0 & \text{if } n = \text{null} \\
R(n, t', \tau') \xrightarrow{p_o} R(n, p, c, p') \xrightarrow{p_o} \text{w}(\text{map}[r'][p'+1], t') & \text{otherwise}
\end{cases}
\]

\[
S_{2} = \begin{cases} 
0 & \text{if } n = \text{null} \\
\text{lin}_{2}=\text{w}(\text{q.head}, h+1) \xrightarrow{p_o} \text{PF} \xrightarrow{p_o} c=\text{w}(\text{map}[r][p]+1, t') \xrightarrow{p_o} \text{PF} & \text{otherwise}
\end{cases}
\]

for some \(\tau', p'\); and

\[
H(\text{enq}(\cdot), \tau, p, n, e) \triangleq \text{inv=I}(\text{enq}, \cdot) \xrightarrow{p_o} R(p, c) \xrightarrow{p_o} R(\text{tid}, \tau)
\]

\[
\xrightarrow{p_o} \text{w}(\text{val}, v) \xrightarrow{p_o} \text{w}(n, \text{tid}, \tau) \xrightarrow{p_o} \text{w}(n, p, c, p)
\]

\[
\xrightarrow{p_o} \text{w}(\text{map}[r][p], n) \xrightarrow{p_o} \text{PF} \xrightarrow{p_o} \text{R}(\text{q.head}, \text{h})
\]

\[
\xrightarrow{p_o} \text{R}(\text{q.data}[h], v_0) \xrightarrow{p_o} A_0 \cdots \text{R}(\text{q.data}[h+s-1], v_{s-1}) \xrightarrow{p_o} A_{s-1}
\]

\[
\xrightarrow{p_o} \text{R}(\text{q.data}[h+s], \text{null}) \xrightarrow{s\text{ times}} \text{lin}_{2}=\text{w}(\text{q.data}[h+s], \text{null}, n)
\]

\[
\xrightarrow{p_o} \text{PF} \xrightarrow{p_o} \text{ack=A}(\text{enq}, \cdot)
\]
We use the respects of node \( q \). Let us write \( E \) for some operation leaves the \( q \) unchanged while increasing \( q \). Similarly, each \( deq \) operation leaves \( q \) unchanged while increasing \( q \).

Let us write \( \tau_j \in dom(P) \) let:

\[
E^{P}_{i,j} = E^{P}_{i} \cap \{ e | \text{tid}(e) = \tau_j \}
\]

\[
E^{P'}_{i,j} = E^{P'}_{i} \cup S_{i,j}
\]

where

\[
S_{i,j} = \left\{ A(i, enq()) | \exists o, p, n, inv, e.
\begin{align*}
&inv = I(i, enq, n) = \max\left(\text{nvo}\right)_{E^{P}_{i,j}} \cap \\
&\land inv \in H(o, \tau_j, p, n, e) \land \forall r'. A(i, enq, r') \notin E^{P}_{i,j} \\
&\land \text{lp}(H(o, \tau_j, p, n, e)) \in E^{P}_{i,j} \\
&\exists o, p, inv, e.
\end{align*}
\right\}
\]

\[
\left\{ A(i, deq()) | \exists o, p, n, inv, e.
\begin{align*}
&inv = I(i, deq, n) = \max\left(\text{nvo}\right)_{E^{P'}_{i,j}} \cap \\
&\land inv \in H(o, \tau_j, p, n, e) \land \forall r'. A(i, deq, r') \notin E^{P'}_{i,j} \\
&\land \text{lp}(H(o, \tau_j, p, n, e)) \in E^{P'}_{i,j} \\
&\land n \neq \text{null} \land \exists o, p, inv, e.
\end{align*}
\right\}
\]

Let \( E'_{i,j} = \bigcup_{\tau_j \in \text{dom}(P)} E'_{i,j} \). From the definition of each \( E'_{i,j} \) and \( E^{P'}_{i,j} \) we then know that \( E^{P'}_{i,j} \subseteq E'_{i,j} \) and \( E'_{i,j} \in \text{comp}(E^{P'}_{i,j}) \). Let \( T_i = \text{trunc}(E'_{i,j}) \).

