Correct Compilation of Relaxed Memory
Concurrency
Technical Appendix

SOHAM SUNDAR CHAKRABORTY, MPI-SWS, Germany

This is the technical appendix of the thesis “Correct Compilation of Relaxed Memory Concurrency.” It contains the proofs of the simulation of the promising semantics by WEAKEST, the evaluation of the proposed models on the Java causality testcases, and the proofs of various compilation correctness results.

CONTENTS

- Appendix A contains the proof of simulation of promising semantics by WEAKEST.
- Appendix B contains the event structures for causality test cases.
- Appendix C contains the proof of monotonicity property.
- Appendix D contains the proof of correctness of reorderings.
- Appendix E contains the proof of correctness of eliminations.
- Appendix F establishes the correctness of speculative load introduction in WEAKESTMO-LLVM.
A. Proving Simulation of Promising Semantics by WEAKEST

We restate the definition of simulation relation.

**Definition 6.** Let \( P \) be a program with \( T \) threads, \( \Pi \subseteq T \) be a subset of threads, \( G \) be a \textsc{weakest} event structure, and \( MS = (TS, S, M) \) be a promise machine state. We say that \( G \sim_\Pi MS \) holds iff there exist \( W, S, \) and \( sc \) such that the following conditions hold:

1. \( G \) is consistent according to the \textsc{weakest} model: \( \text{isCons}_{\text{WEAKEST}}(G) \).
2. The local state of each thread in \( MS \) contains the program of the thread along with the sequence of covered events of that thread: \( \forall i. \ TS(i).\sigma = (P(i), \text{labels}(\text{sequence}_{spo}(S_i))) \).
3. Whenever \( W \) maps an event of \( G \) to a message in \( MS \), then the location accessed and the written values match: \( \forall e \in \text{dom}(W). e.\text{loc} = W(e).\text{loc} \land e.\text{wval} = W(e).\text{wval} \).
4. All outstanding promises of threads \( (T \setminus \Pi) \) have corresponding write events in \( G \) that are po-after \( S \): \( \forall i \in (T \setminus \Pi). \forall e \in (S_0 \cup S_i). \ TS(i).P \subseteq \{W(e') | (e, e') \in G.\text{po}\} \).
5. For every location \( x \) and thread \( i \), the thread view of \( x \) in the promise state \( MS \) records the timestamp of the maximal write visible to the covered events of thread \( i \).
   \[
   \forall i, x. \ TS(i).V(x) = \max\{W(e).ts | e \in \text{dom}([W_x]; G.jf; shb; sc; shb; [S_i])\}
   \]
6. The \( S \) events satisfy coherence: \( \text{shb}; \text{seco} \) is irreflexive.
7. The atomicity condition holds for the \( S \) events: \( \text{sfr}; \text{smo} \) is irreflexive.
8. The \( \text{sc} \) fences are appropriately ordered by \( \text{sc}: [F_{\text{sc}}]; (\text{shb} \cup \text{shb}; \text{seco}; \text{shb}); [F_{\text{sc}}] \subseteq \text{sc} \).
9. The behavior of \( MS \) matches that of the \( S \) events: \( \text{Behavior}(MS) = \text{Behavior}(G, W, S) \).

Before proceeding further we introduce certain definition and observations which we use in the proofs.

**Auxiliary Definitions.**

- We define immediate relation: given a relation \( R \) we use \( \text{imm}(R) \) to denote the immediate edges of \( R \), that is, \( \text{imm}(R) \triangleq R \setminus (R; R) \).
- Given the Behavior, \( \text{Behavior}|_x \) denotes the \( \{(x, v)\} \) where \( v \) is the value at location \( x \).
A. Proving Simulation of Promising Semantics by \textsc{weakest}

- We define \textit{swe} the external synchronization relation, that is, \textit{swe} \(\triangleq\) \textit{sw} \(\setminus\) \textit{po}.
- In the following discussion \(\textit{op}_a\) denotes the promise machine state transition operation which results in event \(a\) in the event structure and the promise machine reaches machine state \(\text{MS}_a\).
- \(\mathbb{EW}\) denotes the set of read write events where a write is \(\mathbb{W}\)-mapped to some PS message or a read reads from a \(\mathbb{W}\)-mapped write.

\[
\mathbb{EW} \triangleq \{ e \in \text{G.E} \mid e \in \mathbb{W} \cap \text{dom} (\mathbb{W}) \lor \exists w \in \text{dom} (\mathbb{W}). \text{G.rf}(w, e) \}
\]
- \(\text{ts}(e)\) returns the timestamp of a write or view of a read on the respective locations.

\[
\text{ts}(e) \triangleq \begin{cases} 
\mathbb{W}(e).ts & \text{if } e \in \text{St} \cap \mathbb{EW} \\
\mathbb{W}(w).ts & \text{if } e \in \text{Ld} \cap \mathbb{EW} \text{ and } \text{G.rf}(w, e) 
\end{cases}
\]
- In the promise machine \(\text{cur}, \text{rel}, \text{acq}\) denotes the current, release, acquire thread views similar to Kang et al. [33]. The \textit{cur} view is default.

Additionally, we enlist certain observations regarding the relation between the promise machine and event structure.

**Observations**

Considering the promising semantics and event structure we observe the followings.

1. The \((\text{G.E} \setminus \text{S})\) events correspond to the certificate steps of a promise. The certificate steps do not have any release or fence operations. Hence there is no release or fence event in \((\text{G.E} \setminus \text{S})\). As a result, these events do not have outgoing \(\text{G.sw}\) edges. Hence the source event of an incoming \(\text{G.sw}\) edge is in \(\text{S}\), that is, \(\text{G.sw} \subseteq (\text{S} \times \text{G.E})\). Also for \((\text{G.E} \setminus \text{S})\) events the outgoing \(\text{G.hb}\) edges are only \(\text{G.po}\) edges.

2. If a write event \(w \in (\text{G.E} \setminus \text{S})\) is mapped to some promise message, that is, \(\mathbb{W}(w) \neq \bot\), then \(w\) can have outgoing \(\text{G.rfe}\) and \textit{mo} edges.

Now we state and prove Lemma 6 which use in further proofs.

**Lemma 6.**

\textit{Given a program }\mathbb{P}\textit{, suppose }\text{MS}\textit{ is a promise machine state and }\text{G}\textit{ is an \textsc{weakest} event structure such that }\text{G}\textit{ simulates }\text{MS}; \text{G} \sim \text{MS}. \textit{Then,}

1. if two events \(a, b \in \mathbb{EW}\) on the same memory location are related by \((\text{G.hb}; \text{G.eco}^{\text{strong}})\) relation in \(\text{G}\), then \(\text{ts}(a) \leq \text{ts}(b)\). Moreover, if \(b\) is a write event then \(\text{ts}(a) < \text{ts}(b)\).

2. if two events \(a, b \in \text{S}\) on the same memory location are related by \((\text{shb}; \text{seco}^{\text{strong}})\), then \(\text{ts}(a) \leq \text{ts}(b)\). Moreover, if \(b\) is a write event then \(\text{ts}(a) < \text{ts}(b)\).

3. If \(r\) reads from \(w\) such that \((w, r) \in (\text{G.ew}; \text{G.jf})\) holds then \(w\) and \(r\) are not \textit{hb} related, that is \((w, r) \notin (\text{G.hb} \cup \text{G.hb}^{-1})\).
Whenever \( \text{imm}(\text{spo})(a,b) \) does not hold, \((a,b) \in [G.F_{SC} \cap S] ; \text{shb} \cup \text{shb} ; \text{seco} ; \text{shb} ; [G.F_{SC} \cap S] \) implies \( MS_a.S < MS_b.S \).

**Proof.** We study the component relations of \((G.hb; G.eco^\prime_{\text{strong}})\) and \((\text{shb}; \text{seco^\prime})\).

- **case** \((a,b) \in G.po_x\)

  Let \(a\) and \(b\) be in the \(i^{th}\) thread in the event structure.

  In that case \(ts(a) = MS_a.TS(i).V(x)\) and \(ts(b) = MS_b.TS(i).V(x)\).

  We know that promise machine always extends thread view on each location.

  Hence \(MS_a.TS(i).V(x) \leq MS_b.TS(i).V(x)\).

  As a result, \(ts(op_a) \leq ts(op_b)\).

- **case** \((a,b) \in G.rf\)

  In this case \(op_a\) creates the message \(\langle x : \neg@t \rangle\) and \(op_b\) reads from the same message in the promise machine. As a result, \(ts(a) = ts(b)\).

- **case** \((a,b) \in G.ew\)

  We create \(G.ew\) for the event pairs corresponding to the promise and fulfill operations. In this case \(op_a\), \(op_b\) are promise and fulfill operations respectively. The promise operation append a message and the fulfill operation removes the same message from the message queue. Hence, \(ts(a) = ts(b)\).

- **case** \((a,b) \in G.rf\)

  We know that

  \[ G.jf(a,b) \implies (ts(a) = ts(b)), \]

  \[ G.ew(a,b) \implies (ts(a) = ts(b)), \] and

  \[ G.rf = G.ew^7; G.jf. \]

  As a result, \(G.rf(a,b) \implies (ts(a) = ts(b))\).

- **case** \((a,b) \in G.hb\)

  In this case \((a,b) \in (G.po \cup G.sw)^+\).

  If \(G.po(a,b)\) then \((a,b) \in G.po_x\) and hence \(ts(a) < ts(b)\).

  Otherwise there exists some event \(c\) and \(d\) such that \((a,c) \in G.po \land (c,d) \in G.sw \land G.hb^+(c,b)\).

  Following the promising semantics \(ts(a) \leq MS_c.TS(c.tid).V(x)\).

  Then considering \(c\) and \(d\) access types
A. Proving Simulation of Promising Semantics by WEAKEST

- \( c \in G.F_{\text{rel}} \cap [\text{Rel}] \) and \( d \in G.R \cap [\text{Acq}] \)

In this case there exists some event \( w \in EW \) such that \( G.po(c, w) \), \( w.loc = d.loc, \)
\( w \in G.W_{\text{rlx}}, \) and \( (w, d) \in G.jf^+ \). and \( op_w \) results in message \( m = \langle - : -@- \rangle, R \).

In this case view \( MS_c.TS(a.tid).V(x) \) is included in the message view \( m.R, \)
that is, \( MS_c.TS(a.tid).V(x) \in m.R. \)

Now if \( G.jf(w, d) \) then \( m.R \in MS_d.TS(d.tid).cur \)
and hence \( MS_c.TS(a.tid).V(x) \in m.R \in MS_b.TS(b.tid).cur. \)

Otherwise if \( G.jf(w, u_1) \land G.jf(u_1, u_2) \land \ldots \land G.jf(u_n, d) \) where \( u_1, u_2, \ldots, u_n \in (G.U \cap EW) \) then following the promising semantics

(i) if \( w.loc \neq c.loc \) then the view \( MS_c.TS(a.tid).V(x) \) propagates through the mes-
\( \)sages created by \( u_1, u_2, \ldots, u_n \) and finally reaches \( d, \)
that is, \( MS_c.TS(a.tid).V(x) \in m.R \in MS_d.TS(d.tid).cur \) holds.

(ii) if \( w.loc = c.loc \) then \( G.po_x(c, w) \) and hence \( ts(c) < ts(w) \)
and in consequence \( ts(c) < MS_d.TS(d.tid).V(x). \)

Hence, considering (i) and (ii), \( MS_c.TS(c.tid).V(x) \leq MS_d.TS(d.tid).V(x) \) holds.

- \( c \in G.W \cap [\text{Rel}] \) and \( d \in G.R \cap [\text{Acq}] \)

Similarly to above, the view \( MS_c.TS(c.tid).V(x) \) propagates to \( MS_d.TS(d.tid).cur \) by a read-from or release sequence and in that case
\( MS_c.TS(c.tid).V(x) \leq MS_d.TS(d.tid).V(x). \)

- \( c \in G.F \cap [\text{Rel}] \) and \( d \in G.F \cap [\text{Acq}] \)

In this case there exists some event \( w, r \in EW \) such that
\( G.po(c, w), w \in G.W_{\text{rlx}}, G.po(r, d), r \in G.R_{\text{rlx}}, \) and \( (w, r) \in G.jf^+ \).

Note that since a fence \( d \) is in \( EW \), the \( G.po \)-predecessor \( r \) is also in \( EW. \)

Similar to the earlier case \( MS_c.TS(c.tid).V(x) \) propagates to \( r \)
and gets included in \( MS_r.TS(r.tid).V.acq. \)

Finally \( MS_c.TS(k).V.acq \) is included in \( MS_d.TS(d.tid).cur \)
and in turn \( MS_c.TS(c.tid).V(x) \leq MS_d.TS(d.tid).V(x). \)

As a result, \( ts(a) \leq MS_d.TS(d.tid).V(x) \) and following the \( G.hb \) path \( ts(a) \leq ts(b). \)

In all these \( G.hb \) cases the \( ts(a) \) propagates to \( b. \) If \( b \) is a write event then it extends the view
and updates with a new timestamp. Hence if \( b \) is a write then \( ts(a) < ts(b). \)

Following from this argument, if \( (a, b) \in G.mo_{\text{strong}} \) then \( ts(a) < ts(b) \) holds.

6
(a, b) ∈ G.fr_{strong}.
There exists a write c such that (a, c) ∈ G.rf^{−1} ∧ (c, b) ∈ G.mo_{strong}.
In this case ts(a) = ts(c) and ts(c) < ts(b) holds.
As a result, ts(a) < ts(b) holds.

Thus considering the component relations of (G.hb; G.eco_{strong}^{2})|_{loc} results in \leq\text{-order following the timestamps of the corresponding promise machine. (1)}

We now study the component relations of (shb; seco^{2}).

• (a, b) ∈ shb
Considering the definition, in this case, shb ⊆ G hb (E W × E W).
Hence shb(a, b) implies ts(a) ≤ ts(b) and if b is a write event then ts(a) < ts(b).

• (a, b) ∈ srf.
Considering the definition, in this case, srf ⊆ G.rf (E W × E W). Hence srf(a, b) implies ts(a) = ts(b)

• (a, b) ∈ smo.
We know smo ⊆ mo and hence following the definition of mo, smo(a, b) implies ts(a) < ts(b).

• (a, b) ∈ sfr.
Hence (a, b) ∈ (srf^{−1}; smo). As a result, ts(a) < ts(b).

Thus considering the component relations of (shb; seco^{2})|_{loc} results in \leq\text{-order following the timestamps of the corresponding promise machine. Moreover, when (a, b) ∈ (shb; seco^{2})|_{loc} and b is a write then ts(a) < ts(b). (2)

We now study the relation between w and r when (w, r) ∈ (G.ew; G.jf).
We consider two cases

• case G’.hb(r, w) does not hold as w ord ⊆ REL.

• case G’.hb(w, r).
From (1), in this case G’.hb(r, w) implies ts(r) < ts(w). However, we know, G.rf(w, r) implies ts(r) = ts(w).
Hence a contradiction and G’.hb(r, w) does not hold.

As a result, (w, r) ∉ (G hb ∪ G hb^{−1}). (3)

We have to show that (a, b) ∈ [G.F_{sc} ∩ S]; shb ∪ seco; shb; [G.F_{sc} ∩ S] implies MS_{a}.S ≤ MS_{b}.S.
A. Proving Simulation of Promising Semantics by WEAKEST

When shb(a, b), then either the SC view MS_a.S propagates to MS_b or is overwritten by intermediate greater timestamps on the locations. MS_a.S = MS_b.S holds only when two consecutive SC fences are executed, that is, imm(G,po)(a, b) holds.

Otherwise, similar to (1) we can perform case analysis on the shb path and show that MS_a.S < MS_b.S for at least one location x ∈ Locs.

When (a, b) ∈ (shb; seco; shb) then let there are intermediate event e, d ∈ E\W such that shb(a, c), seco(c, d), and shb(d, b) holds. In this case MS_a.S < MS_p.TS(c.tid).V.

From the similar argument as (2), we can show that the timestamps increase or remain same through seco edges from c to d on location c.loc.

Hence seco(c, d) implies MS_a.TS(c.tid).V < MS_d.TS(d.tid).V and shb(d, b) implies MS_d.TS(d.tid).V ≤ MS_b.S.

As a result, whenever imm(spo)(a, b) does not hold, (a, b) ∈ [G.Fsc ∩ \S]; shb ∪ shb; seco; shb; [G.Fsc ∩ \S] implies MS_a.S < MS_b.S. □

Lemma 7. Given a program P, suppose MS is a promise machine state and G is an WEAKEST event structure such that G simulates MS: G ∼ MS. In this case there is no outgoing external-synchronization from G.E \ S events, that is, dom(G.swe) ⊆ S.

Proof. The simulation construction steps ensure that the conflicting events of S, that is, G.E \ S events are created only as part of PS certificate steps in the respective threads.

In the promising semantics the certificate steps are not visible to any other thread. Similarly in event structure G there is no outgoing rfe edge from G.E \ S events except the event corresponding to the promise. Let that event be ep.

From PS we know that ep,ord ⊆ RLX and certificate steps do not have any release fence. Hence G.Fwrel \ (G.E \ S) = ∅.

Hence there is no outgoing G.swe edge from G.E \ S events and dom(G.swe) ⊆ S holds. □

Next we restate and prove Lemma 1.

Lemma 1. G ∼{i} MS ∧ MS np \ G' \ MS' =⇒ ∃G'. G →P,WEAKEST* G' ∧ G' ∼{i} MS'.

Before going to the proof we restate the proof idea.

Proof Idea. The G' is constructed in two steps.

(1) First, for a non-promise operation np we either append a corresponding event e' to G or we identify an existing corresponding event e' in G. In earlier case G is extended to G' and in later case G' = G.

(2) Next, we check whether TS_i has outstanding promises. If so, then we know that there is a promise-free certificate which fulfills the outstanding promises. In that case, for each non-promise certificate step we extend the event structure following the rules in WEAKEST and at each step the constructed event structure remains consistent.

In this construction G and MS are related by S, \W, and we define S', \W' to relate the G' and MS'. By using the definitions of S', \W' we show that G' ∼{i} MS' holds. We use the results of Lemma 6 to establish the simulation relation.
Proof. We do a case analysis on the operation \( \text{op} \) of the promise machine transition \( MS \xrightarrow{np} MS' \) where \( \text{op} = np \). From the definition of the simulation relation we know \( \forall i. TS(i) . \sigma = \langle P(i), \text{labels}(\text{sequence}_{\text{spo}}(S_i)) \rangle \). Hence we can also make a step from the event structure \( G \) to \( G' \).

**Case STORE** \( \text{St}(o, x, v) \) creating message \( m' \):

In the event structure we extend the event structure \( G \) to \( G' \). We extend the cover set \( S_i \) as well as the relations (\( \text{spo}, \text{srf}, \text{smo} \)) to \( S'_i \) along with the respective relations (\( \text{spo}', \text{srf}', \text{smo}' \)) by including an event \( e' \) where

1. \( \text{dom}(G . \text{po}; \{e'\}) = S_0 \cup S_i \),
2. \( e' \in S'_i \setminus S_i \), and
3. \( \text{labels}(\text{sequence}_{G . \text{po}}(S_i)).(e'.\text{lab}) \in P(i) \).

In this case the promise machine is updated as follows.

\[
\begin{align*}
M' &= M \cup \{m'\}, \quad S' = S, \text{ and} \\
\mathcal{T}S' &= \mathcal{T}S[i \mapsto (\langle P(i), \text{labels}(\text{sequence}_{\text{spo}}(S'_i)), V', \mathcal{T}S(i).P \rangle) \text{ where } V' = \mathcal{T}S(i).V[x \mapsto m'.ts].
\end{align*}
\]

Now we do a case analysis on whether such a store event \( e' \) exists in \( G \) or we append a new event.

**Subcase** \( \#e' \in (G . \text{E}_i \setminus S_i) \). \( \text{dom}(G . \text{po}; \{e'\}) = S_0 \cup S_i \land e'.\text{lab} = \text{St}_o(x, v) \):

We create \( e' \) such that \( e'.\text{lab} = \text{St}_o(x, v) \) and append \( e' \) to event structure \( G \) to create \( G' \).

Then,

- \( G'.E = G . E \cup \{e'\} \)
- \( G'.\text{po} = (G . \text{po} \cup \{(e', e) \mid e \in (S_i \cup S_0)\})^+ \)
- \( G'.\text{jf} = G . \text{jf} \)
- \( G'.\text{ew} = G . \text{ew} \)

Let: \( \mathbb{W}' \triangleq \mathbb{W}[e' \mapsto m'] \).

Based on \( \mathbb{W}' \), we derive following definitions in \( MS' \).

- \( S' \triangleq S \cup \{e'\} \)
- \( \text{mo}' \triangleq \text{mo} \cup \{(a, e') \mid a \in G . \mathbb{W}_\mathbb{Z} \land \mathbb{W}(a) \neq \bot \land \mathbb{W}'(a).ts < \mathbb{W}'(e').ts \} \cup \{(e', a) \mid a \in G . \mathbb{W}_\mathbb{Z} \land \mathbb{W}(a) \neq \bot \land \mathbb{W}'(e').ts < \mathbb{W}'(a).ts \} \)
- \( \text{sc}' \triangleq \text{sc} \)
- \( \text{spo}' \triangleq (\text{spo} \cup \{(e, e') \mid e \in S_0 \cup S'_i\})^+ \)
- \( \text{srf}' \triangleq \text{srf} \)

Now we check whether \( G' \sim (\mathcal{T}S', S', M') \).

1. Condition to show: \( G' \) is consistent in WEAKEST model.
A. Proving Simulation of Promising Semantics by **WEAKEST**

- **(CF)** We know that $G$ satisfies constraint (CF). Considering the definition of $G'.ecf$, the only incoming $hb$ edge is $G'.po$ and there is no outgoing edge from event $e'$. Hence $G'.ecf$ is irreflexive and $G'$ satisfies (CF).

- **(CFJ)** We know that $G$ satisfies constraint (CFJ). We also know that $G.jf = G.jf$ and event $e'$ has no outgoing $G'.hb$ or $G'.jf$ edge. Hence $G'.jf \cap G'.ecf = \emptyset$ and $G'$ satisfies (CFJ).

- **(VISJ)** Constraint (VISJ) is preserved in $G'$ as $G'.jf = G.jf$ and $G$ satisfies constraint (VISJ).