Let \( C_i \) denote an enumeration of the sequence of linearization points. That is, for all \( H(o, \tau_j, p, n, e) \), \( H(\tau_j, p', n', e') \), if \( \text{lp}(H(o, \tau_j, p, n, e)) \to \text{lp}(H(\tau_j, p', n', e')) \), then \( H(o, \tau_j, p, n, e) <_{C_i} H(\tau_j, p', n', e') \).
When $C_i$ is enumerated as $C_i = H(c_i^1, r_i^1, p_i^1, n_i^1, e_i^1), \ldots, H(c_i^{t_i}, r_i^{t_i}, p_i^{t_i}, n_i^{t_i}, e_i^{t_i})$, let us define:

\[
H_i = H(c_i^1, r_i^1, p_i^1, n_i^1, e_i^1).\text{inv}. \ldots . H(c_i^{t_i}, r_i^{t_i}, p_i^{t_i}, n_i^{t_i}, e_i^{t_i}).\text{ack}
\]

Lemma D.1. Given a PTSO-valid execution $E = G_1; \cdots; G_n$, let for all $i \in \{1 \cdots n\}$, $C_i$ be as defined above. Then, for all $H(o, r, p, n, e), H(o', r', p', n', e')$, $a, b, c, d$, if $a \in H(o, r, p, n, e)$ and $b \in H(o', r', p', n', e')$, $C_i|_c = H(o', r', p', n', e')$ and $(a, b) \in \text{hb}_i$, then either 1) $c = d$ and $(a, b) \in \text{po}_i; \text{or} 2)$ $c < d$.

Proof. The proof of this lemma is analogous to be proof of its counterpart lemma (Lemma C.1) for the blocking MS queue implementation and is omitted here.

Lemma D.2. Given a PTSO-valid execution $E = G_1; \cdots; G_n$, let for all $i \in \{1 \cdots n\}$, $H_i$ be defined as above with $C_i = H(c_i^1, r_i^1, p_i^1, n_i^1, e_i^1), \ldots, H(c_i^{t_i}, r_i^{t_i}, p_i^{t_i}, n_i^{t_i}, e_i^{t_i})$. For all $i \in \{1 \cdots n\}$, and $a, b$, let $Q^0_i = H(c_i^0, r_i^0, p_i^0, n_i^0, e_i^0)$. $Q^1_i = H(c_i^1, r_i^1, p_i^1, n_i^1, e_i^1)$. $Q^2_i = H(c_i^2, r_i^2, p_i^2, n_i^2, e_i^2)$. $Q^0_i = H(c_i^0, r_i^0, p_i^0, n_i^0, e_i^0)$. $Q^1_i = H(c_i^1, r_i^1, p_i^1, n_i^1, e_i^1)$. $Q^2_i = H(c_i^2, r_i^2, p_i^2, n_i^2, e_i^2)$.

For all $G_i = (E_i^0, E_i^1, E_i^2, \text{po}_i, \text{ri}_i, \text{ts}_i, \text{nvo}_i)$, $H_i$, for all $Q^0_i$ and for all $l \in \{0 \cdots t_i\}$, $k = t_i - l$, $E_i^l = E_i^{l-1} \bigcup \{H(c_i^k, r_i^k, p_i^k, n_i^k, e_i^k)\}.E$ and $Q^k_i$.

getQ($Q^0_i, Q^1_i$) = $Q^k_i$ $\land$ isQ(q, $Q^0_i$, $Q^1_i$, nvo, $E_i^0$, $E_i^1$) $\Rightarrow$

$\exists Q^1_i$. getQ($Q^k_i, Q^1_i$) = $Q^k_i$ $\land$ isQ(q, $Q^0_i$, nvo, $E_i^0$, $E_i^1$)

Proof. The proof of this lemma is analogous to be proof of its counterpart lemma (Lemma C.2) for the blocking MS queue implementation and is omitted here.