- **(ICF)** We know that $G$ satisfies (ICF). Suppose there exists an event $e_1 \in G$ which is in immediate conflict with $e'$ in $G'$, that is $G'. ~ (e_1, e')$ holds.

Then (1) $\text{dom}(G.po; \{e_1\}) = S_0 \cup S_i$, 
(2) $e_1 \in S'_i \setminus S_i$, and 
(3) $\text{labels}(\text{sequence}_{G.po}(S_i)).(e_1.\text{lab}) \in \mathcal{P}(i)$.

However, from definition of $e'$ we already know that 
(1) $\text{dom}(G.po; \{e'\}) = S_0 \cup S_i$, 
(2) $e' \in S'_i \setminus S_i$, and 
(3) $\text{labels}(\text{sequence}_{G.po}(S_i)).(e'.\text{lab}) \in \mathcal{P}(i)$.

Hence following the determinacy condition we know either $e_1 = e'$ or there exists no such $e_1$. Hence (ICF) is preserved in $G'$.

- **(ICFJ)** Constraint (ICFJ) is preserved in $G'$ as $e' \notin R$ and $G$ satisfies constraint (ICFJ).

- **(COH)** We know $G$ preserves (COH) constraint, that is, $(G.hb; G.eco^{str}_strong)$ is acyclic. The incoming edges to event $e'$ are $G'.po$, $G'.fr^{str}_strong$, $G'.hb$ and there is no outgoing edge concerning $G'.hb$ or $G'.eco^{str}_strong$. As a result, $(G.hb; G'.eco^{str}_strong)$ is acyclic and $G'$ preserves (COH) constraint.

2. Condition to show: The local state of each thread in $MS'$ contains the program of that thread along with the sequence of covered events in $G'$ of that thread.

In this we have to show $\forall j. TS'(j).\sigma = \langle \mathcal{P}(j), \text{labels}(\text{sequence}_{spo}(S'_j)) \rangle$.

We know that the relation holds between $MS$ and $G$.

**case** For $j \neq i$, it is trivial because $TS'(j) = TS(j)$ holds from $MS$ to $MS'$ and $S'_j = S_j$ holds from $G$ to $G'$.

**case** For $j = i$, we know $TS(i).\sigma = \langle \mathcal{P}(i), \text{labels}(\text{sequence}_{spo}(S_i)) \rangle$.

Hence following the definition of $TS(i).\sigma, S'_i, spo'$ we get

$\langle \mathcal{P}(i), \text{labels}(\text{sequence}_{spo}(S'_i)) \rangle$

$= \langle \mathcal{P}(i), \text{labels}(\text{sequence}_{spo}(S_i)).e'.\text{lab} \rangle$
= ⟨P(i), T′S(i).σ · e′.lab⟩
= T′S(i).σ
Hence the condition is preserved between MS′ and G′.

3. Condition to show: Whenever \( \mathbb{W} \) maps an event of G′ to a message in MS′, then the location accessed and the written values match.

We know that the event to message mappings for existing events in G.E and messages M do not change.

\[ ∀e ∈ G'.E. e ≠ e' \implies \mathbb{W}'(e) = \mathbb{W}(e) \]

If \( e = e' \) then \( \mathbb{W}'(e') = m' \) and \( e'.loc = m'.loc = x \) and \( e'.wval = m'.wval = v \).
Hence \( \mathbb{W}' \) preserves the condition.

4. Condition to show: For all outstanding promises of threads (T \( \setminus \{i\} \)), there are corresponding write events in G′ that are po-after S′.

We know that for each thread \( j ≠ i \) the set of promises are preserved from MS to MS′, that is, \( ∀j ≠ i. T′S(j).P = T′S′(j).P \).
We also know that G satisfies this condition.
Hence the condition is preserved in G′.

5. Condition to show: For every location ℓ and thread j, the thread view of ℓ in the promise state MS′ records the timestamp of the maximal write visible to the covered events in G′ of thread j.

Essentially we have to show
\[ ∀j, ℓ. T′S′(j).V(ℓ) = \max\{\mathbb{W}'(e).ts \mid e ∈ \text{dom}(\mathbb{W}_x; G'.j ℓ; shb'^{γ}; seco'^{γ}; shb'^{γ}; [S']_j)\} \]

**case** For \( j ≠ i \) or \( j = i ∧ ℓ ≠ x \), it is trivial because \( T′S(\cdot).V(ℓ) = T′S′(\cdot).V(ℓ) \).

**case** For \( j = i ∧ ℓ = x \), following the promising semantics \( e' ∈ G.W_x. \mathbb{W}'(e') = m', m'.ts \) extends the view on \( x \) in thread \( i \), and hence \( T′S(i).V(x) < T′S′(i).V(x) \).
In this case \( e' ∈ S'_i \) and hence \( e' ∈ \text{dom}(\mathbb{W}_x; G'.j ℓ; shb'^{γ}; seco'^{γ}; shb'^{γ}; [S']_j) \) holds.
As a result,
\[ T′S′(i).V(x) = m'.ts = \max\{\mathbb{W}'(e).ts \mid e ∈ \text{dom}(\mathbb{W}_x; G'.j ℓ; shb'^{γ}; seco'^{γ}; shb'^{γ}; [S']_j)\} \]
Thus the relation holds between MS′ and G′.

6. Condition to show: The S′ events in G′ preserve coherence: \( shb'; seco'^{γ} \) is irreflexive.

We know \( e' ∈ S' \) and let \( a ∈ S' \) such that \( (a, e') ∈ (shb'; seco'^{γ}) \).
Hence following the definitions of \( shb'^{γ}, seco'^{γ} \), and from Lemma 6 (2)
A. Proving Simulation of Promising Semantics by WEAKEST

we know $\text{MS}'_a.\text{TS}'(a.\text{tid}).V(x) < \text{MS}_e.\text{TS}'(e'.\text{tid}).V(x)$ as $e' \in \text{St}$.

As a result, $(\text{shb'}; \text{seco}')$ is irreflexive.

7. Condition to show: The atomicity condition for update operations holds for $S'$ events in $G'$.

We know that $[G'.U \cap S'] = [G.U \cap S]$ and $[G.U \cap S]; (\text{sfr}; \text{smo}) = \emptyset$ holds.

Assume there exists an update $u \in G'.U \cap S'$, which reads from $w$, such that $\text{sfr'}(u, e')$ and $\text{smo'}(e', u)$ holds.

By the definitions of $\text{sfr'}$ and $\text{smo'}$, $\mathbb{W}(w).ts < m'.ts < \mathbb{W}(u).ts$.

But the promising semantics does not assign a timestamp in that range.

Hence a contradiction and $[G'.U \cap S']; (\text{sfr'}; \text{smo'}) = \emptyset$ holds.

8. Condition to show: The sc fences in $G'$ are appropriately ordered by sc'.

We know $[G.\mathcal{F}_{sc}]; \text{shb} \cup \text{shb}; \text{seco}; \text{shb}; [G.\mathcal{F}_{sc}] \subseteq \text{sc}$ holds in $G$.

From definitions we know, $G'.\mathcal{F}_{sc} = G.\mathcal{F}_{sc}, \text{sc'} = \text{sc}, \text{shb} \subseteq \text{shb'}, \text{seco} \subseteq \text{seco'}$.

Consider $a, b$ are two SC fences such that $(a, b) \in [G.\mathcal{F}_{sc}]; \text{shb} \cup \text{shb}; \text{seco}; \text{shb}; [G.\mathcal{F}_{sc}]$, and $\text{sc}(a, b)$ holds.

In that case $(a, b) \in (\text{shb'} \cup \text{shb}; \text{seco'}; \text{shb'})$ holds and $\text{sc'}(a, b)$ holds.

To show $[G'.\mathcal{F}_{sc}]; \text{shb'} \cup \text{shb}; \text{seco}; \text{shb'}; [G'.\mathcal{F}_{sc}] \subseteq \text{sc'}$, we have to show $(b, a) \notin (\text{shb'} \cup \text{shb}; \text{seco'}; \text{shb'})$. We show this by contradiction.

Assume $(b, a) \in (\text{shb'} \cup \text{shb}; \text{seco}; \text{shb'})$.

This is possible due to the relations created to/from event $e'$.

Considering the relations in $\text{shb'}$ and $\text{seco'}$, the incoming relations to event $e'$ are $\text{shb'}$, $\text{sfr'}$, $\text{smo'}$ and the outgoing edges are $\text{smo'}$.

As there is no outgoing $\text{srf}$ edge from $e'$, no new synchronization edge is created, that is, $\text{ssw'} = \text{ssw}$.

Thus a $\text{smo'}(e', w)$ edge where $w$ is a write event occurs in the $(\text{shb'} \cup \text{shb}; \text{seco}; \text{shb'})$ path from $b$ to $a$.

In this case the path from $b$ to $a$ is $(b, e') \in \text{shb'}; \text{seco}'$ and $(e', a) \in \text{smo'}; \text{seco}'; \text{shb'}$.

We analyze the cases of $(b, e') \in \text{shb'}; \text{seco}'$.

- **case** $\text{shb'}(b, e')$.

  In this case $\text{shb}(b, e)$ and $\text{spo'}(e, e')$ hold.

  Hence $\text{MS}_b.\text{TS}(b.\text{tid}).V(x) \leq \text{MS}_e.\text{TS}(e.\text{tid}).V(x) < \text{MS}_e.\text{TS}(e'.\text{tid}).V(x)$.

- **case** $\text{shb'}; \text{seco'}(b, e)$ and $\text{smo'}(c, e')$.

  Hence $\text{shb}; \text{seco}(b, c)$ and $\text{smo'}(c, e')$ holds.

  So $\text{MS}_b.\text{TS}(b.\text{tid}).V(x) \leq \text{MS}_e.\text{TS}(e.\text{tid}).V(x) < \text{MS}_e.\text{TS}(e'.\text{tid}).V(x)$.
Now we analyze \((e', a) \in smo'; seco'; shb'\).

In this case there exist a write \(w \in S\) such that
\[smo'(e', w) \text{ and } (w, a) \in seco'; shb\]
holds.

Hence \(MS_a. TS(e'. tid). V(x) \leq MS_a. TS(w. tid). V(x)\).

As a result, in all cases \(MS_b. TS(b. tid). V(x) \leq MS_a. TS(a. tid). V(x)\) holds.

Hence in the promise machine \(Behavior|_x(G', W', S') = \{(x, v)\}\).

As a result, \([G'. F_{sc}]; shb' \cup shb'; seco'; shb' ; [G'. F_{sc}] \subseteq sc'\) holds.

9. Condition to show: The behavior of \(MS'\) matches that of the \(S'\) events in \(G'\).

Essentially we have to show, \(Behavior(MS') = Behavior(G', W', S')\).

Following the definitions of \(Behavior(MS')\) and \(Behavior(G', W', S')\); we know following cases for a location \(\ell\):

- **case** \(\ell \neq x\):

  The set of messages on \(\ell \neq x\) remains from \(MS\) to \(MS'\).

  Hence in the promise machine \(Behavior|_{\ell} (MS') = Behavior|_{\ell} (MS)\) holds.

  Similarly \(Behavior|_{\ell} (G', W', S') = Behavior|_{\ell} (G, W, S)\) holds in the event structure.

  We already know that \(Behavior|_{\ell} (MS) \subseteq Behavior|_{\ell} (G, W, S)\) holds for \(MS\) and \(G\).

  As a result, \(Behavior|_{\ell} (MS') = Behavior|_{\ell} (G', W', S')\).

- **case** \(\ell = x\):

  Let \(m\) be the message on \(x\) which results in the behavior of \(MS\). In that case
  \(m. loc = x, \maxmsg(M \cup \bigcup_i TS(i). P, x) = m\), and let \(m. wval = v_1\).

  As a result, \((x, v_1) \in Behavior(MS)\). In this case there exists event \(e_1 \in G. W_x \cap S\) such that
  \(W(e_1) = m, e_1. loc = x, e_1. wval = v_1, \text{ and } \notin e_2 S. mo(e_1, e_2)\).

  Considering the new message is \(m'\), we know \(m' = W'(e')\) and \(m'. wval = v\) holds.

  Comparing the \(m\) and \(m'\) we have two subcases:

  - **subcase** \(m. ts < m'. ts\).

    In this case \(\maxmsg(M' \setminus \bigcup_i TS'(i). P, x) = m'\) and \(Behavior|_{x} (MS') = \{(x, v)\}\).

    In the event structure \(G', mo'(e_1, e')\) holds and hence \(Behavior|_{x} (G', W', S') = \{(x, v)\}\).
A. Proving Simulation of Promising Semantics by WEAKEST

– subcase \( m.ts > m'.ts \).

In this case \( \text{maxmsg}(M' \setminus \bigcup_i \mathcal{TS}'(i).P, x) = \text{maxmsg}(M \setminus \bigcup_i \mathcal{TS}(i).P, x) \)
and \( \text{Behavior}|_x (MS') = \text{Behavior}|_x (MS) = \{(x, v_1)\} \).

In the event structure \( mo'(e', e_1) \) holds and hence
\( \text{Behavior}|_x (G', \mathcal{W}', S') = \text{Behavior}|_x (G, \mathcal{W}, S) = \{(x, v_1)\} \).

In both cases \( \text{Behavior}|_x (G', \mathcal{W}', S') = \text{Behavior}|_x (MS') \) holds.

As a result, \( \text{Behavior}(G', \mathcal{W}', S') = \text{Behavior}(MS') \).

Subcase \( \exists e' \in (G.E_i \setminus S_i). \ \text{dom}(G.po; \{\{e'\}\}) = S_0 \cup S_i \land e'.lab = \text{St}_o(x, v) \):
We take \( G' = G \) and let \( \mathcal{W}' \triangleq \mathcal{W}[e' \mapsto m'] \).

Based on \( \mathcal{W}' \), we derive following definitions in \( MS' \).

- \( S' \triangleq S \union \{e'\} \)
- \( mo' \triangleq mo \union \{(a, e') \mid a \in G.W_x \land \mathcal{W}(a) \neq \bot \land \mathcal{W}'(a).ts < \mathcal{W}'(e').ts\} \)
- \( sc' \triangleq sc \)
- \( spo' \triangleq (spo \union \{(e, e') \mid e \in S_0 \cup S'_i\})^+ \)
- \( srf' \triangleq srf \)

Now we check whether \( G' \sim_{(i)} (\mathcal{TS}', S', M') \).

1. Condition to show: \( G' \) is consistent in WEAKEST model.
   \( G' \) is consistent as \( G \) is consistent.

2. Condition to show: The local state of each thread in \( MS' \) contains the program of that thread along with the sequence of covered events in \( G' \) of that thread.

In this we have to show \( \forall j. \mathcal{TS}'(j).\sigma = \langle P(j), \text{labels}(\text{sequence}_{spo'}(S'_j)) \rangle \).

We know that the relation holds between \( MS \) and \( G \).

- Case For \( j \neq i \), it is trivial because \( \mathcal{TS}'(j) = \mathcal{TS}(j) \) holds from \( MS \) to \( MS' \) and \( S'_j = S_j \) holds from \( G \) to \( G' \).

- Case For \( j = i \), we know \( \mathcal{TS}(i).\sigma = \langle P(i), \text{labels}(\text{sequence}_{spo}(S_i)) \rangle \).

Hence following the definition of \( \mathcal{TS}(i).\sigma, S'_i \), \( spo' \) we get
\[ \langle P(i), \text{labels}(\text{sequence}_{spo'}(S'_i)) \rangle \]
\[ = \langle P(i), \text{labels}(\text{sequence}_{spo}(S_i)) \cdot e'.lab \rangle \]
\[ = \langle P(i), \mathcal{TS}(i).\sigma \cdot e'.lab \rangle \]
\[ = \mathcal{TS}'(i).\sigma \]
Hence the condition is preserved between \( MS' \) and \( G' \).

Note. This was same as the other scenario when we append a new \( \text{St}_o(x, v) \).
3. Condition to show: Whenever $W'$ maps an event of $G'$ to a message in $MS'$, then the location accessed and the written values match.

**Case** The event to message mappings for existing events in $G.E$ and messages $M$ do not change. Hence $\forall e \in G'.E. e \neq e' \implies W'(e) = W(e)$.

If $e = e'$ then $W'(e') = \text{wmsg}(\text{op}) = m'$ and $e'.loc = \text{wmsg}(\text{op}).loc = x$ and $e.wval = m'.wval = v$.

Thus $W'$ preserves the condition between $MS'$ and $G'$.

4. Condition to show: For all outstanding promises of threads ($T \setminus \{i\}$), there are corresponding write events in $G'$ that are po-after $S'$.

We know that for each thread $j \neq i$ the set of promises are preserved from MS to MS', that is, $\forall j \neq i. TS(j).P = TS'(j).P$.

We also know that $G$ satisfies this condition.

Hence the condition is preserved in $G'$.

**Note.** This was same as the other scenario when we append a new $\text{St}_o(x, v)$.

5. Condition to show: For every location $\ell$ and thread $j$, the thread view of $\ell$ in the promise state $MS'$ records the timestamp of the maximal write visible to the covered events in $G'$ of thread $j$.

Essentially we have to show

$$\forall j, \ell. TS'(j).V(\ell) = \max\{W'(e).ts \mid e \in \text{dom}([W_x]; G'.j\ell'; \text{shb}'_\ell'; \text{sc}'_{j\ell}; \text{shb}'_{j\ell}; [S'_i])\}$$

For $j \neq i$ or $j = i \land \ell \neq x$, it is trivial because $TS'.V(\ell) = TS.V(\ell)$.

For $j = i \land \ell = x$, from the definition we know

1. $TS(i).V(x) = \max\{W(e).ts \mid e \in \text{dom}([W_x]; G.jf'; \text{shb}'_i'; \text{sc}'_i; \text{shb}'_i; [S_i])\}$
2. $TS'(i).V(x) = m'.ts$
3. $W'(e') = m'.ts$ holds.

Following the promising semantics, we know $TS'(i).V(x)$ extends the thread view of $x$ from $TS(i).V(x)$ and $TS(i).V(x) < m'.ts$.

Hence following the construction,

$$TS'(i).V(x) = m'.ts = \max\{W'(e).ts \mid e \in \text{dom}([W_x]; G'.jf'; \text{shb}'_{j\ell}; \text{sc}'_{j\ell}; \text{shb}'_{j\ell}; [S'_i])\}$$

holds.

Thus the relation holds between $MS'$ and $G'$.

6. Condition to show: The $S'$ events in $G'$ preserve coherence: $\text{shb}'_i$; $\text{seco}'_{j\ell}$ is irreflexive.

The argument is analogous to the case when we append a new $\text{St}_o(x, v)$. 

15
A. Proving Simulation of Promising Semantics by WEAKEST

7. Condition to show: The atomicity condition for update operations holds for $S'$ events in $G'$.

The argument is analogous to the case when we append a new $St_o(x, v)$.

8. Condition to show: The sc fences in $G'$ are appropriately ordered by $sc'$.

The argument is analogous to the case when we append a new $St_o(x, v)$.

9. Condition to show: The behavior of $MS'$ matches that of the $S'$ events in $G'$.

Essentially we have to show, $Behavior(\text{MS'}) = Behavior(G', W', S')$.

Following the definitions of $Behavior(\text{MS'})$ and $Behavior(G', W', S')$; we know following cases for a location $\ell$:

- **case $\ell \neq x$:**
  The set of messages on $\ell \neq x$ remains from $\text{MS}$ to $\text{MS'}$.

  Hence in the promise machine $Behavior|_{\ell}(\text{MS'}) = Behavior|_{\ell}(\text{MS})$ holds.

  Similarly $Behavior|_{\ell}(G', W', S') = Behavior|_{\ell}(G, W, S)$ holds in the event structure.

  We already know that $Behavior|_{\ell}(\text{MS}) = Behavior|_{\ell}(G, W, S)$ holds for $\text{MS}$ and $G$.

  As a result, $Behavior|_{\ell}(\text{MS'}) = Behavior|_{\ell}(G', W', S')$.

- **case $\ell = x$:**

  Let $m$ be the message on $x$ which results in the behavior of $\text{MS}$. In that case $m.loc = x$, $\text{maxmsg}(M \cup \ell \text{TS}(i).P, x) = m$, and let $m.wval = v$. As a result, $(x, v_1) \in Behavior(\text{MS})$.

  In this case there exists event $e_1 \in G.W_x \cap S$ such that $W(e_1) = m$, $e_1.loc = x$, $e_1.wval = v_1$, and $\exists e_2 \in S$. $\text{mo}(e_1, e_2)$.

  Considering the new message is $m'$, we know $m' = W'(e')$ and $m'.wval = v$ holds.

  Comparing the $m$ and $m'$ we have two subcases:

  - **subcase $m.ts < m'.ts$.**
    In this case $\text{maxmsg}(M' \setminus \bigcup_i \ell \text{TS}(i).P, x) = m'$ and $Behavior|_{x}(\text{MS'}) = \{(x, v)\}$.

    In the event structure $G'$, $\text{mo}'(e_1, e')$ holds and hence $Behavior|_{x}(G', W', S') = \{(x, v)\}$.

  - **subcase $m.ts > m'.ts$.**
    In this case $\text{maxmsg}(M' \setminus \bigcup_i \ell \text{TS}(i).P, x) = \text{maxmsg}(M \setminus \bigcup_i \ell \text{TS}(i).P, x)$ and $Behavior|_{x}(\text{MS'}) = Behavior|_{x}(\text{MS}) = \{(x, v_1)\}$.

    In the event structure $\text{mo}'(e', e_1)$ holds and hence $Behavior|_{x}(G', W', S') = Behavior|_{x}(G, W, S) = \{(x, v_1)\}$.
In both cases \( \text{Behavior}_{|x}(G', W', S') = \text{Behavior}_{|x}(MS') \) holds.

As a result, \( \text{Behavior}(G', W', S') = \text{Behavior}(MS') \).

Note. This was same as the other scenario when we append a new \( St_o(x, v) \).

Case **READ** \( Ld(o, x, v) \) reading from message \( wm = \langle x : v@(-, t), R \rangle \):

In the event structure we extend the event structure \( G \) to \( G' \). We extend the cover set \( S_i \) as well as the relations (\( spo, srf, smo \)) to \( S'_i \) along with the respective relations (\( spo', srf', smo' \)) by including an event \( e' \) where

1. \( \text{dom}(G.po; \{\{e'\}\}) = S_0 \cup S_i \),
2. \( e' \in S_i \setminus S_i \), and
3. \( \text{labels}(\text{sequence}_{G.po}(S_i)).(e'.\text{lab}) \in \mathbb{P}(i) \).

In this case the promise machine is updated as follows.

\[
M' = M, \quad S' = S, \quad \text{and } TS' = TS[i \mapsto ((\mathbb{P}(i), \text{labels}(\text{sequence}_{spo'}(S'_i))), V', TS(i).P)]
\]

where \( V' = TS(i).V[x \mapsto \text{wm}.ts] \).

Now we do a case analysis on whether such an load event \( e' \) exists in \( G \) or we append a new event.

Subcase \( \#e' \in (G.E \setminus S_i). \text{dom}(G.po; \{\{e'\}\}) = S_0 \cup S_i \land e'.\text{lab} = Ld_o(x, v) \land G.jf(w_m, e') \)

where \( wm = \mathbb{W}(w_m) \):

We create \( e' \) such that \( e'.\text{lab} = Ld_o(x, v) \) and append \( e' \) to event structure \( G \) to create \( G' \). In that case

- \( G'.E = G.E \cup \{e'\} \)
- \( G'.po = (G.po \cup \{(e, e') \mid e \in (S_i \cup S_0)\})^+ \)
- \( G'.jf = G.jf \cup \{(w_m, e') \mid \mathbb{W}(w_m) = \text{wm} \land [S_0 \cup S'_i]; G'.po^2; \{\{w_m\}\}] \)
- \( G'.ew = G.ew \)

Let: \( \mathbb{W}' \triangleq \mathbb{W} \).

Based on \( \mathbb{W}' \), we derive following definitions in \( MS' \).

- \( S' \triangleq S \cup \{e'\} \)
- \( mo' \triangleq mo \)
- \( sc' \triangleq sc \)
- \( spo' \triangleq (spo \cup \{(e, e') \mid e \in (S_0 \cup S'_i)\})^+ \)
- \( srf' \triangleq srf \cup \{(w, e') \mid G'.rf(w, e') \land w \in S\} \)

Now we check whether \( G' \sim_{\{i\}} (TS', S', M') \).

1. Condition to show: \( G' \text{ is consistent in WEAKEST model.} \)
A. Proving Simulation of Promising Semantics by WEAKEST

- (CF) We know \( G \) preserves (CF). Hence in \( G' \) we need to only consider the \( e' \).
  
  Assume there exists event \( e_1 \) and \( e_2 \) such that
  
  \[
  G'.\text{hb}(e_1, e'), G'.\text{cf}(e_1, e_2), G'.\text{hb}(e_2, e') \text{ hold.}
  \]

  assert: \( e_1 \in S \).

  We know \( G'.\text{hb}(e_1, e') \).

  Hence either \( G'.\text{po}(e_1, e') \) or \( (e_1, e') \in G'.\text{po} \setminus G'.\text{swe} \cap G'.\text{hb} \).

  case \( G'.\text{po}(e_1, e') \). From the definitions \( e_1 \in S \).

  case \( (e_1, e') \in G'.\text{po} \setminus G'.\text{swe} \cap G'.\text{hb} \).

  Assume \( e_1 \notin S \) and hence \( e_1 \in G.E \setminus S \).

  All po-following events of \( e_1 \) are in \( G.E \setminus S \), that is, \( \text{codom}([\{e_1\}].G.\text{po}) \in G.E \setminus S \).

  However, from Lemma 7 we know that \( \text{dom}(G.\text{swe}) \subseteq S \) and the events in \( G.E \setminus S \) has no outgoing swe edge, that is, \( \text{dom}(G.\text{swe}) \notin (G.E \setminus S) \).

  Hence a contradiction and \( e_1 \in S \).

  assert: \( e_2 \notin S \).

  Assume \( e_2 \in S \).

  From the definition of \( S \) it is conflict-free, that is, \( S \cap G.\text{cf} = \emptyset \). Thus it is not possible and hence a contradiction.

  As a result, \( e_2 \notin S \).

  Now we know that \( G'.\text{hb}(e_2, e') \) hold and thus \( (e_2, e') \in G'.\text{po} \setminus G'.\text{swe} \cap G'.\text{hb} \).

  From Lemma 7 we know that \( e_2 \) has no \( G'.\text{po} \) following event with outgoing \( G'.\text{swe} \). Hence \( G.\text{po}(e_2, e') \) holds.

  In that case \( G'.\text{po}(e_1, e'), G'.\text{po}(e_2, e') \), \( G'.\text{cf}(e_1, e_2) \) result in a contradiction.

  As a result, \( G \) satisfies (CF).

- (CFJ) We know \( G \) preserves (CFJ). Hence in \( G' \) we need to only consider the \( G'.\text{jf}(w_m, e') \).

  Assume there exists event \( e_1 \) and \( e_2 \) such that
  
  \[
  G'.\text{hb}(e_1, e'), G'.\text{cf}(e_1, e_2), G'.\text{hb}(e_2, w_m) \text{ hold.}
  \]

  assert: \( e_1 \in S \).

  We know \( G'.\text{hb}(e_1, e') \).

  Hence either \( G'.\text{po}(e_1, e') \) or \( (e_1, e') \in G'.\text{po} \setminus G'.\text{swe} \cap G'.\text{hb} \).

  case \( G'.\text{po}(e_1, e') \). From the definitions \( e_1 \in S \).

  case \( (e_1, e') \in G'.\text{po} \setminus G'.\text{swe} \cap G'.\text{hb} \).

  Assume \( e_1 \notin S \) and hence \( e_1 \in G.E \setminus S \).
In that case all po following events are in $G.E \setminus S$, that is, $\text{codom}([\{e_1\}].G.po) \in G.E \setminus S$.

However, from Lemma 7 we know that $\text{dom}(G.swe) \subseteq S$ and the events in $G.E \setminus S$ has no outgoing swe edge, that is, $\text{dom}(G.swe) \notin (G.E \setminus S)$.

Hence a contradiction and $e_1 \in S$.

assert: $e_2 \notin S$.

Assume $e_2 \in S$.

From the definition of $S$ it is conflict-free, that is, $S \cap G.cf = \emptyset$. Thus it is not possible and hence a contradiction.

As a result, $e_2 \notin S$.

Now we know that $G'.hb(e_2, w_m)$ as well as $G.hb(e_2, w_m)$ hold and thus $(e_2, w_m) \in G'.po' ; G'.swe; G'.hb^2$.

From Lemma 7 we know that $e_2$ has no $G'.po$ following event with outgoing $G'.swe$. Hence $G.po(e_2, w_m)$ holds.

As a result, $e_1.tid = e_2.tid = w_m.tid$ holds.

However, from the definition of $G'.jf(w_m, e')$ we know that $G'.po(e_1, w_m)$ holds.

In that case $G'.po(e_1, w_m), G'.po(e_2, w_m), G'.cf(e_1, e_2)$ result in a contradiction.

As a result, $G$ satisfies (CFJ).

• (VISJ) We study the possible cases of $w_m$.

  – If $G'.po(w_m, e')$ then the condition holds as $(w_m, e') \notin G'.jfe$.

  – We will show that $G'$ satisfies (CFJ) constraint. Hence $w_m$ cannot be in conflict with $e'$, that is, $(w_m, e') \notin G'.cf$.

  – $w_m$ is in different thread and $G'.jfe(w_m, e')$ holds. We know that $G \sim_{\{i\}} MS$ and the simulation rules ensures that there is no invisible event in the $(T \setminus \{i\})$ threads. Hence $w_m$ is a visible event in $G$ as well as in $G'$.

Considering the above mentioned cases $G'.jfe(w_m, e') \implies w_m \in \text{vis}(G')$ holds and $G$ satisfies (VISJ) constraint.

• (ICF). We know $G$ satisfies constraint (ICF). Following the construction $e' \in G'.R$ and following the determinacy condition if $G'. \sim (e_1, e')$ then $e_1 \in \text{ld}$.

Thus $(e_1, e') \in (G'.R \times G'.R)$ and hence $G'$ satisfies (ICF).

• (ICFJ) From the construction we know there exists no $e_1$ such that $\text{imm}(cf)(e_1, e')$ and $G'.rf(\overline{w^{-1}}(w_m), e_1)$. Moreover, $G$ satisfies constraint (ICFJ). As a result, $G'$ satisfies (ICFJ).

• (COH) We know that $G$ satisfies (COH) constraint and hence $(G.hb; G.eco^2_{strong})$ is acyclic. We check if $(G'.hb; G'.eco^2_{strong})$ is acyclic.

The incoming edges to event $e'$ are $G'.hb$, $G'.rf$ and there is outgoing $G'.fr_{strong}$ edges.
A. Proving Simulation of Promising Semantics by WEAKEST

If \((G'.hb; G'.eco_{strong}^2)\) forms a cycle then

(i) event \(e'\) is in the cycle.

(ii) \(G'.fr_{strong}(e', w')\) is in the cycle where \(w'\) is some write on \(x\).

(iii) Either \(G'.rf(-, e')\) or \(G'.hb(-, e')\)

incoming edge is part of the \((G'.hb; G'.eco_{strong}^2)\) cycle.

Analyzing the cases on incoming edges to event \(e'\) the \((G'.hb; G'.eco_{strong}^2)\) cycle can be as follows.

– case \(G'.rf(-, e')\) completes the the \((G'.hb; G'.eco_{strong}^2)\) cycle.

The \(G'.rf(-, e')\) is either \(G'.jf(w_m, e')\) or there exists \(w_1\) such that
\(G'.ew(w_m, w_1)\) and \((w_1, e') \in (G'.ew; G'.jf)\).

Thus the cycle can be one of the followings ways.

(1) \(G'.rf(w_m, e'), G'.fr_{strong}(e', w'), \text{ and } (w', w_m) \in (G'.hb; G'.eco_{strong}^2)\).

(2) \(G'.rf(w_1, e'), G'.fr_{strong}(e', w'), \text{ and } (w', w_1) \in (G'.hb; G'.eco_{strong}^2)\).

Also note that \(G'.fr_{strong}(e', w')\) implies

either \(G.mo_{strong}(w_m, w')\) or \(G.mo_{strong}(w_1, w')\) already hold in \(G\).

Considering (1), (2), and possible reasons for \(G'.fr_{strong}(e', w')\), we consider following subcases.

* subcase

(i) \(G'.rf(w_m, e'), G'.fr_{strong}(e', w'), \text{ and } (w', w_m) \in (G'.hb; G'.eco_{strong}^2)\)

is the cycle, and \(G.mo_{strong}(w_m, w')\)

(ii) \(G'.rf(w_1, e'), G'.fr_{strong}(e', w'), \text{ and } (w', w_1) \in (G'.hb; G'.eco_{strong}^2)\)

is the cycle, and \(G.mo_{strong}(w_1, w')\)

In case (i) \((w', w_m) \in (G'.hb; G'.eco_{strong}^2)\) implies
\((w', w_m) \in (G hb; G.eco_{strong}^2)\) holds in \(G\).

In that case \((w', w_m) \in (G hb; G.eco_{strong}^2)\) and \(G.mo_{strong}(w_m, w')\)

form a \((G hb; G.eco_{strong}^2)\) cycle in \(G\).

This is not possible as \((G hb; G.eco_{strong}^2)\) is acyclic and hence a contradiction.

Thus \((G hb; G.eco_{strong}^2)\) is acyclic in this case.

Following the similar argument \((G hb; G.eco_{strong}^2)\) is acyclic in case (ii).

* subcase

(i) \(G'.rf(w_m, e'), G'.fr_{strong}(e', w'), \text{ and } (w', w_m) \in (G'.hb; G'.eco_{strong}^2)\)

is the cycle, and \(G.mo_{strong}(w_1, w')\)

(ii) \(G'.rf(w_1, e'), G'.fr_{strong}(e', w'), \text{ and } (w', w_1) \in (G'.hb; G'.eco_{strong}^2)\)

is the cycle, and \(G.mo_{strong}(w_m, w')\)
In case (i) following Lemma 6,

(a) \((w', w_m) \in (G'.hb; G'.eco^2_{strong})\) implies \\
\((w', w_m) \in (G.hb; G.eco^2_{strong})\) and in turn \(ts(w') < ts(w_m)\),

(b) \(G.ew(w_m, w_1)\) implies \(ts(w_m) = ts(w_1)\), and

(c) \(G.mo_{strong}(w_1, w')\) implies \(ts(w_1) < ts(w')\).

The combination of (a), (b), (c) contradicts the total order of timestamps. Thus \((G'.hb; G'.eco^2_{strong})\) is acyclic in this case.

Following the similar argument \((G'.hb; G'.eco^2_{strong})\) is acyclic in case (ii).

- case \(G'.hb(-, e')\) completes the \((G'.hb; G'.eco^2_{strong})\) cycle.

In this case \(G'.rf(-, e')\) is not part of the \((G'.hb; G'.eco^2_{strong})\) cycle.

Hence \((w', e') \in (G'.hb; G'.eco^2_{strong})\) and \(G'.fr_{strong}(e', w')\) form the \((G'.hb; G'.eco^2_{strong})\) cycle.

\(G'.fr_{strong}(e', w')\) suggests two possibilities:

* **subcase** \(G'.hb(w_m, w')\).

  Following Lemma 6,

  (a) \(ts(w_m) < ts(w')\).

  (b) From \((w', e') \in (G'.hb; G'.eco^2_{strong})\) we know \(ts(w') < ts(e')\).

  (c) We also know \(G'.jf(w_m, e')\) implies \(ts(w_m) = ts(e')\).

  (d) However, \(G'.fr_{strong}(e', w')\) implies \(ts(e') < ts(w')\).

  The combination of (a), (b), (c), (d) contradicts the total order of timestamps and hence \((G'.hb; G'.eco^2_{strong})\) is acyclic in this case.

* **subcase** \(G'.hb(w_1, w')\).

  Following Lemma 6,

  (a) \(ts(w_1) < ts(w')\).

  (b) From \((w', e') \in (G'.hb; G'.eco^2_{strong})\) we know \(ts(w') < ts(e')\).

  (c) We also know \(G'.rf(w_1, e')\) implies \(ts(w_1) = ts(e')\).

  (d) However, \(G'.fr_{strong}(e', w')\) implies \(ts(e') < ts(w')\).

  The combination of (a), (b), (c), (d) contradicts the total order of timestamps and hence \((G'.hb; G'.eco^2_{strong})\) is acyclic in this case.

As a result, \(G'\) satisfies (COH).

Thus \(G'\) is consistent in **WEAK**est model.
A. Proving Simulation of Promising Semantics by WEAKEST

2. Condition to show: The local state of each thread in MS’ contains the program of that thread along with the sequence of covered events in G’ of that thread.

In this we have to show \( \forall j. TS’(j).\sigma = \langle P(j), labels(sequence_{spo}(S’_j)) \rangle \).

We know that the relation holds between MS and G.

For \( j \neq i \), it is trivial because \( TS’(j) = TS(j) \) holds from MS to MS’ and \( S’_j = S_j \) holds from G to G’.

For \( j = i \), we know \( TS(i).\sigma = \langle P(i), labels(sequence_{spo}(S_i)) \rangle \).

Hence following the definition of \( TS’(i).\sigma, S’_i, spo’ \) we get

\[
\langle P(i), labels(sequence_{spo}(S’_i)) \rangle = \langle P(i), labels(sequence_{spo}(S_i)).e’lab \rangle = \langle P(i), TS(i).\sigma.e’lab \rangle = TS’(i).\sigma
\]

Hence the condition is preserved between MS’ and G’.

Note. This was same as the other scenario when we append a new St_o(x, v).

3. Condition to show: Whenever \( W’ \) maps an event of G’ to a message in MS’, then the location accessed and the written values match.

We know \( M’ = M \) and \( W(e') = \bot \). Hence, if \( e \neq e’ \) then \( W’(e) = W(e) \). If \( e = e’ \) then \( W(e’) = \bot \) and the assertion holds.

4. Condition to show: For all outstanding promises of threads \( T \setminus \{i\} \), there are corresponding write events in G’ that are po-after S’.

We know that for each thread \( j \neq i \) the set of promises are preserved from MS to MS’, that is, \( \forall j \neq i. TS(j).P = TS’(j).P \).

We also know that G satisfies this condition.

Hence the condition is preserved in G’.

Note. This was same as the other scenario when we append a new St_o(x, v).

5. Condition to show: For every location \( \ell \) and thread \( j \), the thread view of \( \ell \) in the promise state MS’ records the timestamp of the maximal write visible to the covered events in G’ of thread \( j \).

Essentially we have to show

\[
\forall j, \ell. TS’(j).V(\ell) = \max \{ W'(e).ts \mid e \in \text{dom}([W_x]; G’; jf^j; \text{shb}^j; \text{sc}^j; \text{shb}’^j; [S’_j]) \}
\]

For \( j \neq i \) or \( j = i \land \ell \neq x \), it is trivial because \( TS’(j).V(\ell) = TS.V(\ell) \).

For \( j = i \land \ell = x \), we have to show

\[
TS’(i).V(x) = \max \{ W'(e).ts \mid e \in \text{dom}([W_x]; G’; jf^j; \text{shb}^j; \text{sc}^j; \text{shb}’^j; [S’_j]) \}.
\]
From the definitions we know
(1) $TS(i).V(x) = \max\{W(e).ts | e \in \text{dom}(\{W_x\}; G.jf'; shb'; sc'; shb'; [S_i])\}$
(2) $TS'(i).V(x) = ts(e') = \text{wm}.ts$.

Following the promising semantics, we know $TS'(i).V(x)$ extends the thread view of $x$ from $TS(i).V(x)$ by reading from $\text{wm}$, and $TS(i).V(x) \leq \text{wm}.ts$.

As a result, $TS'(i).V(x) = \text{wm}.ts = \max\{W'(e).ts | e \in \text{dom}(\{W_x\}; G'.jf'; shb'^'; sc'^'; shb'^'; [S'_i])\}$.

Thus the condition is preserved between $MS'$ and $G'$.

6. Condition to show: The $S'$ events in $G'$ preserve coherence: $shb'; seco'^G$ is irreflexive.

We know $shb; seco'^G$ is irreflexive in $G$.

Let event $a \in S'$ and assume $(a, e') \in (shb'; seco'^G)$ and $(e', a) \in (shb'; seco'^G)$.

Following the definitions of $shb'$, $seco'$, and from Lemma 6 (2) we know $MS'_a.TS'(a.tid).V(x) \leq MS_e.TS'(e'.tid).V(x)$.

However, the only outgoing edge from $e'$ is $fr'$ and from the definition we know $sfr'(e', b)$ implies that $MS'_a.TS'(e'.tid).V(x) \leq MS_e.TS'(e'.tid).V(x)$.

Hence a contradiction and $shb'; seco'^G$ is irreflexive.

7. Condition to show: The atomicity condition for update operations holds for $S'$ events in $G'$.

We know that $[G'.U \cap S'] = [G.U \cap S]$ and $[G.U \cap S]; (sfr; smo) = \emptyset$ holds.

The $e'$ does not introduce any $[G.U]; G'.sfr'$ or $[G.U]; G'.smo'$ edge.

As a result, $[G'.U \cap S']; (sfr'; smo') = \emptyset$ holds.

8. Condition to show: The $SC$ fences in $G'$ are appropriately ordered by $sc'^G$.

We know $[G.F_{sc}]; shb \cup shb; seco; shb; [G.F_{sc}] \subseteq sc$ holds in $G$.

From definitions we know, $G'.F_{sc} = G.F_{sc}, sc' = sc, shb \subseteq shb', seco \subseteq seco'$.

Consider $a, b$ are two $SC$ fences such that
$(a, b) \in [G.F_{sc}]; shb \cup shb; seco; shb; [G.F_{sc}], \text{ and } sc(a, b)$ holds.

In that case $(a, b) \in (shb' \cup shb'; seco'; shb')$ holds and $sc'(a, b)$ holds.

To show $[G'.F_{sc}]; shb' \cup shb'; seco'; shb'; [G'.F_{sc}] \subseteq sc'$,
we have to show $(a, b) \notin (shb' \cup shb'; seco'; shb')$.

We show that by contradiction. Assume $(b, a) \in (shb' \cup shb'; seco'; shb')$.

This is possible due to the relations created to/from event $e'$. 

23
A. Proving Simulation of Promising Semantics by WEAKEST

Considering the relations in \(shb'\) and \(seco'\), the incoming relations to event \(e'\) are \(shb'\) and \(sfr'\), and the outgoing edges are \(sfr'\).

Thus a \(sfr'(e', w)\) edge where \(w\) is a write event occurs in the \((shb' \cup shb' ; seco'; shb')\) path from \(b\) to \(a\).

In this case the path from \(b\) to \(a\) is \((b, e') \in shb'; sfr'^2\) and \((e', a) \in sfr'; seco'; shb'\).

It implies \((b, e') \in shb; sfr'^r\) and \((e', a) \in sfr'; seco'; shb\).

In this case there exists \(w, w' \in G'.W \cap S\) such that \(srf'(w, e')\) and \(sfr'(e', w')\) holds.

However, from the definitions, in this case \(smo(w, w')\) already holds and hence \((b, a) \in (shb \cup shb; seco; shb)\) already holds.

This is a contradiction and hence \([G'.F_{sc}]; shb' \cup shb' ; seco'; shb'; [G'.F_{sc}] \subseteq sc'\) holds.

9. Condition to show: The behavior of \(MS'\) matches that of the \(S'\) events in \(G'\).

Essentially we have to show, \(\text{Behavior}(MS') = \text{Behavior}(G', W', S')\).

We know \(\text{Behavior}(MS) = \text{Behavior}(G, W, S)\) holds.

From the definition we know,

\(\text{Behavior}(MS') = \text{Behavior}(MS)\) and \(\text{Behavior}(G', W', S') = \text{Behavior}(G, W, S)\) hold.

As a result, \(\text{Behavior}(MS') = \text{Behavior}(G', W', S')\) holds.

Subcase \(\exists e' \in (G.E_i \setminus S_i). \text{dom}(G.po; \{\{e'\}\}) = S_0 \cup S_i \land e'.\text{lab} = Ld_0(x, v) \land G.jf(w_m, e')\)

where \(wm = W(w_m)\):

We take \(G' = G\) and let \(W' = W\).

Based on \(W'\), we derive following definitions in \(MS'\).