Corollary 4. Given a PTSO-valid execution $E = G_1; \cdots; G_n$, let for all $i \in \{1 \cdots n\}$, $H_i$ be defined as above. For all $G_i = (E_i^0, E_i^1, E_i^2, \text{po}_i, \text{ri}_i, \text{ts}_i, \text{nvo}_i)$, $H_i$ and for all $Q^0_i$:

isQ(q, $Q^0_i$, nvo, $E_i^0$, $E_i^1$) $\Rightarrow$

$\exists Q^1_i$. getQ($Q^0_i, H_i$) = $Q^1_i$ $\land$ isQ(q, $Q^0_i$, nvo, $E_i^0$, $E_i^1$)

Proof. Follows immediately from the previous lemma when $k = 0$.

Lemma D.3. Given a PTSO-valid execution $E = G_1; \cdots; G_n$, if $H = H_1 \cdots H_n$ with $H_i$ defined as above for all $i \in \{1 \cdots n\}$, then:

$\exists Q_i$. getQ($Q_i, H$) = $Q_i$

Proof. Pick an arbitrary PTSO-valid execution $E = G_1; \cdots; G_n$, with $H = H_1 \cdots H_n$ and $H_i$ defined as above for all $i \in \{1 \cdots n\}$. Let $Q^0_i = \epsilon$. By definition we then have isQ(q, $Q^0_i$, nvo, $E_i^0$, $E_i^1$).

On the other hand from Corollary 4 we have:

$\exists Q^1_i$. getQ($Q^0_i, H_i$) = $Q^1_i$ $\land$ isQ(q, $Q^1_i$, nvo, $E_i^0$, $E_i^1$)

$\forall Q^0_i$. isQ(q, $Q^0_i$, nvo, $E_i^0$, $E_i^1$) $\Rightarrow$

$\exists Q^1_i$. getQ($Q^0_i, H_i$) = $Q^1_i$ $\land$ isQ(q, $Q^1_i$, nvo, $E_i^0$, $E_i^1$)

$\forall Q^1_i$. getQ($Q^1_i, H_i$) = $Q^1_i$ $\land$ isQ(q, $Q^1_i$, nvo, $E_i^0$, $E_i^1$)

$\forall Q^0_i$. isQ(q, $Q^0_i$, nvo, $E_i^0$, $E_i^1$) $\Rightarrow$

$\exists Q^1_i$. getQ($Q^0_i, H_i$) = $Q^1_i$ $\land$ isQ(q, $Q^1_i$, nvo, $E_i^0$, $E_i^1$)

For all $j \in \{2 \cdots n\}$, let $Q^0_j = \text{getQ}(Q^0_{j-1}, H_{j-1})$. From above we then have:

$\exists Q^1_i, \cdots, Q^1_{n}$. $\exists Q^1_i, \cdots, Q^1_{n}$. $\exists Q^1_i, \cdots, Q^1_{n}$. $\exists Q^1_i, \cdots, Q^1_{n}$.

getQ($Q^1_i, H_1$) = $Q^1_i$ $\land$ getQ($Q^1_i, H_2$) = $Q^1_i$ $\land$ $\cdots$ $\land$ getQ($Q^1_n, H_n$) = $Q^1_n$
From its definition we thus know there exists $Q'_n$ such that $\text{getQ}(Q'_0, H_1, \ldots, H_n) = Q'_n$. That is, there exists $Q$ such that $\text{getQ}(\epsilon, H) = Q$, as required. □

**Theorem 9.** For all client programs $P$ of the queue library (comprising calls to $\text{enq}$ and $\text{deq}$ only) and all PTSO-valid executions $\mathcal{E}$ of $\text{start}(P)$, $\mathcal{E}$ is persistently linearisable.