- \(S' \triangleq S \cup \{e'\}\)
- \(mo' \triangleq mo\)
- \(sc' \triangleq sc\)
- \(spo' \triangleq (spo \cup \{(e, e') \mid e \in S_0 \cup S'_i\})^+\)
- \(srf' \triangleq srf \cup \{(w, e') \mid G'.rf(w, e') \land w \in S\}\)

Now we check whether \(G' \sim_1 (TS', S', M')\).

1. Condition to show: \(G'\) is consistent in WEAKEST model.

We know \(G'.E = G.E, G'.po = G.po, G'.jf = G.jf\), and \(G\) is consistent. Hence \(G'\) is also consistent.
2. Condition to show: The local state of each thread in \( MS' \) contains the program of that thread along with the sequence of covered events in \( G' \) of that thread.

In this we have to show \( \forall j. TS'(j).\sigma = \langle P(j), \text{labels}(\text{sequence}_{spo}(S'_j)) \rangle \).

We know that the relation holds between \( MS \) and \( G \).

For \( j \neq i \), it is trivial because \( TS'(j) = TS(j) \) holds from \( MS \) to \( MS' \) and \( S'_j = S_j \) holds from \( G \) to \( G' \).

For \( j = i \), we know \( TS(i).\sigma = \langle P(i), \text{labels}(\text{sequence}_{spo}(S_i)) \rangle \).

Hence following the definition of \( TS(i).\sigma, S'_i, \text{spo}' \) we get

\[
\langle P(i), \text{labels}(\text{sequence}_{spo}(S'_i)) \rangle \\
= \langle P(i), \text{labels}(\text{sequence}_{spo}(S_i)) \cdot e'.\text{lab} \rangle \\
= \langle P(i), TS(i).\sigma \cdot e'.\text{lab} \rangle \\
= TS'(i).\sigma
\]

Hence the condition is preserved between \( MS' \) and \( G' \).

Note. This was same as the other scenario when we append a new \text{St}_o(x, v) or \text{Ld}_o(x, v).

3. Condition to show: Whenever \( \forall \) maps an event of \( G' \) to a message in \( MS' \), then the location accessed and the written values match.

We know \( M' = M \) and \( \forall(e') = \bot \). Hence, if \( e \neq e' \) then \( \forall'(e) = \forall(e) \). If \( e = e' \) then \( \forall(e') = \bot \) and the assertion holds.

Note. This was same as the the scenario when we append a new \text{Ld}_o(x, v).

4. Condition to show: For all outstanding promises of threads \( (T \setminus \{i\}) \), there are corresponding write events in \( G' \) that are po-after \( S' \).

We know that for each thread \( j \neq i \) the set of promises are preserved from \( MS \) to \( MS' \), that is, \( \forall j \neq i. TS(j).P = TS'(j).P \).

We also know that \( G \) satisfies this condition.

Hence the condition is preserved in \( G' \).

Note. This was same as the other scenario when we append a new \text{St}_o(x, v) or \text{Ld}_o(x, v).

5. Condition to show: For every location \( \ell \) and thread \( j \), the thread view of \( \ell \) in the promise state \( MS' \) records the timestamp of the maximal write visible to the covered events in \( G' \) of thread \( j \).

The argument is analogous to the case when we append a new \text{Ld}_o(x, v).

6. Condition to show: The \( S' \) events in \( G' \) preserve coherence: \( \text{shb}' \); \text{seco}' is irreflexive.

The argument is analogous to the case when we append a new \text{Ld}_o(x, v).
A. Proving Simulation of Promising Semantics by WEAKEST

7. Condition to show: The atomicity condition for update operations holds for \( S' \) events in \( G' \).

The argument is analogous to the case when we append a new \( \text{ld}_o(x, v) \).

8. Condition to show: The sc fences in \( G' \) are appropriately ordered by \( \text{sc}' \).

The argument is analogous to the case when we append a new \( \text{ld}_o(x, v) \).

9. Condition to show: The behavior of \( \text{MS}' \) matches that of the \( S' \) events in \( G' \).

Essentially we have to show, \( \text{Behavior}(\text{MS}') = \text{Behavior}(G', \mathcal{W}', S') \).

We know \( \text{Behavior}(\text{MS}) = \text{Behavior}(G, \mathcal{W}, S) \) holds.

By definition, we have \( \text{Behavior}(\text{MS}') = \text{Behavior}(\text{MS}) \) and \( \text{Behavior}(G', \mathcal{W}', S') = \text{Behavior}(G, \mathcal{W}, S) \) by definition. As a result, \( \text{Behavior}(\text{MS}') = \text{Behavior}(G', \mathcal{W}', S') \) holds.

**Case UPDATE** \( U(o, x, v, v') \) reading from message \( \mathcal{w}_m = \langle x : v@(-, t), R \rangle \) and creating message \( m' = \langle x : v'@(-, t'), R' \rangle \):

In the event structure we extend the event structure \( G \) to \( G' \). We extend the cover set \( S_i \) as well as the relations \( \text{srf}, \text{smo} \) to \( S_i' \) along with the respective relations \( \text{srf}', \text{smo}' \) by including an event \( e' \) where

1. \( \text{dom}(G.\text{po}; \{\{e'\}\}) = S_0 \cup S_i \),
2. \( e' \in S_i' \setminus S_i \), and
3. \( \text{labels}(\text{sequence}_{G.\text{po}}(S_i)).(e'.\text{lab}) \in \mathcal{P}(i) \).

In this case the promise machine is updated as follows.

\( M' = M \cup \{m'\} \), \( S' = S \), and \( TS' = TS[\langle \mathcal{P}(i), \text{labels}(\text{sequence}_{srf}(S_i')) \rangle, V', TS(i).P] \)

where \( V' = TS(i).V[x \mapsto m'.ts] \).

Now we do a case analysis on whether such an update event \( e' \) exists in \( G \) or we append a new event.

**Subcase** \( \exists e' \in (G.E \setminus S_i) \). \( \text{dom}(G.\text{po}; \{\{e'\}\}) = S_0 \cup S_i \wedge e'.\text{lab} = U(o, x, v, v') \wedge G.\text{rf}(w_m, e') \)

where \( \mathcal{W}(w_m) = \mathcal{w}_m \):

We create \( e' \) such that \( e'.\text{lab} = U_o(x, v, v') \) and append \( e' \) to event structure \( G \) to create \( G' \).

In that case

- \( G'.E = G.E \uplus \{e'\} \)
- \( G'.\text{po} = (G.\text{po} \cup \{(e, e') \mid e \in (S_i \cup S_0)\})^+ \)
- \( G'.\text{jf} = G.\text{jf} \uplus \{(w_m, e') \mid \mathcal{W}(w_m) = \mathcal{w}_m \wedge [S_0 \cup S_i]; G'.\text{po}; \{w_m\} \} \)
- \( G'.\text{ew} = G.\text{ew} \)

Let: \( \mathcal{W'} \triangledown \mathcal{W}[e' \mapsto m'] \), and Based on \( \mathcal{W'} \), we derive following definitions in \( \text{MS}' \).
\[ \mathcal{S}^{' \triangleq} \mathcal{S} \cup \{e'\} \]

\[ \text{mo}^{'} \triangleq \text{mo} \cup \{(a, e') | a \in G \cdot \mathcal{W}_x \land \mathcal{W}(a) \neq \bot \land \mathcal{W}'(a).ts < \mathcal{W}'(e').ts \} \]

\[ \cup \{(e', a) | a \in G \cdot \mathcal{W}_x \land \mathcal{W}(a) \neq \bot \land \mathcal{W}'(e').ts < \mathcal{W}'(a).ts \} \]

\[ \text{sc}^{'} \triangleq \text{sc} \]

\[ \text{spo}^{'} \triangleq (\text{spo} \cup \{(e, e') | e \in \mathcal{S}_0 \cup \mathcal{S}_1\})^+ \]

\[ \text{srf}^{'} \triangleq \text{srf} \cup \{(w, e') | G'.\text{rf}(w, e') \land w \in \mathcal{S}\} \]

Now we check whether \( G' \sim_{\{i\}} (TS', S', M') \).

1. Condition to show: \( G' \) is consistent in weakest model.

   - (CF) and (CFJ) constraints are preserved in \( G' \). The arguments are analogous to the scenario when we append a new \( \text{Ld}_o(x, v) \).

   - (VISJ) We study the possible cases of \( w_m \).
     - If \( G'.\text{po}(w_m, e') \) then the condition holds as \((w_m, e') \notin G'.\text{rfe}\).
     - We will show that \( G' \) satisfies (CFJ) constraint. Hence \( w_m \) cannot be in conflict with \( e' \), that is, \((w_m, e') \notin G'.\text{cf}\).
     - \( w_m \) is in different thread and \( G'.\text{jfe}(w_m, e') \) holds. \( G \sim_{\{i\}} \) MS and the simulation rules ensures that there is no invisible event in the \( (T \setminus \{i\}) \) threads. Hence \( w_m \) is a visible event in \( G \) as well as in \( G' \).

   Considering the above mentioned cases \( G'.\text{jfe}(w_m, e') \Rightarrow w_m \in \text{vis}(G') \) holds and \( G' \) satisfies (VISJ) constraint.

   Note. This was same as the other scenario when we append a new \( \text{Ld}_o(x, v) \).

   - (ICF). We know \( G \) satisfies constraint (ICF). Following the construction \( e' \in G'.\mathcal{R} \) and following the determinacy condition if \( G' \sim (e_1, e') \) then \( e_1 \in \text{Ld} \) or \( e_1 \in \text{U} \). Thus \((e_1, e') \in (G'.\mathcal{R} \times G'.\mathcal{R})\) and hence \( G' \) satisfies (ICF).

   Note. This was same as the other scenario when we append a new \( \text{Ld}_o(x, v) \).

   - (ICFJ) From the construction we know there exists no \( e_1 \) such that \( \text{imm} (\text{cf})(e_1, e') \) and \( G'.\text{rf}(\mathcal{W}^{-1}(w_m), e_1) \). Moreover, \( G \) satisfies constraint (ICFJ). As a result, \( G' \) satisfies (ICFJ).

   - (COH) We know that \( G \) satisfies (COH) constraint and hence \((G'.\text{hb}; G'.\text{eco}_\text{strong})\) is acyclic. We check if \((G'.\text{hb}; G'.\text{eco}_\text{strong})\) is acyclic.

   The incoming edges to event \( e' \) are \( G'.\text{hb}, G'.\text{jf} \) and there is outgoing \( G'.\text{fr}_{\text{strong}} \) edges.

   If \((G'.\text{hb}; G'.\text{eco}_\text{strong})\) forms a cycle then
   
   (i) event \( e' \) is in the cycle.

   (ii) \( G'.\text{fr}_{\text{strong}}(e', w') \) is in the cycle where \( w' \) is some write on \( x \).
A. Proving Simulation of Promising Semantics by WEAKEST

(iii) Either $G'.rf(−, e')$ or $G'.hb(−, e')$ incoming edge is part of the $(G'.hb; G'.eco^\ast_{strong})$ cycle.

Analyzing the cases on incoming edges to event $e'$ the $(G'.hb; G'.eco^\ast_{strong})$ cycle can be as follows.

- case $G'.rf(−, e')$ completes the $(G'.hb; G'.eco^\ast_{strong})$ cycle.

The $G'.rf(−, e')$ is either $G'.jf(w_m, e')$ or there exists $w_1$ such that $G'.ew(w_m, w_1)$ and $(w_1, e') \in (G'.ew; G'.jf)$. Thus the cycle can be one of the followings ways.

1. $G'.rf(w_m, e'), G'.fr_{strong}(e', w'),$ and $(w', w_m) \in (G'.hb; G'.eco^\ast_{strong})$.
2. $G'.rf(w_1, e'), G'.fr_{strong}(e', w'),$ and $(w', w_1) \in (G'.hb; G'.eco^\ast_{strong})$.

Also note that $G'.fr_{strong}(e', w')$ implies either $G.mo_{strong}(w_m, w')$ or $G.mo_{strong}(w_1, w')$ already hold in $G$.

Considering (1), (2), and possible reasons for $G'.fr_{strong}(e', w')$, we consider following subcases.

* subcase

(i) $G'.rf(w_m, e'), G'.fr_{strong}(e', w'),$ and $(w', w_m) \in (G'.hb; G'.eco^\ast_{strong})$ is the cycle, and $G.mo_{strong}(w_m, w')$

(ii) $G'.rf(w_1, e'), G'.fr_{strong}(e', w'),$ and $(w', w_1) \in (G'.hb; G'.eco^\ast_{strong})$ is the cycle, and $G.mo_{strong}(w_1, w')$

In case (i) $(w', w_m) \in (G'.hb; G'.eco^\ast_{strong})$ implies $(w', w_m) \in (G.hb; G.eco^\ast_{strong})$ holds in $G$.

In that case $(w', w_m) \in (G.hb; G.eco^\ast_{strong})$ and $G.mo_{strong}(w_m, w')$ forms a $(G.hb; G.eco^\ast_{strong})$ cycle in $G$.

This is not possible as $(G.hb; G.eco^\ast_{strong})$ is acyclic and hence a contradiction.

Thus $(G'.hb; G'.eco^\ast_{strong})$ is acyclic in this case.

Following the similar argument $(G'.hb; G'.eco^\ast_{strong})$ is acyclic in case (ii).

* subcase

(i) $G'.rf(w_m, e'), G'.fr_{strong}(e', w'),$ and $(w', w_m) \in (G'.hb; G'.eco^\ast_{strong})$ is the cycle, and $G.mo_{strong}(w_1, w')$

(ii) $G'.rf(w_1, e'), G'.fr_{strong}(e', w'),$ and $(w', w_1) \in (G'.hb; G'.eco^\ast_{strong})$ is the cycle, and $G.mo_{strong}(w_m, w')$

In case (i) following Lemma 6,
(a) \((w', w_m) \in (G'.hb; G'.eco\^2\_{\text{strong}})\) implies
\((w', w_m) \in (G.hb; G.eco\^2\_{\text{strong}})\) and hence \(ts(w') < ts(w_m)\),
(b) \(G.ew(w_m, w_1)\) implies \(ts(w_m) = ts(w_1)\), and
(c) \(G.mo\_{\text{strong}}(w_1, w')\) implies \(ts(w_1) < ts(w')\).

The combination of (a), (b), (c) contradicts the total order of timestamps.

Thus \((G'.hb; G'.eco\^2\_{\text{strong}})\) is acyclic in this case.

Following the similar argument \((G'.hb; G'.eco\^2\_{\text{strong}})\) is acyclic in case (ii).

- case \(G'.hb(-, e')\) completes the \((G'.hb; G'.eco\^2\_{\text{strong}})\) cycle.

In this case \(G'.rf(-, e')\) is not part of the \((G'.hb; G'.eco\^2\_{\text{strong}})\) cycle.

Hence \((w', e') \in (G'.hb; G'.eco\^2\_{\text{strong}})\) and \(G'.fr\_{\text{strong}}(e', w')\)
forms the \((G'.hb; G'.eco\^2\_{\text{strong}})\) cycle.

\(G'.fr\_{\text{strong}}(e', w')\) suggests two possibilities:

* subcase \(G'.hb(w_m, w')\).

Following Lemma 6,
(a) \(ts(w_m) < ts(w')\).
(b) From \((w', e') \in (G'.hb; G'.eco\^2\_{\text{strong}})\) we know \(ts(w') < ts(e')\).
(c) We also know \(G'.jf(w_m, e')\) implies \(ts(w_m) < ts(e')\).
(d) However, \(G'.fr\_{\text{strong}}(e', w')\) implies \(ts(e') < ts(w')\).

The combination of (a), (b), (c), (d) contradicts the total order of timestamps and hence \((G'.hb; G'.eco\^2\_{\text{strong}})\) is acyclic in this case.

* subcase \(G'.hb(w_1, w')\).

Following Lemma 6,
(a) \(ts(w_1) < ts(w')\).
(b) From \((w', e') \in (G'.hb; G'.eco\^2\_{\text{strong}})\) we know \(ts(w') < ts(e')\).
(c) We also know \(G'.rf(w_1, e')\) implies \(ts(w_1) = ts(e')\).
(d) However, \(G'.fr\_{\text{strong}}(e', w')\) implies \(ts(e') < ts(w')\).

The combination of (a), (b), (c), (d) contradicts the total order of timestamps and hence \((G'.hb; G'.eco\^2\_{\text{strong}})\) is acyclic in this case.

As a result, \(G'\) satisfies \((\text{COH})\).

Thus \(G'\) is consistent in \(\text{WEAKEST}\) model.
2. Condition to show: *The local state of each thread in MS' contains the program of that thread along with the sequence of covered events in G' of that thread.*

   In this we have to show \( \forall j. TS'(j).\sigma = \langle P(j), labels(sequence_{spo}(S'_j)) \rangle \).

   We know that the relation holds between MS and G.

   For \( j \neq i \), it is trivial because \( TS'(j) = TS(j) \) holds from MS to MS' and \( S'_j = S_j \) holds from G to G'.

   For \( j = i \), we know \( TS(i).\sigma = \langle P(i), labels(sequence_{spo}(S_i)) \rangle \).

   Hence following the definition of \( TS(i).\sigma, S'_i, spo' \) we get

   \[
   \langle P(i), labels(sequence_{spo}(S'_i)) \rangle = \langle P(i), labels(sequence_{spo}(S_i)) \cdot e'.lab \rangle = \langle P(i), TS(i).\sigma \cdot e'.lab \rangle = TS'(i).\sigma
   \]

   Hence the condition is preserved between MS' and G'.

   Note. This was similar to the other scenario when we append a new St_\( x, v \).

3. Condition to show: *Whenever W' maps an event of G' to a message in MS', then the location accessed and the written values match.*

   We know that the event to message mappings for existing events in G.E and messages M do not change.

   \( \forall e \in G'.E. e \neq e' \implies W'(e) = W(e) \)

   If \( e = e' \) then \( W'(e') = m' \) and \( e'.loc = m'.loc = x \) and \( e'.wval = m'.wval = v \).

   Hence \( W' \) preserves the condition.

   Note. This was similar to the other scenario when we append a new St_\( x, v \).

4. Condition to show: *For all outstanding promises of threads (T \{ i \}), there are corresponding write events in G' that are po-after S'.*

   We know that for each thread \( j \neq i \) the set of promises are preserved from MS to MS', that is, \( \forall j \neq i. TS(j).P = TS'(j).P \).

   We also know that G satisfies this condition.

   Hence the condition is preserved in G'.

   Note. This was similar to the other scenario when we append a new St_\( x, v \).

5. Condition to show: *For every location l and thread j, the thread view of l in the promise state MS' records the timestamp of the maximal write visible to the covered events in G' of thread j.*

   Essentially we have to show
∀j, ℓ. \( TS'(j).V(ℓ) = \max\{W'(e).ts \mid e \in \text{dom}([W_e]; G'.jf'; \text{shb}'_; \text{sc}'_; \text{shb}'_; [S'_j])}\)

For \( j \neq i \) or \( j = i \land ℓ \neq x \), it is trivial because \( TS'.V(ℓ) = TS.V(ℓ) \).

For \( j = i \land ℓ = x \), from the definition we know

\( TS(i).V(x) = \max\{W'(e).ts \mid e \in \text{dom}([W_e]; G'.jf'; \text{shb}'_; \text{sc}'_; \text{shb}'_; [S_i])\}\)

Following the promising semantics, we know \( TS'(i).V(x) \) extends the thread view of \( x \) from \( TS(i).V(x) \) by reading from \( \text{wm} \), and hence \( TS(i).V(x) < \text{wm}.ts \).

Moreover, following the semantics of update operation in promise machine \( \text{wm}.ts < m'.ts \).

Hence following the construction,

\( TS'(i).V(x) = m'.ts = \max\{W'(e).ts \mid e \in \text{dom}([W_e]; G'.jf'; \text{shb}'_; \text{sc}'_; \text{shb}'_; [S'_i])\}\)

Thus the condition is preserved between \( MS' \) and \( G' \).

6. Condition to show: The \( S' \) events in \( G' \) preserve coherence: \( \text{shb}' ; \text{seco}' ; \text{irreflexive} \).

The argument is analogous to the case when we append a new \( \text{St}_o(x, v) \).

7. Condition to show: The atomicity condition for update operations holds for \( S' \) events in \( G' \).

Assume \([G'.U \cap S'] ; (\text{sfr}' ; \text{smo}') \neq \emptyset\).

We know that \([G.U \cap S] ; (\text{sfr} ; \text{smo}) = \emptyset\) holds.

Hence \( e' \) is involved in atomicity violation. In that case two possibilities as follows:

- **case** There exists an update \( u \in (G.U_x \cap S) \) such that \( \text{sfr}(u, e') \) and \( \text{smo}(e'; u) \) holds.

  Assume \( u \) reads from \( w_1 \), that is, \( \text{srf}(w_1, u) \).
  
  \( \text{sfr}'(u, e') \) implies that \( \text{mo}(w_1, e') \) holds.
  
  \( \text{mo}'(w_1, e') \) implies \( \mathbb{W}'(w_1).ts < \mathbb{W}'(e').ts \).

  However, \( \text{srf}'(w_1, u) \) implies \( \mathbb{W}'(w_1).ts < \mathbb{W}'(u).ts \)
  
  and there is no write on \( x \) in the time range \( (\mathbb{W}'(w_1).ts, \mathbb{W}'(u).ts) \), that is, \( \#w' \in S' \cap G'.\mathbb{W}_x \). \( \mathbb{W}'(w_1).ts < \mathbb{W}'(w').ts < \mathbb{W}'(u).ts \).

  As a result, \( \mathbb{W}'(w_1).ts < \mathbb{W}'(e').ts < \mathbb{W}'(u).ts \) is not possible and hence \( \mathbb{W}'(u).ts < \mathbb{W}'(e').ts \) which implies \( \text{smo}'(u, e') \).

  \( \text{smo}'(u, e') \) and \( \text{smo}'(e', u) \) both cannot hold.

  Hence a contradiction and in this case atomicity holds in \( S' \) events in \( G' \).
A. Proving Simulation of Promising Semantics by WEAKEST

- **case** There exists a write \( w' \in (G'.W_x \cap S') \) such that \( \text{sfr}'(e', w') \) and \( \text{smo}'(w', e') \) hold.

\( \text{sfr}'(e', w') \) implies \( \text{smo}'(w, w') \), that is, \( \mathbb{W}'(w).ts < \mathbb{W}'(w').ts \).