**Proof.** Pick an arbitrary program $P$ and a PTSO-valid execution $\mathcal{E} = G_1; \ldots; G_n$ of $P$. For each $i \in \{1 \cdots n\}$, construct $T_i$ and $H_i$ as above. It then suffices to show that:

\[ \forall i \in \{1 \cdots n\}. \forall a, b \in T_i. (a, b) \in \text{hb}_i \Rightarrow a \prec_{H_i} b \]  

(38)

\[ \text{fifo}(\epsilon, H) \text{ holds when } H = H_1, \ldots, H_n \]  

(39)

**TS.** (38) Pick arbitrary $i \in \{1 \cdots n\}, a, b \in T_i$ such that $(a, b) \in \text{hb}_i$. We then know there exist $c, \tau, p, n, e, c', \tau'$, $p', n', e'$ such that $a \in H(c, \tau, p, n, e), b \in H(c', \tau', p', n', e')$ and either:

1) $H(c, \tau, p, n, e) = H(c', \tau', p', n', e'), a = H(c, \tau, p, n, e).\text{inv}$ and $b = H(c, \tau, p, n, e).\text{ack}$; or

2) $H(c, \tau, p, n, e) = H(c', \tau', p', n', e'), a = H(c, \tau, p, n, e).\text{ack}$ and $b = H(c, \tau, p, n, e).\text{inv}$; or

3) $H(c, \tau, p, n, e) \neq H(c', \tau', p', n', e')$.

In case (1) the desired result holds immediately. In case (2) we have $b \xrightarrow{\text{po}_i} a \xrightarrow{\text{hb}_i} b$, and since $\text{po}_i \subseteq \text{hb}_i$ we have $b \xrightarrow{\text{hb}_i} a \xrightarrow{\text{hb}_i} b$. Consequently, from the transitivity of $\text{hb}_i$ we have $(b, b) \in \text{hb}_i$, contradicting the acyclicity of $\text{hb}_i$ in Lemma E.1. In case (3) from Lemma D.1 and the definition of $H_i$ we have $a \prec_{H_i} b$, as required.

**TS.** (39) From Lemma D.3 we know there exists $Q$ such that $\text{getQ}(\epsilon, H) = Q$. From the definition of $\text{fifo}(\epsilon, H)$ we know $\text{fifo}(\epsilon, H)$ holds if and only if there exists $Q$ such that $\text{getQ}(\epsilon, H) = Q$. As such we have $\text{fifo}(\epsilon, H)$, as required. □
E  AUXILIARY RESULTS

Lemma E.1. For all PTSO-valid execution graphs \( G = (E^0, E^p, E, p_o, r_f, tso, nvo) \), then acyclic(\( h_b \)) holds, where \( h_b = (p_o \cup r_f)^+ \).

Proof. We proceed by contradiction. Let us assume that acyclic(\( h_b \)) does not hold and there exists \( a \) such that \((a, a) \in h_b\). From Lemma E.2 below we then have \((a, a) \in p_o \cup tso\). That is, either: 1) \((a, a) \in p_o\); or 2) \((a, a) \in tso\). However, both cases lead to a contradiction as since \( G \) is valid, we know both \( p_o \) and \( tso \) are strict orders.

\[ \square \]

Lemma E.2. For all PTSO-valid execution graphs \( G = (E^0, E^p, E, p_o, r_f, tso, nvo) \) and for all \( a, b \), if \((a, b) \in h_b = (p_o \cup r_f)^+ \), then \((b, a) \in p_o \cup tso\).

Proof. Pick an arbitrary PTSO-valid execution graph \( G = (E^0, E^p, E, p_o, r_f, tso, nvo) \). Note that \( h_b = (p_o \cup r_f)^+ = (p_o \cup (r_f \setminus p_o))^+ \). Let \( h_b_0 = p_o \cup (r_f \setminus p_o) \) and \( h_b_{i+1} = h_b_i \), for all \( i \in \mathbb{N} \). As \( h_b \) is a transitive closure, it is straightforward to demonstrate that \( h_b = \bigcup_{i \in \mathbb{N}} h_b_i \). We thus show instead that:

\[ \forall i \in \mathbb{N}. \forall a, b. (a, b) \in h_b_i \Rightarrow (a, b) \in p_o \cup tso \]

We proceed by induction on \( i \).

Base case \( i = 0 \)

Pick an arbitrary \( a, b \) such that \((a, b) \in h_b_0 \). There are two cases to consider: either \((a, b) \in p_o\), or \((a, b) \in r_f \setminus p_o\). In the former case the desired result holds immediately. In the latter case, as from the PTSO-validity of \( G \) we know \( r_f \subseteq tso \cup p_o \) and as \((a, b) \in r_f \setminus p_o\), we know that \((a, b) \in tso\), as required.