However, \( \text{sfr}'(w, e') \) implies \( \mathbb{W}'(w).ts < \mathbb{W}'(e').ts \)

and there is no write on \( x \) in the time range \( (\mathbb{W}'(w).ts, \mathbb{W}'(e').ts] \), that is, \( \nexists w' \in (G'.W_x \cap S') \). \( \mathbb{W}'(w).ts < \mathbb{W}'(w').ts < \mathbb{W}'(e').ts \).

As a result, neither \( \mathbb{W}'(w).ts < \mathbb{W}'(e').ts < \mathbb{W}'(e').ts \) is not possible and hence \( \mathbb{W}'(e').ts < \mathbb{W}'(w').ts \) which implies \( \text{smo}'(e', w') \).

\( \text{smo}'(e', w') \) and \( \text{smo}'(w', e') \) both cannot hold.

Hence a contradiction and in this case atomicity holds in \( S' \) events in \( G' \).

8. **Condition to show:** *The SC fences in \( G' \) are appropriately ordered by \( \text{sc}' \).*

We know \( [G.F_\text{sc}]; \text{shb} \cup \text{shb}; \text{seco}; \text{shb}; [G.F_\text{sc}] \subseteq \text{sc} \) holds in \( G \).

From definitions we know, \( G'.F_\text{sc} = G.F_\text{sc}, \text{sc}' = \text{sc}, \text{shb} \subseteq \text{shb}', \text{seco} \subseteq \text{seco}' \).

Consider \( a, b \) are two SC fences such that \( (a, b) \in [G.F_\text{sc}]; \text{shb} \cup \text{shb}; \text{seco}; \text{shb}; [G.F_\text{sc}] \subseteq \text{sc} \), and \( \text{sc}(a, b) \) holds.

In that case \( (a, b) \in (\text{shb} \cup \text{shb}; \text{seco}'; \text{shb}') \) holds and \( \text{sc}'(a, b) \) holds.

To show \( [G'.F_\text{sc}]; \text{shb} \cup \text{shb}; \text{seco}; \text{shb}; [G'.F_\text{sc}] \subseteq \text{sc} \), we have to show \( (b, a) \notin (\text{shb} \cup \text{shb}; \text{seco}; \text{shb}') \).

We show that by contradiction. Assume \( (b, a) \in (\text{shb} \cup \text{shb}; \text{seco}; \text{shb}') \).

This is possible due to the relations created to/from event \( e' \).

Considering the relations in \( \text{shb}' \) and \( \text{seco}' \), the incoming relations to event \( e' \) are \( \text{shb}' \), \( \text{srf}' \), \( \text{sf}' \), \( \text{smo}' \) and the outgoing edges are \( \text{sfr}' \), \( \text{smo}' \).

Since \( e' \) is an update, for a write event \( w_1 \), relation \( \text{sfr}'(u, w_1) \) implies \( \text{smo}'(u, w_1) \).

Hence we consider only \( \text{smo}' \) as outgoing edge.

In this case the path from \( b \) to \( a \) is \( (b, e') \in (\text{shb}; \text{seco}' \) and \( (e', a) \in (\text{smo}; \text{seco}' \); \text{shb}' \).

As there is no outgoing \( \text{srf} \) edge from \( e' \), no new synchronization edge is created, that is, \( \text{ssw}' = \text{ssw} \).

We analyze the cases of \( (b, e') \in (\text{shb}; \text{seco}'). \)

In this case there exists some event \( c \) such that

- \( \text{shb}'(b, e') \).

Two possible subcases:

- **subcase** In this case \( \text{shb}(b, e) \) and \( \text{spo}'(e, e') \) holds.

So \( \text{MS}_b.\text{TS}(b.\text{tid}).V(x) \leq \text{MS}_c.\text{TS}(e.\text{tid}).V(x) < \text{MS}_e.\text{TS}(e'.\text{tid}).V(x) \).
– **subcase** $shb(b, c)$ and $ssw'(c, e')$ holds.

Hence $MS_b . TS(b.tid).V(x) \leq MS_c . TS(c.tid).V(x)$ holds.

Moreover, consider the cases of $ssw'$, following from Lemma 6, we can show that

$MS_{c'} . TS(c'.tid).V(x) < MS_{e'} . TS(e'.tid).V(x)$ holds.

Considering both subcases $MS_b . TS(b.tid).V(x) < MS_{e'} . TS(e'.tid).V(x)$ holds.

- **subcase** $shb'(b, c)$ and $srf'(c, e')$.

Hence $shb; seco(b, c)$ and $srf'(c, e')$ holds.

As a result, following promising semantics,

$MS_b . TS(b.tid).V(x) \leq MS_c . TS(c.tid).V(x) < MS_{e'} . TS(e'.tid).V(x)$.

- **subcase** $shb'; seco'(b, c)$ and $srf'(c, e')$.

Hence $shb; seco(b, c)$ and $smo'(c, e')$ holds.

As a result, following promising semantics,

$MS_b . TS(b.tid).V(x) \leq MS_c . TS(c.tid).V(x) < MS_{e'} . TS(e'.tid).V(x)$.

Now we analyze $(e', a) \in smo'; seco'; shb'$.

In this case there exist a write $w \in S$ such that

$smo'(e', w)$ and $(w, a) \in seco'; shb$ holds.

Hence $MS_{e'} . TS(e'.tid).V(x) < MS_{a'} . TS(w.tid).V(x) \leq MS_a . TS(a.tid).V(x)$.

As a result, in all cases $MS_b . TS(b.tid).V(x) < MS_a . TS(a.tid).V(x)$ holds.

However, we know that $sc(a, b)$ holds and hence $MS_a . V \leq MS_b . V$.

This is a contradiction and hence $(b, a) \notin (shb' \cup shb'; seco'; shb')$.

As a result, $[G'. F_{sc}]; shb' \cup shb'; seco'; shb; [G'. F_{sc}] \subseteq sc'$ holds.

9. **Condition to show:** *The behavior of MS' matches that of the S' events in G'.*

The argument is analogous to the case when we append a new $St_w(x, v)$.

**Subcase** $\exists e' \in (G, E_i \setminus S_i), \ dom(G, po; \{e'\}) = S_0 \cup S_i \cap e'.lab = U(o, x, v, v') \land G.jf(w_m, e')$

where $wm = \mathbb{W}(w_m)$:

- We take $G' = G$ and let $\mathbb{W}' = \mathbb{W}[e' \mapsto m'].$

Based on $\mathbb{W}'$, we derive following definitions in MS'.
A. Proving Simulation of Promising Semantics by WEAKEST

• $S' \triangleq S \cup \{e'\}$
• $mo' \triangleq mo \cup \{(a,e') | a \in G.\mathcal{W}(a) \neq \bot \land \mathcal{W}'(a).ts < \mathcal{W}'(e').ts\}$
• $\cup \{(e', a) | a \in G.\mathcal{W}(a) \neq \bot \land \mathcal{W}'(e').ts < \mathcal{W}'(a).ts\}$
• $sc' \triangleq sc$
• $spo' \triangleq (spo \cup \{(e, e') | e \in S_0 \cup S_i\})^+$
• $srf' \triangleq srf \cup \{(w, e') | G'.rf(w, e') \land w \in S\}$

Now we check whether $G' \sim_{\{i\}} (TS', S', M')$.

1. Condition to show: $G'$ is consistent in WEAKEST model.
   We know $G'.E = G.E, G'.po = G.po, G'.jf = G.jf$, and $G$ is consistent. Hence $G'$ is also consistent in WEAKEST model.

2. Condition to show: The local state of each thread in $MS'$ contains the program of that thread along with the sequence of covered events in $G'$ of that thread.
   In this we have to show $\forall j. TS'(j).\sigma = \langle P(j), labels(sequence_{spo}(S_j'))\rangle$.
   We know that the relation holds between $MS$ and $G$.
   For $j \neq i$, it is trivial because $TS'(j) = TS(j)$ holds from $MS$ to $MS'$ and $S_j' = S_j$ holds from $G$ to $G'$.
   For $j = i$, we know $TS(i).\sigma = \langle P(i), labels(sequence_{spo}(S_i))\rangle$.
   Hence following the definition of $TS(i).\sigma, S'_i, spo'$ we get
   $$\langle P(i), labels(sequence_{spo}(S'_i))\rangle$$
   $$= \langle P(i), labels(sequence_{spo}(S_i)).e'.lab\rangle$$
   $$= \langle P(i), TS(i).\sigma.e'.lab\rangle$$
   $$= TS'(i).\sigma$$
   Hence the condition is preserved between $MS'$ and $G'$.
   Note. This was same as the other scenario when we append a new $St_e(x, v)$.

3. Condition to show: Whenever $\mathcal{W}'$ maps an event of $G'$ to a message in $MS'$, then the location accessed and the written values match.
   The event to message mappings for existing events in $G.E$ and messages $M$ do not change.
   $$\forall e \in G'.E. e \neq e' \implies \mathcal{W}'(e) = \mathcal{W}(e)$$
   If $e = e'$ then $\mathcal{W}'(e') = wmsg(op) = m'$ and $e'.loc = m'.loc = x$ and $e.wval = m'.wval = v$.
   Hence $\mathcal{W}'$ preserves the condition.
4. Condition to show: For all outstanding promises of threads \((T \setminus \{i\})\), there are corresponding write events in \(G'\) that are po-after \(S'\).

We know that for each thread \(j \neq i\) the set of promises are preserved from \(MS\) to \(MS'\), that is, \(\forall j \neq i. TS(j).P = TS'(j).P\).

We also know that \(G\) satisfies this condition.

Hence the condition is preserved in \(G'\).

Note. This was same as the other scenario when we append a new \(St_o(x, v)\).

5. Condition to show: For every location \(\ell\) and thread \(j\), the thread view of \(\ell\) in the promise state \(MS'\) records the timestamp of the maximal write visible to the covered events in \(G'\) of thread \(j\).

The argument is analogous to the case when we append a new \(U_o(x, v, v')\).

6. Condition to show: The \(S'\) events in \(G'\) preserve coherence: \(shb'; seco'\) is irreflexive.

The argument is analogous to the case when we append a new \(U_o(x, v, v')\).

7. Condition to show: The atomicity condition for update operations hold for \(S'\) events in \(G'\).

The argument is analogous to the case when we append a new \(U_o(x, v, v')\).

8. Condition to show: The \(SC\) fences in \(G'\) are appropriately ordered by \(sc'\).

We know \([G.F_{sc}]; shb \cup shb; seco; shb; [G.F_{sc}] \subseteq sc\) holds in \(G\).

The argument is analogous to the case when we append a new \(U_o(x, v, v')\).

9. Condition to show: The behavior of \(MS'\) matches that of the \(S'\) events in \(G'\).

The argument is analogous to the case when we append a new \(U_o(x, v, v')\).

Case Release fence \(F_{rel}\):

In the event structure we extend the event structure \(G\) to \(G'\). We extend the cover set \(S_i\) as well as the relations \((spo, srf, smo)\) to \(S'_i\) along with the respective relations \((spo', srf', smo')\) by including an event \(e'\) where

1. \(\text{dom}(G.po; \{e'\}) = S_0 \cup S_i\),
2. \(e' \in S'_i \setminus S_i\), and
3. \(\text{labels}(\text{sequence}_{G.po}(S_i)).(e'.lab) \in P(i)\).

In this case the promise machine is updated as follows.

\(M' = M, S' = S,\)

and \(TS' = TS[i \mapsto (\langle P(i), \text{labels}(\text{sequence}_{spo'}(S'_i)), \langle V.cur, V.acq, V.rel'\rangle, TS(i).P)\]
A. Proving Simulation of Promising Semantics by \textsc{weakest}

Now we do a case analysis on whether such an release fence event \(e'\) exists in \(G\) or we append a new event.

**Subcase** \(\exists e' \in (G.E_i \setminus S_i)\). \(\text{dom}(G.po; \{\{e'\}\}) \subseteq S_i \land e'.lab = F_{\text{REL}}\):

We create \(e'\) such that \(e'.lab = F_{\text{REL}}\) and append \(e'\) to event structure \(G\) to create \(G'\). Then,

- \(G'.E = G.E \uplus \{e' \mid e'.lab = F_{\text{REL}}\}\)
- \(G'.po = (G.po \cup \{(e, e') \mid e \in (S_i \cup S_0)\})^+\)
- \(G'.jf = G.jf\)
- \(G'.ew = G.ew\)

Let: \(\mathcal{W}' \triangleq \mathcal{W}\).

Based on \(\mathcal{W}'\), we derive following definitions in \(\text{MS}'\).

- \(S' \triangleq S \uplus \{e'\}\)
- \(\text{mo}' \triangleq \text{mo}\)
- \(\text{sc}' \triangleq \text{sc}\)
- \(\text{spo}' \triangleq (\text{spo} \uplus \{(e, e') \mid e \in S \cup S'\})^+\)
- \(\text{srf}' \triangleq \text{srf}\)

Now we check whether \(G' \sim_{\{i\}} (\mathcal{T}S', S', M')\).

1. **Condition to show:** \(G'\) is consistent in \textsc{weakest} model.

   - (CF) and (CFJ) constraints are preserved in \(G'\). The arguments are analogous to the scenario when we append a new \(St_o(x, v)\).
   - (VISJ) Constraint (VISJ) is preserved in \(G'\) as \(G'.jf = G.jf\) and \(G\) satisfies constraint (VISJ).
   - (ICF)

We know that \(G\) satisfies (ICF). Suppose there exists an event \(e_1 \in G\) which is in immediate conflict with \(e'\) in \(G'\), that is \(G'. \sim (e_1, e')\) holds.

Then (1) \(\text{dom}(G.po; \{\{e_1\}\}) = S_0 \cup S_i\),
(2) \(e_1 \in S'_i \setminus S_i\), and
(3) \(labels(\text{sequence}_{G.po}(S_i)).(e_1.lab) \in \mathcal{P}(i)\).

However, from definition of \(e'\) we already know that
(1) \(\text{dom}(G.po; \{\{e'\}\}) = S_0 \cup S_i\),
(2) \(e' \in S'_i \setminus S_i\), and
(3) \(labels(\text{sequence}_{G.po}(S_i)).(e'.lab) \in \mathcal{P}(i)\).
Hence following the determinacy condition we know either \(e_1 = e'\) or there exists no such \(e_1\).

Hence (ICF) is preserved in \(G'\).

Note. This was similar to the scenario when we append a new \(S_{t_o}(x, v)\).

- (ICFJ) Constraint (ICFJ) is preserved in \(G'\) as \(e' \notin \mathcal{R}\) and \(G\) satisfies constraint (ICFJ).

- (COH) We know \(G\) preserves (COH) constraint, that is, \((G.\text{hb}; G.\text{eco}_{\text{strong}}')\) is acyclic. The incoming edges to event \(e'\) are \(G'.\text{po}\) and there is no outgoing edge concerning \(G'.\text{hb}\) or \(G'.\text{eco}_{\text{strong}}\). As a result, \((G'.\text{hb}; G'.\text{eco}_{\text{strong}}')\) is acyclic and \(G'\) preserves (COH) constraint.

2. Condition to show: The local state of each thread in \(MS'\) contains the program of that thread along with the sequence of covered events in \(G'\) of that thread.

In this we have to show \(\forall j. TS'(j).\sigma = (\mathcal{P}(j), \text{labels}(\text{sequence}_{\text{spo}}(S'_j)))\).

We know that the relation holds between \(MS\) and \(G\).

For \(j \neq i\), it is trivial because \(TS'(j) = TS(j)\) holds from \(MS\) to \(MS'\) and \(S'_j = S_j\) holds from \(G\) to \(G'\).

For \(j = i\), we know \(TS(i).\sigma = (\mathcal{P}(i), \text{labels}(\text{sequence}_{\text{spo}}(S_i)))\).

Hence following the definition of \(TS(i).\sigma, S'_i, \text{spo}'\) we get

\[
\begin{align*}
\langle \mathcal{P}(i), \text{labels}(\text{sequence}_{\text{spo}}(S'_i)) \rangle &= \langle \mathcal{P}(i), \text{labels}(\text{sequence}_{\text{spo}}(S_i)) \cdot e'.\text{lab} \rangle \\
&= \langle \mathcal{P}(i), TS(i).\sigma \cdot e'.\text{lab} \rangle \\
&= TS'(i).\sigma
\end{align*}
\]

Hence the condition is preserved between \(MS'\) and \(G'\).

3. Condition to show: Whenever \(\mathcal{W}'\) maps an event of \(G'\) to a message in \(MS'\), then the location accessed and the written values match.

We know that the event to message mappings for existing events in \(G.\text{E}\) and messages \(M\) do not change, that is, \(\forall e \in G'.\text{E}. e \neq e' \implies \mathcal{W}'(e) = \mathcal{W}(e)\). If \(e = e'\) then \(\mathcal{W}'(e') = \bot\).

Hence \(\mathcal{W}'\) preserves the condition.

4. Condition to show: For all outstanding promises of threads \((T \setminus \{i\})\), there are corresponding write events in \(G'\) that are po-after \(S'\).

We know that for each thread \(j \neq i\) the set of promises are preserved from \(MS\) to \(MS'\), that is, \(\forall j \neq i. TS(j).P = TS'(j).P\).

We also know that \(G\) satisfies this condition.

Hence the condition is preserved in \(G'\).
A. Proving Simulation of Promising Semantics by WEAKEST

5. Condition to show: For every location $\ell$ and thread $j$, the thread view of $\ell$ in the promise state $MS'$ records the timestamp of the maximal write visible to the covered events in $G'$ of thread $j$.

Essentially we have to show
\[
\forall j, \ell. \; TS'(j).V(\ell) = \max\{\forall'(e).ts \mid e \in \text{dom}(\forall[j]; G'.jf'; \text{shb'}; \text{sc'}; \text{shb'}; [S'_j])\}
\]
We know the relation holds in $G$.
In $G'$, for all $j, \ell$, $TS'(j).V(\ell) = TS(j).V(\ell)$ considering the mapping of $TS'$.
Hence $TS'$ satisfies the same condition and the relation holds between $MS'$ and $G'$.

6. Condition to show: The $S'$ events in $G'$ preserve coherence: $\text{shb'}; \text{seco}'$ is irreflexive.

We know $\text{shb}; \text{seco}'$ is irreflexive.
Following the definition of components of $\text{shb'}$ and $\text{seco}'$ we know $\text{shb'}; \text{seco}'$ is irreflexive.

7. Condition to show: The atomicity condition for update operations holds for $S'$ events in $G'$.

We know that $[G'.U \cap S'] = [G.U \cap S]$ and $[G.U \cap S]; (\text{sfr}; \text{smo}) = \emptyset$ holds.
The $e'$ does not introduce any $[G.U]; G'.\text{sfr'}$ or $[G.U]; G'.\text{smo'}$ edge.
As a result, $[G'.U \cap S']; (\text{sfr'}; \text{smo'}) = \emptyset$ holds.

8. Condition to show: The $\text{sc}$ fences in $G'$ are appropriately ordered by $\text{sc'}$.

There is no outgoing edge from $e'$ to any event in $S'$.
Hence event $e'$ cannot introduce a new ($\text{shb'} \cup \text{shb'}; \text{seco'}; \text{shb'}$) path between two SC fences.
Hence $[G'.\mathcal{F}_{\text{sc}}]; \text{shb'} \cup \text{shb'}; \text{seco'}; \text{shb'}; [G'.\mathcal{F}_{\text{sc}}]$ implies $[G.\mathcal{F}_{\text{sc}}]; \text{shb} \cup \text{shb}; \text{seco}; \text{shb}; [G.\mathcal{F}_{\text{sc}}]$.
We also know $\text{sc'} = \text{sc}$.
We also know $[G.\mathcal{F}_{\text{sc}}]; \text{shb} \cup \text{shb}; \text{seco}; \text{shb}; [G.\mathcal{F}_{\text{sc}}] \subseteq \text{sc}$.
Hence $[G'.\mathcal{F}_{\text{sc}}]; \text{shb'} \cup \text{shb'}; \text{seco'}; \text{shb'}; [G'.\mathcal{F}_{\text{sc}}] \subseteq \text{sc'}$ holds.

9. Condition to show: The behavior of $MS'$ matches that of the $S'$ events in $G'$.

Essentially we have to show, $\text{Behavior}(MS') = \text{Behavior}(G', \forall', S')$.
We know $\text{Behavior}(MS) = \text{Behavior}(G, \forall, S)$ holds.
From the definition we know,
$\text{Behavior}(MS') = \text{Behavior}(MS)$ and $\text{Behavior}(G', \forall', S') = \text{Behavior}(G, \forall, S)$ hold.
As a result, $\text{Behavior}(MS') = \text{Behavior}(G', \forall', S')$ holds.
Subcase $\exists e' \in (G.E_i \setminus S_i). \operatorname{dom}(G.\text{po}; \{e'\}) = S_0 \cup S_i \land e'.\operatorname{lab} = F_{\text{REL}}$:

Note that promising semantics does not promise over a release fence. As a result, the certificate steps do not have any release fence. Hence there is no existing release fence event correspond to any certificate step which can be referred later in the simulation step. As a result, this case is not possible.

Case ACQUIRE FENCE $F_{\text{ACQ}}$:

In the event structure we extend the event structure $G$ to $G'$. We extend the cover set $S_i$ as well as the relations $(\text{spo}, \text{srf}, \text{smo})$ to $S'_i$ along with the respective relations $(\text{spo}', \text{srf}', \text{smo}')$ by including an event $e'$ where

1. $\operatorname{dom}(G.\text{po}; \{e'\}) = S_0 \cup S_i$,
2. $e' \in S'_i \setminus S_i$, and
3. $\operatorname{labels}(\text{sequence}_{G.\text{po}}(S_i)).(e'.\operatorname{lab}) \in \mathbb{P}(i)$.

In this case the promise machine is updated as follows.

Let:

$W' \triangleq W$.

Based on $W'$, we derive following definitions in $M'$. Let $S' \triangleq S \cup \{e'\}$

- $m'o' \triangleq m'o$
- $s'c' \triangleq s'c$
- $s'p'o' \triangleq (s'p'o \cup \{(e, e') \mid e \in S_0 \cup S'_i\})^+$
- $s'r'f \triangleq s'r$.

Note that there may be incoming synchronization edges to the acquire fence, that is, $\text{ssw} \subseteq \text{ssw}'$ and hence $\text{shb} \subseteq \text{shb}'$.

Now we check whether $G' \sim_i (TS', S', M')$.

1. Condition to show: $G'$ is consistent in WEAKEST model.

- (CF) The constraint is preserved in $G'$. The argument is analogous to the scenario when we append a new $\text{Ld}_o(x, v)$. 

39
A. Proving Simulation of Promising Semantics by WEAKEST

- (CFJ) Constraint (CFJ) is preserved in \( G' \). The argument is analogous to the scenario when we append a new \( S_{t_0}(x, v) \).
- (VISJ) Constraint (VISJ) is preserved in \( G' \) as \( G'.jf = G.jf \) and \( G \) satisfies constraint (VISJ).
- (ICF)

We know that \( G \) satisfies (ICF). Suppose there exists an event \( e_1 \in G \) which is in immediate conflict with \( e' \) in \( G' \), that is \( G'. \sim (e_1, e') \) holds.

Then (1) \( \text{dom}(G.po; \{e_1\}) = S_0 \cup S_i \),
(2) \( e_1 \in S'_i \setminus S_i \), and
(3) \( \text{labels(sequence}_{G,po}(S_i)):(e_1.\text{lab}) \in \mathbb{P}(i) \).

However, from definition of \( e' \) we already know that
(1) \( \text{dom}(G.po; \{e'\}) = S_0 \cup S_i \),
(2) \( e' \in S'_i \setminus S_i \), and
(3) \( \text{labels(sequence}_{G,po}(S_i)):(e'.\text{lab}) \in \mathbb{P}(i) \).

Hence following the determinacy condition we know either \( e_1 = e' \) or there exists no such \( e_1 \).

Hence (ICF) is preserved in \( G' \).

Note. This was similar to the scenario when we append a new \( F_{\text{REL}} \).

- (ICFJ) Constraint (ICFJ) is preserved in \( G' \) as \( e' \not\in \mathcal{R} \) and \( G \) satisfies constraint (ICFJ).

- (COH) We know \( G \) preserves (COH) constraint, that is, \( (G.\text{hb}; G.\text{eco}_{\text{strong}}^G) \) is acyclic. The incoming edges to event \( e' \) are \( G'.\text{po} \) and \( G'.\text{hb} \) (due to \( G'.\text{sw} \) edges), and there is no outgoing edge concerning \( G'.\text{hb} \) or \( G'.\text{eco}_{\text{strong}}^G \). As a result, \( (G'.\text{hb}; G'.\text{eco}_{\text{strong}}) \) is acyclic and \( G' \) preserves (COH) constraint.

2. Condition to show: The local state of each thread in MS' contains the program of that thread along with the sequence of covered events in G' of that thread.

In this we have to show \( \forall j. \mathcal{T}S'(j).\sigma = \langle \mathbb{P}(j), \text{labels(sequence}_{\text{spo}}(S'_j)) \rangle \).

We know that the relation holds between MS and \( G \).

For \( j \neq i \), it is trivial because \( \mathcal{T}S'(j) = \mathcal{T}S(j) \) holds from MS to MS' and \( S'_j = S_j \) holds from \( G \) to \( G' \).

For \( j = i \), we know \( \mathcal{T}S(i).\sigma = \langle \mathbb{P}(i), \text{labels(sequence}_{\text{spo}}(S_i)) \rangle \).

Hence following the definition of \( \mathcal{T}S(i).\sigma, S'_i, \text{spo'} \) we get
\[
\langle \mathbb{P}(i), \text{labels(sequence}_{\text{spo}}(S'_i)) \rangle = \langle \mathbb{P}(i), \text{labels(sequence}_{\text{spo}}(S_i)) \cdot e'.\text{lab} \rangle
\]
\[ \langle P(i), TS(i).\sigma \cdot e'.\text{lab} \rangle = TS'(i).\sigma \]

Hence the condition is preserved between MS' and G'.

3. Condition to show: Whenever \( W' \) maps an event of \( G' \) to a message in MS', then the location accessed and the written values match.

We know that the event to message mappings for existing events in G.E and messages M do not change, that is, \( \forall e \in G'.E. \ e \neq e' \implies W'(e) = W(e) \). If \( e = e' \) then \( W'(e') = \bot \).

Hence \( W' \) preserves the condition.

4. Condition to show: For all outstanding promises of threads \( (T \setminus \{i\}) \), there are corresponding write events in \( G' \) that are po-after \( S' \).

We know that for each thread \( j \neq i \) the set of promises are preserved from MS to MS', that is, \( \forall j \neq i. \ TS(j).P = TS'(j).P \).

We also know that \( G \) satisfies this condition.

Hence the condition is preserved in \( G' \).

5. Condition to show: For every location \( \ell \) and thread \( j \), the thread view of \( \ell \) in the promise state MS' records the timestamp of the maximal write visible to the covered events in \( G' \) of thread \( j \).

Essentially we have to show
\[ \forall j, \ell. \ TS'(j).V(\ell) = \max\{W'(e).ts \mid e \in \text{dom}(\{W_i\}; G'.jf'; \text{shb'}; \text{sc'}; \text{shb'}; [S'_j])\} \]

We know the relation holds in \( G \).

In \( G' \),

- for all \( j \neq i \), \( TS'(j).V(\ell) = TS(j).V(\ell) \) considering the mapping of \( TS' \).
- For \( j = i \), \( TS'(j).V.\text{cur} = TS(j).V.\text{acq} \).

We know that \( TS(i).V.\text{cur} \leq TS(i).V.\text{acq} \) for all location \( \ell \).

As a result, in this case \( TS'(i).V.\text{cur} \geq TS(i).V.\text{cur} \).

Hence
\[ \forall \ell. \ TS'(i).V(\ell) = \max\{W'(e).ts \mid e \in \text{dom}(\{W_i\}; G'.jf'; \text{shb'}; \text{sc'}; \text{shb'}; [S'_j])\} \]

holds.

Thus the relation holds between MS' and G'.
A. Proving Simulation of Promising Semantics by WEAKEST

6. Condition to show: The $S'$ events in $G'$ preserve coherence: $shb'$; $seco'$ is irreflexive.
   We know $shb'; seco'$ is irreflexive.

   Following the definition of components of $shb'$ and $seco'$ we know $shb'; seco'$ is irreflexive.

7. Condition to show: The atomicity condition for update operations holds for $S'$ events in $G'$.
   The argument is analogous to the case when we append a new $F_{REL}$.

8. Condition to show: The SC fences in $G'$ are appropriately ordered by $sc'$.
   The argument is analogous to the case when we append a new $F_{REL}$.

9. Condition to show: The behavior of $MS'$ matches that of the $S'$ events in $G'$.
   The argument is analogous to the case when we append a new $F_{REL}$.

Subcase $\exists e' \in (G.E_i \setminus S_i), \, \text{dom}(G.po; \{e'\}) = S_0 \cup S_i \wedge e'.lab = F_{ACQ}$:

Note that promising semantics does not promise over an acquire fence. As a result, the certificate steps do not have any acquire fence. Hence there is no existing acquire fence event correspond to any certificate step which can be referred later in the simulation step. As a result, this case is not possible.

Case SC FENCE $F_{sc}$:

In the event structure we extend the event structure $G$ to $G'$. We extend the cover set $S_i$ as well as the relations (spo, srf, smo) to $S'_i$ along with the respective relations (spo', srf', smo') by including an event $e'$ where

1. $\text{dom}(G.po; \{e'\}) = S_0 \cup S_i$,
2. $e' \in S'_i \setminus S_i$, and
3. $\text{labels(}sequence_{G,po}(S_i)), e'.lab) \in \mathbb{P}(i)$.

In this case the promise machine is updated as follows.

$M' = M, S' = \{(x,t) \mid x \in Locs \wedge \max(TS(i).V.cur(x), t') \wedge (x,t') \in S\}$, and

$TS' = TS[i \mapsto (\{\mathbb{P}(i), \text{labels(}sequence_{spo}(S'_i)), S', TS(i).P\})$.

Now we do a case analysis on whether such an SC fence event $e'$ exists in G or we append a new event.

Subcase $\exists e' \in (G.E_i \setminus S_i), \, \text{dom}(G.po; \{e'\}) \subseteq S_i \wedge e'.lab = F_{sc}$:

We create $e'$ such that $e'.lab = F_{sc}$ and append $e'$ to event structure $G$ to create $G'$. Then,

- $G'.E = G.E \cup \{e' \mid e'.lab = F_{sc}\}$ and $G'.po = G.po \cup \{(e, e') \mid e \in (S_i \cup S_0)\}$
- $G'.jf = G.jf$
- $G'.ew = G.ew$
Let: \( W' \triangleq W \).

Based on \( W' \), we derive following definitions in \( MS' \).

- \( S' \triangleq S \cup \{ e' \} \)
- \( m' \triangleq m \)
- \( sc' \triangleq sc \cup \{ (a, e') \mid a \in (G.F_{SC} \cap S) \} \)
- \( spo' \triangleq (spo \cup \{ (e, e') \mid e \in S_0 \cup S_i \})^+ \)
- \( srf' \triangleq srf \)

Note that there may be incoming synchronization edges to the acquire fence, that is, \( ss_w \subseteq ss_w' \) and hence \( sh_b \subseteq sh_b' \).

Now we check whether \( G' \sim_{\{i\}} \{ TS', S', M' \} \).

1. Condition to show: \( G' \) is consistent in WEAKEST model.

   - \( (CF) \) The constraint is preserved in \( G' \). The argument is analogous to the scenario when we append a new \( L_d(x,v) \).
   - \( (CFJ) \) Constraint \( (CFJ) \) is preserved in \( G' \). The argument is analogous to the scenario when we append a new \( S_t(x,v) \).
   - \( (VISJ) \) Constraint \( (VISJ) \) is preserved in \( G' \) as \( G'.jf = G.jf \) and \( G \) satisfies constraint \( (VISJ) \).
   - \( (ICF) \)
     
     We know that \( G \) satisfies \( (ICF) \). Suppose there exists an event \( e_1 \in G \) which is in immediate conflict with \( e' \) in \( G' \), that is \( G'. \sim (e_1, e') \) holds.
     
     Then (1) \( \text{dom}(G,po; \{ \{ e_1 \} \}) = S_0 \cup S_i \),
     
     (2) \( e_1 \in S_i' \setminus S_i \), and
     
     (3) \( labels(\text{sequence}_{G,po}(S_i))(e_1, \text{lab}) \in P(i) \).
     
     However, from definition of \( e' \) we already know that
     
     (1) \( \text{dom}(G,po; \{ \{ e' \} \}) = S_0 \cup S_i \),
     
     (2) \( e' \in S_i' \setminus S_i \), and
     
     (3) \( labels(\text{sequence}_{G,po}(S_i))(e', \text{lab}) \in P(i) \).
     
     Hence following the determinacy condition we know either \( e_1 = e' \) or there exists no such \( e_1 \).
     
     Hence \( (ICF) \) is preserved in \( G' \).
     
     Note. This was similar to the scenario when we append a new \( F_{rel}(x,v) \).
   - \( (ICFJ) \) Constraint \( (ICFJ) \) is preserved in \( G' \) as \( e' \notin R \) and \( G \) satisfies constraint \( (ICFJ) \).
A. Proving Simulation of Promising Semantics by WEAKEST

- (COH) We know $G$ preserves (COH) constraint, that is, $(G.hb; G.eco^{strong})$ is acyclic. The incoming edges to event $e'$ are $G'.po$ and $G'.hb$ (due to $G'.sw$ edges), and there is no outgoing edge concerning $G'.hb$ or $G'.eco^{strong}$. As a result, $(G'.hb; G'.eco^{strong})$ is acyclic and $G'$ preserves (COH) constraint.

2. Condition to show: The local state of each thread in $MS'$ contains the program of that thread along with the sequence of covered events in $G'$ of that thread.

In this we have to show $\forall j. TS'(j).\sigma = \langle P(j), labels(sequence_{spo}(S'_j))\rangle$.

We know that the relation holds between $MS$ and $G$.

For $j \neq i$, it is trivial because $TS'(j) = TS(j)$ holds from $MS$ to $MS'$ and $S'_j = S_j$ holds from $G$ to $G'$.

For $j = i$, we know $TS(i).\sigma = \langle P(i), labels(sequence_{spo}(S_i))\rangle$.

Hence following the definition of $TS(i).\sigma, S'_i, spo$ we get

\[
\langle P(i), labels(sequence_{spo}(S'_i))\rangle = \langle P(i), labels(sequence_{spo}(S_i)) . e'.lab\rangle = \langle P(i), TS(i).\sigma . e'.lab\rangle = TS'(i).\sigma
\]

Hence the condition is preserved between $MS'$ and $G'$.

3. Condition to show: Whenever $\mathbb{W}'$ maps an event of $G'$ to a message in $MS'$, then the location accessed and the written values match.

We know that the event to message mappings for existing events in $G.E$ and messages $M$ do not change, that is, $\forall e \in G'.E. e \neq e' \implies \mathbb{W}'(e) = \mathbb{W}(e)$. If $e = e'$ then $\mathbb{W}'(e') = \bot$.

Hence $\mathbb{W}'$ preserves the condition.

4. Condition to show: For all outstanding promises of threads ($T \setminus \{i\}$), there are corresponding write events in $G'$ that are po-after $S'$.

We know that for each thread $j \neq i$ the set of promises are preserved from $MS$ to $MS'$, that is, $\forall j \neq i. TS(j).P = TS'(j).P$.

We also know that $G$ satisfies this condition.

Hence the condition is preserved in $G'$.

5. Condition to show: For every location $\ell$ and thread $j$, the thread view of $\ell$ in the promise state $MS'$ records the timestamp of the maximal write visible to the covered events in $G'$ of thread $j$.

Essentially we have to show

$$\forall j, \ell. TS'(j).V(\ell) = \max\{\mathbb{W}'(e).ts \mid e \in \text{dom}([\mathbb{W}]; G'; Jf'; shb'; \text{sc}'; shb'; [S'_j])\}.$$
We know the relation holds in $G$.

For $j \neq i$, it is trivial because $TS'.V(\ell) = TS.V(\ell)$.

For $j = i$, we know that for a given location $x$,

$TS'(i).V(x)$ extends $TS(i).V(x)$ by choosing between timestamp from $TS(i).V(x)$ and timestamp from $MS_c.TS'(c.tid).V(x)$ where $imm(sc')(c, e')$ holds.

Hence $\forall \ell. TS'(i).V(\ell) = max\{W(e).ts | e \in dom([W]; G'.j?; shb'; sc'; shb'; [S_i])\}$ holds.

Thus the relation holds between $MS'$ and $G'$.

6. Condition to show: The $S'$ events in $G'$ preserve coherence: $shb'; seco$ is irreflexive.

We know $shb; seco$ is irreflexive.

Following the definition of components of $shb'$ and $seco$ we know $shb'; seco$ is irreflexive.

7. Condition to show: The atomicity condition for update operations holds for $S'$ events in $G'$.

The argument is analogous to the case when we append a new $F_{\text{REL}}$.

8. Condition to show: The SC fences in $G'$ are appropriately ordered by $sc'$. 

There is no outgoing edge from $e'$ to any event in $S'$.

Hence event $e'$ cannot introduce a new ($shb' \cup shb'; seco'; shb'$) path between two SC fences.

Hence $[G'.\mathcal{F}_{sc}]; shb' \cup shb'; seco'; shb'; [G'.\mathcal{F}_{sc}]$ implies $[G.\mathcal{F}_{sc}]; shb \cup shb; seco; shb; [G.\mathcal{F}_{sc}]$.

We also know $sc \subset sc'$.

We also know $[G.\mathcal{F}_{sc}]; shb \cup shb; seco; shb; [G.\mathcal{F}_{sc}] \subseteq sc$.

Hence $[G'.\mathcal{F}_{sc}]; shb' \cup shb'; seco'; shb'; [G'.\mathcal{F}_{sc}] \subseteq sc'$ holds.

9. Condition to show: The behavior of $MS'$ matches that of the $S'$ events in $G'$.

The argument is analogous to the case when we append a new $F_{\text{REL}}$.

Subcase $\exists e' \in (G.E_i \setminus S_i). \text{dom}(G.po; \{e'\}) = S_0 \cup S_i \wedge e'.lab = F_{sc}$:

Note that promising semantics does not promise over an SC fence. As a result, the certificate steps do not have any SC fence. Hence there is no existing SC fence event correspond to any certificate step which can be referred later in the simulation step. As a result, this case is not possible.

Case FULFILL op = fulfill($m'$):
A. Proving Simulation of Promising Semantics by WEAKEST

In the event structure we extend the event structure \( G \) to \( G' \). We extend the cover set \( S_i \) as well as the relations (spo, srf, smo) to \( S_i' \) along with the respective relations (spo', srf', smo') by including a write (store or update) event \( e' \) where

1. \( \text{dom}(G\cdot po; \{e'\}) = S_0 \cup S_i \),
2. \( e' \in S_i' \setminus S_i \), and
3. \( \text{labels}(sequence_{G\cdot po}(S_i)).(e'.\text{lab}) \in \mathbb{P}(i) \).

Now we check whether \( G' \sim \{i\}(TS', S', M') \).

1. Condition to show: \( G' \) is consistent in WEAKEST model.

Subcase \( \exists e' \in (G.E \setminus S_i). \text{dom}(G\cdot po; \{e'\}) = S_0 \cup S_i \land (e'.\text{lab} = \text{St}_o(x, v') \lor (e'.\text{lab} = U_o(x, v, v') \land G.jf(w_m, e'))) \) where \( w_m = \mathbb{W}(w_m) \):

We create \( e' \) such that \( e'.\text{lab} = \text{St}_o(x, v') \) or \( e'.\text{lab} = U_o(x, v, v') \) accordingly and append \( e' \) to event structure \( G \) to create \( G' \). Then,

- \( G'\cdot E = G.E \uplus \{e'\} \)
- \( G'\cdot po = (G\cdot po \cup \{(e, e') \mid e \in (S_i \cup S_0)\})^+ \)
- \( G'\cdot jf = G.jf 
\cup \{(w_m, e') \mid e' \in U \land w_m \in G.W_x \land w.\text{wval} = v \land \mathbb{W}(w_m) = m\} \)
- \( G'\cdot ew = G.\text{ew} \uplus \{(w_p, e') \mid w_p.\text{id} \neq e'.\text{id} \land \mathbb{W}(w_p) = m'\} \)

Let: \( \mathbb{W}' \triangleq \mathbb{W}[e' \mapsto m'] \).

Based on \( \mathbb{W}' \), we derive following definitions in MS'.

- \( S' \triangleq S \uplus \{e'\} \)
- \( mo' \triangleq mo \uplus \{(a, e') \mid a \in G.W_x \land \mathbb{W}(a) \neq \perp \land \mathbb{W}'(a).ts < \mathbb{W}'(e').ts\} \)
  \uplus \{(e', a) \mid a \in G.W_x \land \mathbb{W}(a) \neq \perp \land \mathbb{W}'(e').ts < \mathbb{W}'(a).ts\} \)
- \( sc' \triangleq sc \)
- \( spo' \triangleq spo \uplus \{(e, e') \mid e \in S_0 \cup S_i'\})^+ \)
- \( srf' \triangleq srf \uplus \{(e', r) \mid (e', r) \in G'\cdot rf(e', r) \land r \in S'\} \)
  \uplus \{(w_m, e') \mid e' \in G'.U \land G'.rf(w_m, e') \land w_m \in S' \land w_m.\text{wval} = v \land \mathbb{W}'(w_m) = wm\} \)

Now we check whether \( G' \sim \{i\}(TS', S', M') \).

1. Condition to show: \( G' \) is consistent in WEAKEST model.
• (CF)

We know that $G$ satisfies (CF).

New $G'.hb$ edges are created by the incoming edges to $e'$. The outgoing $G'.rf$ edge from $e'$ does not result in any new synchronization.

The constraint is preserved in $G'$. If $e' \in G'.St$ then the argument is analogous to the scenario when we append a new $St_o(x, v)$ event. If $e' \in G'.U$ then the argument is analogous to the scenario when we append a new $U_o(x, v, v')$ event.

Hence $G'$ satisfies (CF).

• (CFJ)

We know that $G$ satisfies (CFJ).

Hence the new $hb$ edges are created by the incoming edges to $e'$. The outgoing $G'.rf$ edge from $e'$ does not result in any new synchronization.

In that case the (CFJ) constraint is preserved in $G'$. If $e' \in G'.St$ then the argument is analogous to the scenario when we append a new $St_o(x, v)$ event. If $e' \in G'.U$ then the argument is analogous to the scenario when we append a new $U_o(x, v, v')$ event.

• (VISJ)

– case $e' = St_o(x, v')$.

Constraint (VISJ) is preserved in $G'$ as $G'.jf = G.jf$ and $G$ satisfies constraint (VISJ).

Note. This was same as the other scenario when we append a new $St_o(x, v')$.

– case $e' = U_o(x, v, v')$.

We study the possible cases of $w_m$.

* If $G'.po(w_m, e')$ then the condition holds as $(w_m, e') \notin G'.jfe$.

* We will show that $G'$ satisfies (CFJ) constraint. Hence $w_m$ cannot be in conflict with $e'$, that is, $(w_m, e') \notin G'.cf$.

* $w_m$ is in different thread and $G'.jfe(w_m, e')$ holds. We know that $G \sim_{\{i\}} MS$ and the simulation rules ensures that there is no invisible event in the $(T \setminus \{i\})$ threads. Hence $w_m$ is a visible event in $G$ as well as in $G'$.

Considering the above mentioned cases $G'.jfe(w_m, e') \implies w_m \in vis(G')$ holds and $G'$ satisfies (VISJ) constraint.

Note. This was same as the other scenario when we append a new $U_o(x, v, v')$.

• (ICF) Constraint (ICF) is preserved in $G$. Now considering the cases of $e'$:

– case $e' = St_o(x, v')$.

Suppose there exists an event $e_1 \in G$ which is in immediate conflict with $e'$ in $G'$, that is $G'. \sim (e_1, e')$ holds.
A. Proving Simulation of Promising Semantics by **WEAKEST**

Then (1) \( \text{dom}(G.po; \{e_1\}) = S_0 \cup S_i \),
(2) \( e_1 \in S'_i \setminus S_i \), and
(3) \( \text{labels}(\text{sequence}_{G.po}(S_i)).(e_1.\text{lab}) \in \mathbb{P}(i) \).