Inductive case \( i = n + 1 \)

\[ \forall j \in \mathbb{N}. \forall a, b. j \leq n \wedge (a, b) \in h_b_j \Rightarrow (a, b) \in p_o \cup tso \]

(I.H.)

Pick an arbitrary \( a, b \) such that \((a, b) \in h_b_i \). From the definition of \( h_b_i \) we then know there exists \( c \) such that \((a, c) \in p_o \cup (r_f \setminus p_o) \) and \((c, b) \in h_b_n \).

There are two cases to consider: either 1) \((a, c) \in p_o\); or 2) \((a, c) \in r_f \setminus p_o\).

In case (1), let \( h_b_{i-1} = \text{id} \). From the definition of \( h_b_n \) we then know there exists \( d \) such that \((c, d) \in p_o \cup (r_f \setminus p_o) \) and \((d, b) \in h_b_{n-1} \). There are two more cases to consider: i) \((c, d) \in p_o\); or ii) \((c, d) \in r_f \setminus p_o\).

In case (1.i) we have \( a \xrightarrow{p_o} c \xrightarrow{p_o} d \) and thus from the transitivity of \( p_o \) we have \((a, d) \in p_o \subseteq h_b_0 \). As \((d, b) \in h_b_{n-1} \), from the definition of \( h_b_n \) we have \((a, b) \in h_b_n \). Consequently, from (I.H.) we have \((a, b) \in p_o \cup tso\), as required.

In case (1.ii), from the PTSO-validity of \( G \) we know \( r_f \subseteq tso \cup p_o \). Since \((c, d) \in r_f \setminus p_o\), we thus know that \((c, d) \in tso\). On the other hand, from the validity of \( G \) we know \( p_o \setminus (W \times R) \subseteq tso \). Moreover, as \((c, d) \in r_f\), we know that \( c \in W \). As \((a, c) \in p_o \) and \( c \in W \), we thus have \((a, c) \in tso \).

We then have \( a \xrightarrow{tso} c \xrightarrow{tso} d \), and thus from the transitivity of \( tso \) we have \((a, d) \in tso \). There are now to cases to consider: a) \( n = 0 \) and thus \( h_b_{n-1} = \text{id} \); or b) \( n > 0 \).

In case (1.ii.a), as \((d, b) \in h_b_{n-1} = \text{id}\), we have \( d = b \) and thus \((a, b) \in tso\), as required.

In case (1.ii.b), since \((d, b) \in h_b_{n-1} \), from (I.H.) we have \((d, b) \in p_o \cup tso\). On the other hand, from the validity of \( G \) we know \( p_o \setminus (W \times R) \subseteq tso \). Moreover, as \((c, d) \in r_f\), we know that \( d \in R \).

As such, we have \((d, b) \in tso \). We then have \( a \xrightarrow{tso} d \xrightarrow{tso} b \), and thus from the transitivity of \( tso \) we have \((a, b) \in tso \), as required.
In case (2), from the PTSO-validity of $G$ we know $rf \subseteq tso \cup po$. Since $(a, c) \in rf \setminus po$, we thus know that $(a, c) \in tso$. On the other hand, since $(c, b) \in hbn$, from (I.H.) we have $(c, b) \in po \cup tso$.

There are two more cases to consider: i) $(c, b) \in tso$; or ii) $(c, b) \in po$.

In case (2.i) we have $a \rightarrow c \rightarrow b$, and thus from the transitivity of $tso$ we have $(a, b) \in tso$, as required.

In case (2.ii), from the validity of $G$ we know $po \setminus (W \times R) \subseteq tso$. On the other hand, since $(a, c) \in rf$, we know that $c \in R$. As such, we have $(c, b) \in tso$. We thus have $a \rightarrow c \rightarrow b$, and thus from the transitivity of $tso$ we have $(a, b) \in tso$, as required.

\[ \square \]