However, from definition of \( e' \) we already know that
(1) \( \text{dom}(G.po; \{e'\}) = S_0 \cup S_i \),
(2) \( e' \in S'_i \setminus S_i \), and
(3) \( \text{labels}(\text{sequence}_{G.po}(S_i)).(e'.\text{lab}) \in \mathbb{P}(i) \).

Hence following the determinacy condition we know either \( e_1 = e' \) or there exists no such \( e_1 \).

Hence (ICF) is preserved in \( G' \).

- **case** \( e' = U_o(x, v, v') \).

  Following the construction \( e' \in G'.R \) and following the determinacy condition,

  if \( G'. \sim (e_1, e') \) then \( e_1 \in Ld \) or \( e_1 \in U \). Thus \( (e_1, e') \in (G'.R \times G'.R) \) and hence \( G' \) satisfies (ICF).

- (ICFJ) From the construction we know either \( e' \in St \) or there exists no \( e_1 \) such that \( \text{imm}(cf)(e_1, e') \) and \( G.rf(W^{-1}(wm), e_1) \). Moreover, \( G \) satisfies constraint (ICFJ).

  As a result, \( G' \) satisfies (ICFJ).

- (COH) We know \( G \) preserves (COH) constraint, that is, \( (G.hb; G.eco^2_{strong}) \) is acyclic.

Now we check if \( G' \) has \( (G'.hb; G'.eco^2_{strong}) \) cycle.

If there exists \( (G'.hb; G'.eco^2_{strong}) \) cycle then the cycle contains \( G'.rf(e', r) \)
and \( (r, e') \in (G'.hb; G'.eco^2_{strong}) \) holds.

Since \( (r, e') \notin (G'.hb), (r, e') \in (G'.hb; G'.eco_{strong}) \).

Now we consider the cases of event \( e' \).

- **case** \( e' = St_o(x, v') \).

  The incoming edges to event \( e' \) are \( G'.ew, G'.hb, G'.fr_{strong} \) edges and the outgoing edges are \( G'.ew, G'.rf \) edges.

  Note that as \( e' \) is a newly appended event and no read event reads from \( e' \) no new \( G'.rf(w_m, -) \) is created.

  In that case the incoming edge to \( e' \) is \( G'.fr_{strong} \) or \( G'.mo_{strong} \).

  * **subcase** \( G'.mo_{strong} \). Let \( G'.mo_{strong}(w_1, e') \) be the incoming edge. In that case, considering Lemma 6, \( W'(w_m).ts < W'(w_1).ts < W'(w').ts < W'(e').ts \). However, we know \( W'(w_m).ts = m'.ts = W'(e').ts \). Hence this is not possible.
subcase G'.fr\textsubscript{strong}. Let G'.fr\textsubscript{strong}(r_1, e') be the incoming edge.

Let G'.jf(w_1, r_1) holds. In that case G'.mo\textsubscript{strong}(w_1, e') holds and hence like the earlier case \( \mathbb{W}'(w_1).ts < m'.ts \) holds.

However, we know that \((r, r_1) \in G'.hb; G'.eco \textsubscript{strong} \) and hence following Lemma 6, \( m'.ts \leq \mathbb{W}'(w_1).ts \). Hence a contradiction. As a result, \((G'.hb; G'.eco \textsubscript{strong}) \) is irreflexive.

- case e' = U_o(x, v, v').

The incoming edges to event e' are G'.ew, G'.hb, G'.fr\textsubscript{strong}, and G'.rf edges and the outgoing edges are G'.ew, G'.rf edges.

Note that as e' is a newly appended event and no read event reads from e' no new G'.rf(w_m, -) is created.

The argument for incoming G'.ew, G'.hb, G'.fr\textsubscript{strong} edges are same as the earlier cases where e' is a store event.

So now we consider the case where G'.rf(-, e') is the incoming edge to e'. Let the edge be G'.rf(w'', e') and hence \((r, w'') \in (G'.hb; G'.eco \textsubscript{strong}) \).

Following Lemma 6,

1. \( m'.ts \leq \mathbb{W}'(w'').ts \). However, following the promising semantics for update operation we know that (2) \( \mathbb{W}'(e'.ts > \mathbb{W}'(w'').ts \) holds which implies \( m'.ts > \mathbb{W}'(w'').ts \).

The (1) and (2) contradicts and hence there is no \((G'.hb; G'.eco \textsubscript{strong}) \) cycle.

Hence \((G'.hb; G'.eco \textsubscript{strong}) \) is irreflexive.

Thus G' satisfies (COH).

As a result, G' is consistent in WEAKEST model.

2. Condition to show: The local state of each thread in MS' contains the program of that thread along with the sequence of covered events in G' of that thread.

In this we have to show \( \forall j. TS'(j).\sigma = (\mathbb{P}(j), \text{labels(sequence}_{spo}(S'_i))) \).

We know that the relation holds between MS and G.

For \( j \neq i \), it is trivial because \( TS'(j) = TS(j) \) holds from MS to MS' and \( S'_i = S_j \) holds from G to G'.

For \( j = i \), we know \( TS(i).\sigma = (\mathbb{P}(i), \text{labels(sequence}_{spo}(S_i))) \).

Hence following the definition of \( TS(i).\sigma, S'_i, spo' \) we get

\[
\langle \mathbb{P}(i), \text{labels(sequence}_{spo}(S'_i)) \rangle = \langle \mathbb{P}(i), \text{labels(sequence}_{spo}(S_i)) \cdot e'.lab \rangle = \langle \mathbb{P}(i), TS(i).\sigma \cdot e'.lab \rangle = TS'(i).\sigma
\]

Hence the condition is preserved between MS' and G'.
A. Proving Simulation of Promising Semantics by WEAKEST

3. Condition to show: Whenever $W'$ maps an event of $G'$ to a message in $MS'$, then the location accessed and the written values match.

We know that the event to message mappings for existing events in $G.E$ and messages $M$ do not change.

$$\forall e \in G'.E. e \neq e' \implies W'(e) = W(e)$$

If $e = e'$ then $W'(e') = m'$ and $e'.loc = m'.loc = x$ and $e'.wval = m'.wval = v'$.

Hence $W'$ preserves the condition.

4. Condition to show: For all outstanding promises of threads $(T \setminus \{i\})$, there are corresponding write events in $G'$ that are po-after $S'$.

We know that for each thread $j \neq i$ the set of promises are preserved from $MS$ to $MS'$, that is, $\forall j \neq i. TS(j).P = TS'(j).P$.

We also know that $G$ satisfies this condition.

Hence the condition is preserved in $G'$.

5. Condition to show: For every location $\ell$ and thread $j$, the thread view of $\ell$ in the promise state $MS'$ records the timestamp of the maximal write visible to the covered events in $G'$ of thread $j$.

Essentially we have to show

$$\forall j, \ell. TS'(j).V(\ell) = \max\{ W'(e).ts | e \in \text{dom}([W_i]; G'.j; f^f; shb^f; sc^f; shb^f; [S'_j])\}.$$  

For $j \neq i$ or $j = i \land \ell \neq x$, it is trivial because $TS'.V(\ell) = TS.V(\ell)$.

For $j = i \land \ell = x$,

Based on the type of event $e'$

- **case** $e' \in G.St_x$,

  following the promising semantics $W'(e') = m'$, $m'.ts$ extends the view on $x$ in thread $i$, and hence $TS(i).V(x) < TS'(i).V(x)$.

  In this case, $e' \in \text{dom}([W_i]; G'.j; f^f; shb^f; sc^f; shb^f; [S'_j])$.

  So $TS'(i).V(x) = \max\{ W'(e).ts | e \in \text{dom}([W_x]; G'.j; f^f; shb^f; sc^f; shb^f; [S'_j])\}$ holds.

- **case** $e' \in G.U_x$,

  Then, $TS(i).V(x) = \max\{ W(e).ts | e \in \text{dom}([W_x]; G.j; f^f; shb^f; sc^f; shb^f; [S_i])\}$ holds.

  Following the promising semantics, we know $TS'(i).V(x)$ extends the thread view of $x$ from $TS(i).V(x)$ by reading from some message $wm$, and so $TS(i).V(x) < wm.ts$.

  Moreover, following the semantics of update in the promise machine, $wm.ts < m'.ts$.  

50
So $\mathcal{T}S'(i).V(x) = \max \{\mathcal{W}'(e).ts \mid e \in \text{dom}(\mathcal{W}_x); G'.jf^o; \text{shb}^o; \text{sc}^o; \text{shb}^o; \mathcal{S}'[i])\}$. Thus the relation holds between $\text{MS}'$ and $G'$.

6. **Condition to show:** The $S'$ events in $G'$ preserve coherence: $\text{shb}'; \text{seco}^o$ is irreflexive. The argument is analogous to the new $\mathcal{S}_o(x, v, v')$ or new $\mathcal{U}_o(x, v, v')$ events.

7. **Condition to show:** The atomicity condition for update operations holds for $S'$ events in $G'$.

The argument is analogous to the new $\mathcal{S}_o(x, v, v')$ or new $\mathcal{U}_o(x, v, v')$ events.

8. **Condition to show:** The $\text{sc}$ fences in $G'$ are appropriately ordered by $\text{sc}'$.

   We know $[G'.F_{\text{sc}}]; \text{shb} \cup \text{shb}; \text{seco}; \text{shb}; [G'.F_{\text{sc}}] \subseteq \text{sc}$ holds in $G$.

   From definitions we know, $G'.F_{\text{sc}} = G.F_{\text{sc}}$, $\text{sc}' = \text{sc}$, $\text{shb} \subseteq \text{shb}'$, $\text{seco} \subseteq \text{seco}'$.

   Consider $a, b$ are two SC fences such that $(a, b) \in [G'.F_{\text{sc}}]; \text{shb} \cup \text{shb}; \text{seco}; \text{shb}; [G'.F_{\text{sc}}]$, and $\text{sc}(a, b)$ holds.

   In that case $(a, b) \in (\text{shb} \cup \text{shb}'; \text{seco}'; \text{shb}')$ holds and $\text{sc}'(a, b)$ holds.

   To show $[G'.F_{\text{sc}}]; \text{shb} \cup \text{shb}'; \text{seco}'; \text{shb}' \subseteq \text{sc}'$, we have to show $(b, a) \notin (\text{shb} \cup \text{shb}'; \text{seco}'; \text{shb}')$.

   We show that by contradiction. Assume $(b, a) \in (\text{shb} \cup \text{shb}'; \text{seco}'; \text{shb}')$.

   This is possible due to the relations created to/from event $e'$.

   Considering the relations in $\text{shb}'$ and $\text{seco}'$,

   (1) when $e' \in G'.\mathcal{S}_t$, the incoming relations to event $e'$ are $\text{shb}'$, $\text{srf}'$, $\text{smo}'$ and the outgoing edges are $\text{srf}'$, $\text{smo}'$.

   (2) when $e' \in G'.\mathcal{U}$, the incoming and outgoing relations to event $e'$ are same as when $e' \in G'.\mathcal{S}_t$. Additionally, there are $\text{srf}'$ incoming edges to $e'$.

   In this case the path from $b$ to $a$ is $(b, e') \in \text{shb}'; \text{seco}^o$,

   and $(e', a) \in \text{srf}'; \text{seco}^o; \text{shb}'$ or $(e', a) \in \text{smo}'; \text{seco}^o; \text{shb}'$.

   We analyze the cases of $(b, e') \in \text{shb}'; \text{seco}^o$.

   Similar to the new $\mathcal{S}_o(x, v, v')$ or the new $\mathcal{U}_o(x, v, v')$, in this case also $\text{MS}_b.\mathcal{T}\mathcal{S}(b.tid).V(x) < \text{MS}_e.\mathcal{T}\mathcal{S}(e'.tid).V(x)$ holds.

   Now we consider the outgoing edges:

   • $(e', a) \in \text{srf}'; \text{seco}^o; \text{shb}'$.

     There exists $r$ such that $\text{srf}'(e', a)$ and $(r, a) \in \text{seco}^o; \text{shb}'$.

     Hence, $\text{MS}_e.\mathcal{T}\mathcal{S}(e'.tid).V(x) = \text{MS}_r.\mathcal{T}\mathcal{S}(r.tid).V(x) \leq \text{MS}_a.\mathcal{T}\mathcal{S}(a.tid).V(x)$. 

51
A. Proving Simulation of Promising Semantics by WEAKEST

- \((e', a) \in \text{smo}' \cup \text{seco}' \cup \text{shb}'\).

There exists a write \(w \in S\) such that \(\text{smo}'(e', w)\) and \((w, a) \in \text{seco}' \cup \text{shb}'\).

Hence, \(\text{MS}_{e'.\text{TS}(e'.\text{tid}).\text{V}(x)} \leq \text{MS}_{a'.\text{TS}(a.\text{tid}).\text{V}(x)}\).

Considering both cases \(\text{MS}_{b'.\text{TS}(b.\text{tid}).\text{V}(x)}\) holds.

This is a contradiction and hence \((b, a) \notin (\text{shb'} \cup \text{seco}' \cup \text{shb}')\).

As a result, \([G'.F_{sc}]; \text{shb}' \cup \text{shb'} ; \text{seco}' ; \text{shb'} ; [G'.F_{sc}] \subseteq \text{sc}'\) holds.

9. Condition to show: The behavior of \(\text{MS}'\) matches that of the \(S'\) events in \(G'\).

The argument is analogous to the case when we append a new store or update event.

**Subcase** \(\exists e' \in (G.E_x \setminus S_x). \text{dom}(G.\text{po}([e'])) = S_0 \cup S_i \land (e'.\text{lab} = \text{St}_o(x, v') \lor (e'.\text{lab} = U_o(x, v, v') \land G.\text{rf}(w, m, e'))\) where \(w_m = w(v_m)\).

In this case an event created for the promise certificate corresponds to the fulfill operation. We take \(G' = G\) and let \(w' = w(v' \rightarrow m')\) and Based on \(w'\), we derive following definitions in \(\text{MS}'\).

- \(S' \triangleq S \cup \{e'\}\)
- \(\text{mo}' \triangleq \text{mo}\)
- \(\text{sc}' \triangleq \text{sc}\)
- \(\text{spo}' \triangleq \text{spo} \cup \{(e, e') \mid e \in S_0 \cup S_i\}^+\)
- \(\text{srf}' \triangleq \text{srf} \cup \{(e', r) \mid (e', r) \in G'.\text{rf}(e', r) \land r \in S'\}\)
  \(\cup \{(w_m, e') \mid e' \in G'.\text{U} \land G'.\text{rf}(w_m, e') \land w_m S' \land w_m \text{wval} = v \land w'(w_m) = w_m\}\)

Now we check whether \(G' \sim_{(i)} (\text{TS}', S', M')\).

1. Condition to show: \(G'\) is consistent in WEAKEST model.

\(G'\) is consistent as \(G\) is consistent in WEAKEST model.

2. Condition to show: The local state of each thread in \(\text{MS}'\) contains the program of that thread along with the sequence of covered events in \(G'\) of that thread.

In this we have to show \(\forall j. \text{TS}'(j).\sigma = \langle \text{P}(j), \text{labels}(\text{sequence}_{\text{spo}'}(S'_j))\rangle\).

We know that the relation holds between \(\text{MS}\) and \(G\).

For \(j \neq i\), it is trivial because \(\text{TS}'(j) = \text{TS}(j)\) holds from \(\text{MS}\) to \(\text{MS}'\) and \(S'_j = S_j\) holds from \(G\) to \(G'\).

For \(j = i\), we know \(\text{TS}(i).\sigma = \langle \text{P}(i), \text{labels}(\text{sequence}_{\text{spo}'}(S_i))\rangle\).

Hence following the definition of \(\text{TS}(i).\sigma, S'_i, \text{spo}'\) we get
\[ \langle P(i), \text{labels(sequencespo'}(S'_{i})) \rangle = \langle P(i), \text{labels(sequencespo}(S_i)) \cdot e'.\text{lab} \rangle = \langle P(i), TS(i).\sigma \cdot e'.\text{lab} \rangle = TS'(i).\sigma \]

Hence the condition is preserved between MS' and G'.

3. Condition to show: Whenever \( \mathbb{W}' \) maps an event of \( G' \) to a message in MS', then the location accessed and the written values match.

We know that the event to message mappings for existing events in \( G.E \) and messages \( M \) do not change.

\[ \forall e \in G'.E, e \neq e' \implies \mathbb{W}'(e) = \mathbb{W}(e) \]

If \( e = e' \) then \( \mathbb{W}'(e') = m' \) and \( e'.\text{loc} = m'.\text{loc} = x \) and \( e'.\text{wval} = m'.\text{wval} = v' \).

Hence \( \mathbb{W}' \) preserves the condition.

4. Condition to show: For all outstanding promises of threads \( (T \setminus \{i\}) \), there are corresponding write events in \( G' \) that are po-after \( S' \).

We know that for each thread \( j \neq i \) the set of promises are preserved from MS to MS', that is, \( \forall j \neq i, TS(j).P = TS'(j).P \).

We also know that \( G \) satisfies this condition.

Hence the condition is preserved in \( G' \).

5. Condition to show: For every location \( \ell \) and thread \( j \), the thread view of \( \ell \) in the promise state MS' records the timestamp of the maximal write visible to the covered events in \( G' \) of thread \( j \).

The argument is analogous to the new \( \text{St}_o(x, v, v') \) or new \( \text{U}_o(x, v, v') \) events.

Thus the relation holds between MS' and G'.

6. Condition to show: The S' events in \( G' \) preserve coherence: shb'; seco' is irreflexive.

The argument is analogous to the case when we append a new store or update event for a fulfill operation.

7. Condition to show: The atomicity condition for update operations holds for S' events in \( G' \).

The argument is analogous to the new store or update event.

8. Condition to show: The SC fences in \( G' \) are appropriately ordered by sc'.

The argument is analogous to the case when we append a new store or update event for a fulfill operation.
A. Proving Simulation of Promising Semantics by WEAKEST

9. Condition to show: The behavior of $MS'$ matches that of the $S'$ events in $G'$.

The argument is analogous to the case when we append a new store or update event.

Now we prove Lemma 2.

**Lemma 2.** $G \sim MS \land MS \rightarrow MS' \implies \exists G'. G \rightarrow_{\text{P},\text{WEAKEST}}^* G' \land G' \sim MS'$.

**Proof.** Following the promise machine step:

\[
\begin{align*}
\langle TS(i), S, M \rangle & \rightarrow \langle TS', S', M' \rangle & \langle TS', S', M' \rangle & \stackrel{\text{op}}{\rightarrow} \langle TS'', S'', M'' \rangle \\
\text{MS} = \langle TS, S, M \rangle & \quad \text{MS} = \langle TS[i \mapsto TS'], S', M' \rangle & \quad M''. P = \emptyset \\
\text{MS} & \stackrel{\text{op}}{\rightarrow} \text{MS}'
\end{align*}
\]

Case analysis on the op:

\[
\begin{align*}
\langle TS(i), S, M \rangle & \stackrel{\text{np}+}{\rightarrow} \langle TS', S', M' \rangle & \langle TS', S', M' \rangle & \stackrel{\text{np}}{\rightarrow} \langle TS'', S'', M'' \rangle \\
\text{MS} = \langle TS, S, M \rangle & \quad \text{MS} = \langle TS[i \mapsto TS'], S', M' \rangle & \quad M''. P = \emptyset \\
\text{MS} & \stackrel{\text{op}}{\rightarrow} \text{MS}'
\end{align*}
\]

**Case Non-promise step:**

From $G \sim MS$, we get $G \sim_{\{i\}} MS$.

By Lemma 1 and induction, we have

\[
\exists G'. G \rightarrow^* G' \land G' \sim_{\{i\}} \langle TS[i \mapsto TS'], S', M' \rangle
\]  

(i)

and by Lemma 1 and induction, we have

\[
\exists G''. G' \rightarrow^* G'' \land G'' \sim_{\{i\}} \langle TS[i \mapsto TS''], S'', M'' \rangle
\]  

(ii)

It remains to show $G'' \sim MS'$.

We know that a certificate does not create any new message or SC fence. Hence $M'' = M'$ and $S'' = S'$.

We take $W'' = W'$ as there exists a write event in the certificate which maps to the promise message and in this case $mo'' = mo'$ and $S'' = S'$, $sc'' = sc'$, $spo'' = spo'$, $srf'' = srf'$, $seco'' = seco'$ hold.
1. From Equation (ii) we know that \( G'' \sim_{\{i\}} \langle TS[i \mapsto T S''], S'', M'' \rangle \). Hence \( G'' \) is consistent.

2. From Equation (i) we know that
\[
\forall j. TS'(j).\sigma = (\mathbb{P}(j), labels(sequence_{spo'}(S'_j))) \text{ holds.}
\]
Hence \( \forall j. TS'(j).\sigma = (\mathbb{P}(j), labels(sequence_{spo}(S''_j))) \) also holds since \( S'' = S' \).

3. From Equation (i) we know \( G'' \sim_{\{i\}} \langle TS'[i \mapsto TS''], S'', M'' \rangle \). We also know that \( M'' = M' \) holds. Hence whenever \( \mathbb{W}''(e) = m \) then \( e.\text{loc} = m.\text{loc} \) and \( e.\text{wval} = m.\text{wval} \).

4. From Equation (i) we know \( G'' \sim_{\{i\}} \langle TS'[i \mapsto TS''], S'', M'' \rangle \). Hence the following also holds. \( \forall j \in (T \setminus \{i\}). \forall e \in (S''_0 \cup S'_j). \exists e.e. TS'(j).P \subseteq \{\mathbb{W}'(e') | (e, e') \in G''.\mathbb{P} \} \).

It implies
\[
\forall j \in (T \setminus \{i\}). \forall e \in (S''_0 \cup S'_j). TS'(j).P \subseteq \{\mathbb{W}''(e') | (e, e') \in G''.\mathbb{P} \} \tag{a}
\]

In thread \( i \) events in \( (S''_0 \cup S'_j) \) in \( G'' \) has \( G''.\mathbb{P} \)-following events \( e' \) corresponding to the certificate of outstanding promises. Hence \( \forall e \in (S''_0 \cup S'_j). TS'(i).P \subseteq \{\mathbb{W}'(e') | (e, e') \in G'.\mathbb{P} \} \).

It implies
\[
\forall e \in (S''_0 \cup S'_j). TS'(i).P \subseteq \{\mathbb{W}'(e') | (e, e') \in G'.\mathbb{P} \} \tag{b}
\]

Thus considering Equation (a), Equation (b) the following also holds
\[
\forall j \in T. \forall e \in (S''_0 \cup S'_j). TS'(j).P \subseteq \{\mathbb{W}''(e') | (e, e') \in G''.\mathbb{P} \}
\]

Thus the condition is satisfied between \( G'' \) and \( MS' \).

5. From Equation (i) we know
\[
\forall i, x. TS'(i).V(x) = \max\{\mathbb{W}(e).ts | e \in \text{dom}([\mathbb{W}]; G'.jf'; shb''; sc''; shb''; [S'_i])\}
\]

We know that \( G'.\mathbb{P} \subseteq G''.\mathbb{P} \), \( G'.jf \subseteq G''.jf \), \( G'.ew \subseteq G''.ew \).

Hence from the definitions following holds:
\[
TS'(i).V(x) = \max\{\mathbb{W}''(e).ts | e \in \text{dom}([\mathbb{W}]; G''.jf''; shb''; sc''; shb''; [S''_i])\}
\]

6. From Equation (ii) we already know \( (shb''; sc''; seco'') \) is irreflexive.

7. From Equation (ii) we already know \( [G''.U \cap S'']; (sfr''; smo'') \) \( = \emptyset \) holds.
A. Proving Simulation of Promising Semantics by WEAKEST

8. From Equation (i) we know \([G'.F_{sc}] \subseteq \text{shb'} \cup \text{shb'} \cup \text{seco}; \text{shb'} \cup \text{seco}; \text{shb'}; [G'.F_{sc}] \subseteq \text{sc'}.\)

   From Equation (ii) we know \([G''.F_{sc}] \subseteq \text{shb''} \cup \text{shb''} \cup \text{seco''}; \text{shb''}; [G''.F_{sc}] \subseteq \text{sc''}.\)

   However, we know \(\text{sc''} = \text{sc'}, G''.F_{sc} = G'.F_{sc}, \) and \(S'' = S'.\)

   Hence \([G''.F_{sc}] \cup \text{shb''}; \text{shb''} \cup \text{seco''}; \text{shb''}; [G''.F_{sc}] \subseteq \text{sc'}.\)

9. From Equation (i) we know \(\text{Behavior}(MS') = \text{Behavior}(G', W', S').\)

   From Equation (ii) we know \(\text{Behavior}(MS'') = \text{Behavior}(G'', W'', S'').\)

   However, \(\text{Behavior}(MS'') = \text{Behavior}(MS')\)

   and as a result, \(\text{Behavior}(MS') = \text{Behavior}(G', W', S').\)

As a result, \(G'' \sim MS'\) holds.

Case Promise step:

From \(G \sim MS, \) we get \(G \sim_{\{i\}} MS.\)

Also let \(MS \xrightarrow{\text{op}_i} MS'\) holds where \(\text{op} = \text{promise}(m)\) in the thread \(i.\)

We show: \(\exists G'. G \rightarrow^* G' \land G' \sim_{\{i\}} MS'\)

In this case \(TS' = TS[i \mapsto TS'], \) and \(M' = M \cup \{m\}, \) and we take \(G' = G.\)

Thus it remains to show that \(G \sim_{\{i\}} MS'.\)

We take \(W' = \emptyset.\)

As a result \(mo' = mo\) and \(S' = S, sc' = sc, spo' = spo, srf' = srf, seco' = seco\) hold.

1. From \(G \sim MS\) we know \(G\) is consistent and hence \(G'\) is also consistent.

2. From \(G' \sim_{\{i\}} MS'\) we know that \(\forall j \neq i. TS'(j).\sigma = \langle \mathcal{P}(j), \text{labels}(\text{sequence}_{spo}(S'_j)) \rangle\)

   holds.

   Hence from the definitions \(\forall j \neq i. TS'(j).\sigma = \langle \mathcal{P}(j), \text{labels}(\text{sequence}_{spo}(S'_j)) \rangle\) also holds.

   For \(j = i, TS'(i).\sigma = \langle \mathcal{P}(i), \text{labels}(\text{sequence}_{spo}(S'_i)) \rangle\) holds.

   It implies, \(TS'(i).\sigma = \langle \mathcal{P}(i), \text{labels}(\text{sequence}_{spo}(S_i)) \rangle\) also holds.

   Hence \(\forall j. TS'(i).\sigma = \langle \mathcal{P}(i), \text{labels}(\text{sequence}_{spo}(S_i)) \rangle\) holds.

   Thus the relation is preserved between \(G\) and \(MS'.\)

3. From \(G \sim MS\) we know whenever \(W(m) = e\) then \(e.\text{loc} = m.\text{loc}\) and \(e.\text{wval} = m.\text{wval}\)

   holds. Since \(W' = W,\) the same also holds for \(W'.\)

4. We know \(\forall j \in (T \setminus \{i\}), \forall e \in (S'_0 \cup S'_j). TS'(j).P \subseteq \{W'(e') \mid (e, e') \in G'.po\}.\)

   Hence from the definitions \(\forall j \in (T \setminus \{i\}), \forall e \in (S_0 \cup S_j). TS'(j).P \subseteq \{W(e) \mid (e, e) \in G.poe\}\) holds.
5. From $G \sim_{\{i}\text{MS}}$ we know
\[ \forall j \neq i. TS(j).V(\ell) = \max\{W(e).ts \mid e \in \text{dom}([W_{\ell}]; G.jf^2; \text{shb}^2; \text{sc}^2; \text{shb}^2; [S_j])\} \]
Since $G' = G$, $\mathcal{W}' = \mathcal{W}$, and $TS' = TS[i \mapsto TS']$ the following also holds.
\[ \forall j \neq i. TS'(j).V(\ell) = \max\{W'(e).ts \mid e \in \text{dom}([W'_{\ell}]; G.jf^2; \text{shb}^2; \text{sc}^2; \text{shb}^2; [S_j])\} \]

6. From $G \sim_{\{i\}} MS$ we know $[G.\mathcal{F}_{sc}]; \text{shb} \cup \text{shb}; \text{sec0}; \text{shb}; [G.\mathcal{F}_{sc}] \subseteq \text{sc}$ holds.
We know $G'.\mathcal{F}_{sc} = G.\mathcal{F}_{sc}$, $\text{shb}' = \text{shb}$, $\text{sec0}' = \text{sec0}$, and $\text{sc}' = \text{sc}$.
Hence, $[G'.\mathcal{F}_{sc}]; \text{shb}' \cup \text{shb}; \text{sec0}; \text{shb}; [G'.\mathcal{F}_{sc}] \subseteq \text{sc}'$ also holds.

7. From $G \sim_{\{i\}} MS$ we know $(\text{shb}; \text{sec0}')$ is irreflexive.
From the definition $\text{shb}' = \text{shb}$ and $\text{sec0}' = \text{sec0}$ hold.
Hence $(\text{shb}'; \text{sec0}')$ is irreflexive.

8. From $G \sim_{\{i\}} MS$ we know $[G.U \cap S]; (\text{sfr}; \text{smo}) = \emptyset$ holds.
We also know $\text{sfr}' = \text{sfr}$ and $\text{smo}' = \text{smo}$, $S' = S$, and $G.U \subseteq G'.U$.
Hence $[G'.U \cap S']; (\text{sfr}'; \text{smo}') = \emptyset$ also holds.

9. From $G \sim_{\{i\}} MS$ we know Behavior$(MS) = \text{Behavior}(G, \mathcal{W}, S)$. We also know that $S' = S$ and $G' = G$.
Now following the definitions of $MS'$ and $G'$, we get Behavior$(MS) = \text{Behavior}(MS')$
and Behavior$(G, \mathcal{W}, S) = \text{Behavior}(G', \mathcal{W}', S')$.
Hence Behavior$(MS') = \text{Behavior}(G', \mathcal{W}', S')$ holds.

Thus $G' \sim_{\{i\}} MS'$ holds.

Subcase Certificate step following the promise step:
From $G' \sim MS'$ we have $G' \sim_{\{i\}} MS'$ and also the following holds.
\[ \exists G''. G' \rightarrow^* G'' \land G'' \sim_{\{i\}} MS'' = \langle TS'[i \mapsto TS''], M'' \rangle \]
It remains to show $G'' \sim MS'$
We know that $TS'' = TS'$. Moreover a certificate does not create any new message and hence $M'' = M'$.
We take $S'' = S'$, and $\mathcal{W}'' = \mathcal{W}'[e' \mapsto m]$ where $e'.\text{loc} = m.\text{loc}$, $e'.\text{wval} = m.\text{wval}$.
As a result, $\text{mo}' \subseteq \text{mo}'$, and $S'' = S'$, $\text{sc}' = \text{sc}'$.
However, $e' \notin S''$ and hence $\text{smo}'' = \text{smo}'$.

1. We know that $G'' \sim_{\{i\}} MS''$. Hence $G''$ is consistent.
A. Proving Simulation of Promising Semantics by WEAKEST

2. From $G' \sim MS'$ we know that
   \[
   \forall j. TS'(j).\sigma = \langle \mathbb{P}(j), \text{labels}(\text{sequence}_{spo'}(S'_j)) \rangle \text{ holds.}
   \]
   We also know that $S' = S'$ and $TS' = TS'$.
   Hence $\forall j. TS'(j).\sigma = \langle \mathbb{P}(j), \text{labels}(\text{sequence}_{spo'}(S''_j)) \rangle$ also holds.

3. We know $G' \sim_{[i]} MS'$. We also know that $M'' = M'$ holds.
   Hence whenever $\mathbb{W}''(e) = m$, then $e.\text{loc} = m.\text{loc}$ and $e.\text{wval} = m.\text{wval}$ holds.

4. We know $G' \sim_{[i]} \langle TS[i \mapsto TS'], S', M' \rangle$. Hence the following also holds.
   \[
   \forall j \in (T \setminus \{i\}). \forall e \in (S'_0 \cup S'_j). TS'(j).P \subseteq \{ \mathbb{W}'(e') \mid (e, e') \in G'.\text{po} \}.
   \]
   It implies
   \[
   \forall j \in (T \setminus \{i\}). \forall e \in (S''_0 \cup S''_j). TS'(j).P \subseteq \{ \mathbb{W}''(e') \mid (e, e') \in G''.\text{po} \}
   \]
   (c)
   In thread $i$ events in $(S'_0 \cup S'_j)$ in $G'$ has $G'$-po-following events $e'$ corresponding to the certificate of outstanding promises.
   Hence $\forall e \in (S'_0 \cup S'_j). TS'(i).P \subseteq \{ \mathbb{W}'(e') \mid (e, e') \in G'.\text{po} \}$.
   It implies
   \[
   \forall e \in (S''_0 \cup S''_j). TS'(i).P \subseteq \{ \mathbb{W}''(e') \mid (e, e') \in G''.\text{po} \}
   \]
   (d)
   Thus considering Equation (c), Equation (d) the following also holds
   \[
   \forall j \in T. \forall e \in (S''_0 \cup S''_j). TS'(j).P \subseteq \{ \mathbb{W}''(e') \mid (e, e') \in G''.\text{po} \}
   \]
   Thus the condition is satisfied between $G''$ and $MS'$.

5. From $G' \sim_{[i]} MS'$ We know
   \[
   TS'(i).V(\ell) = \max\{ \mathbb{W}'(e).ts \mid e \in \text{dom}(\mathbb{W}'_i; G'.jf^\ell; shb^{G'}_i; sc^\ell; shb'^G_i; [S'_i]) \}
   \]
   We know that $G'.E \subseteq G''.E$, $G'.\text{po} \subseteq G''.\text{po}$, $G'.jf \subseteq G''.jf$, $G'.ew \subseteq G''.ew$, $TS'' = TS'$, $S'' = S'$, and $\mathbb{W}'' = \mathbb{W}'[e' \mapsto m]$.
   Hence from the definitions following holds:
   \[
   TS'(i).V(x) = \max\{ \mathbb{W}''(e).ts \mid e \in \text{dom}(\mathbb{W}''_i; G''.jf^\ell; shb''^{G''}_i; sc''^\ell; shb''^{G''}_i; [S''_i]) \}
   \]

6. We know $(shb'; seco'^G)$ is irreflexive.
   From the definition $shb'' = shb'$ and $seco'' = seco'$.
   Hence $(shb''; seco''^G)$ is irreflexive.
7. From \( G' \sim_{\{i\}} MS' \) we know \([G'.U \cap S']; (sfr'; smo') = \emptyset \) holds.
   We also know \( sfr'' = sfr' \) and \( smo'' = smo' \), \( S'' = S' \), and \( G'.U \subseteq G''.U \).
   Hence \([G''.U \cap S'']; (sfr''; smo'') = \emptyset \) also holds.

8. We know \( S'' = S', mo' \subseteq mo'', sc'' = sc' \).
   We also know that \([G'.F_{sc}]; shb' \cup shb'; seco'; shb'; [G'.F_{sc}] \subseteq sc' \) holds.
   Hence, \([G''.F_{sc}]; shb'' \cup shb'''; seco''; shb''; [G''.F_{sc}] \subseteq sc'' \) also holds.

9. From \( G' \sim_{\{i\}} MS' \) we know \( \text{Behavior}(MS') = \text{Behavior}(G', \mathbb{W}', S') \).

   From \( G'' \sim_{\{i\}} MS'' \) we know \( \text{Behavior}(MS'') = \text{Behavior}(G'', \mathbb{W}'', S'') \).
   From definitions \( \text{Behavior}(MS'') = \text{Behavior}(MS') \)
   and \( \text{Behavior}(G'', \mathbb{W}'', S'') = \text{Behavior}(G', \mathbb{W}', S') \) holds.
   Hence \( \text{Behavior}(MS') = \text{Behavior}(G'', \mathbb{W}'', S'') \) holds.

Hence \( G'' \sim MS' \) holds. \( \Box \)

Finally we restate and prove Theorem 1.

**Theorem 1.** For a program \( P \), \( \text{Behavior}_{PS}(P) \subseteq \text{Behavior}_{\text{WEAKEST}}(P) \).

Formal statement:

\[
\begin{align*}
\forall P. \forall MS. (MS_{\text{init}}(P) \rightarrow^* MS \land MS \not\sigma). & \exists G, X. G_{\text{init}} \rightarrow_{P,\text{WEAKEST}}^* G \land X \in \text{ex}_{\text{WEAKEST}}(G). \\
\land \text{Behavior}(MS) &= \text{Behavior}(X)
\end{align*}
\]

**Proof.** **Step 1.** Given a program \( P \), from Lemma 2 we show that using the simulation relation in Definition 6, we can follow the promise machine steps and for a promise machine state state MS we can construct an WEAKEST event structure \( G \), that is, \( G_{\text{init}} \rightarrow_{P,\text{WEAKEST}}^* G \).

**Step 2.** Now we extract a consistent execution \( X \) from \( G \) where \( X \in \text{ex}_{\text{WEAKEST}}(G) \), such that \( \text{Behavior}(MS) = \text{Behavior}(X) \).

Given the event structure \( G \) along with \( S \) and related sets, the execution \( X = (E, po, rf, mo) \) is as follows.

- \( X.E = S \),
- \( X.po = spo \),
- \( X.rf = srf \), and
- \( X.mo = smo \)

Note that the events in \( X.E \) is conflict-free as \( S \) is conflict-free in \( G \).

Now we check whether execution \( X \) is consistent.
A. Proving Simulation of Promising Semantics by \textit{WEAKEST}

- from the definitions of \texttt{spo}, \texttt{srf}, \texttt{smo}, we know
  \[ X.\text{po} \subseteq (S \times S), X.\text{rf} \subseteq (S \times S), \text{and } X.\text{mo} \subseteq (S \times S). \]
  Hence \( X \) is (Well-formed).

- From the definition, we know \texttt{smo} is total as the order on the timestamps on the same location is total in the promise machine.
  Hence \( X.\text{mo} \) is total and \((\text{total-MO})\) holds in \( X \).

- From the construction of \( G \) we know that \texttt{shb} \texttt{seco} is irreflexive.
  Hence \( (X.\text{hb}_{C11}; X.\text{eco}^{-}) \) is irreflexive and \((\text{Coherence})\) holds in \( G \).

- From the definition we know that \( [G.U \cap S]; (sfr; smo) = \emptyset \) holds. From the definition we know that \( X.U = (G.U \cap S), X.fr = sfr \), and also \( X.\text{mo} = smo \) holds.
  Hence \( [X.U]; (X.fr; X.mo) = \emptyset \) hold and \( X \) preserves \((\text{Atomicity})\).

- From the simulation relation in the construction we know that \texttt{sc} is total in \( G \) and
  \[ [G.\mathcal{F}_{\text{sc}}]; \texttt{shb} \cup \texttt{shb}; \texttt{seco}; \texttt{shb}; [G.\mathcal{F}_{\text{sc}}] \subseteq \texttt{sc} \]
  holds.
  Hence \( [G.\mathcal{F}_{\text{sc}}]; \texttt{shb} \cup \texttt{shb}; \texttt{seco}; \texttt{shb}; [G.\mathcal{F}_{\text{sc}}] \) is irreflexive.
  From definition we know that \( X.\mathcal{F}_{\text{sc}} = G.\mathcal{F}_{\text{sc}}, X.\text{hb}_{C11} = \text{shb} \), and \( X.\text{eco} = \text{seco} \).
  As a result, \( X.\text{psc}_F = [X.\mathcal{F}_{\text{sc}}]; X.\text{hb}_{C11} \cup X.\text{hb}_{C11}; X.\text{eco}; X.\text{hb}_{C11}; [X.\mathcal{F}_{\text{sc}}] \text{ is irreflexive.} \)
  Note that \( X \) does not have any SC memory access and hence \( X.\text{psc}_{\text{base}} = \emptyset \).
  Hence \( X \) preserves \((\text{SC})\).

Thus \( X \) is consistent and hence \( X \in \text{ex}_{\text{WEAKEST}}(G) \).

**Step 3.** From the construction we know that \( \text{Behavior}(MS) = \text{Behavior}(G, W, S) \).
Hence from the definitions \( \text{Behavior}(MS) = \text{Behavior}(X) \).
Thus considering step 1, 2, 3 the theorem holds.
B. Causality Test Cases

Figure B.1.: Case 1. Allowed \( r_1 = r_2 = 1 \).

Figure B.2.: Case 2. Allowed \( r_1 = r_2 = r_3 = 1 \).

Figure B.3.: Case 3. Allowed \( r_1 = r_2 = r_3 = 1 \).
B. Causality Test Cases

\[ r_1 = X; \quad r_2 = Y; \quad Y = r_1; \quad X = r_2; \]

Figure B.4.: Case 4. Forbidden \( r_1 \iff r_2 \iff 1 \).

\[ r_1 = X; \quad r_2 = Y; \quad r_3 = Z; \quad X = r_2; \quad Y = r_1; \quad Z = 1; \]

Figure B.5.: Case 5. Forbidden \( r_1 \iff r_2 \iff 1, r_3 \iff 0 \). However, a sequence of transformations result this behavior.

\[ r_1 = A; \quad r_2 = B; \quad if(r_2 == 1) \quad A = 1; \quad if(r_2 == 0) \quad A = 1; \]

Figure B.6.: Case 6. Allowed \( r_1 \iff r_2 \iff 1 \).

\[ r_1 = Z; \quad r_3 = Y; \quad r_2 = X; \quad Z = r_3; \quad Y = r_2; \quad X = 1; \]

Figure B.7.: Case 7. Allowed \( r_1 \iff r_2 \iff r_3 \iff 1 \).
Figure B.8.: Case 8. Allowed $r_1 == r_2 == 1$.

Figure B.9.: Case 9. Allowed $r_1 == r_2 == 1$.

Figure B.10.: Case 9a. Allowed $r_1 == r_2 == 1$.

Figure B.11.: Case 10. Forbidden $r_1 == r_2 == 1, r_3 == 0$. Same event structure as Figure B.5. Similar to test case 5, a sequence of transformations result in this behavior.
B. Causality Test Cases

\[ r_1 = Z; \quad W = r_1; \quad r_2 = X; \quad Y = r_2; \]
\[ r_3 = Z; \quad X = 1; \]
\[ r_4 = W; \]

Figure B.12.: Case 11. Allowed \( r_1 == r_2 == r_3 == r_4 == 1. \)

\[ X = Y = 0; a[0] = 1; a[1] = 2; \]
\[ r_1 = X; \quad a[r_1] = 0; \quad r_2 = a[0]; \quad Y = r_2; \]

Figure B.13.: Case 12. Forbids \( r_1 == r_2 == r_3 == 1. \)

\[ r_1 = X; \quad if(r_1 == 1) \]
\[ Y = 1; \quad if(r_2 == 1) \]
\[ Y = 1; \quad X = 1; \]

Figure B.14.: Case 13. Forbids \( r_1 == r_2 == 1. \)

\[ r_1 = A; \quad if(r_1 == 0) \]
\[ Y_{sc} = 1; \quad Y = Y_{sc}; \quad r_2 = Y_{sc}; \quad r_3 = B; \]
\[ B = 1; \quad do\{ \]
\[ } while(r_2 + r_3 == 0); \]
\[ A = 1; \]

Figure B.15.: Case 14. Forbids \( r_1 = r_3 = 1; r_2 = 0. \) In [45] \( Y \) is ‘volatile’ in Java. We map Java volatile to SC in C11 as the reordering rules are same.
\( r_0 = X_{sc}; \)
\[
\text{if}(r_0 == 1) \quad r_1 = A; \\
\text{else} \quad r_1 = 0; \\
\text{if}(r_1 == 0) \quad Y_{sc} = 1; \\
\text{else} \quad B = 1; \\
\]
\[
\text{do} \begin{cases} \\
  r_2 = Y_{sc}; \quad r_3 = B; \\
 \end{cases} \text{while}(r_2 + r_3 == 0); \quad X_{sc} = 1; \\
\]

Figure B.16.: Case 15. Forbids \( r_1 == r_3 == 1; r_2 == 0 \). In [45] \( X \) and \( Y \) are ‘volatile’ in Java. We map Java volatile to SC in C11 as the reordering rules are same.

\[ [A = B = X = Y = 0] \]

Figure B.17.: Case 16. Behavior in question: \( r_1 = 2, r_2 = 1 \). This is allowed in Manson et al. [45]. The behavior is allowed in basic event structure and in extracted execution as they do not enforce coherence. The \textsc{weakest} model constructs an event structure with these events but disallows the incoherent behavior in the extracted execution. The \textsc{weakestmo} model does not accommodate all these events together in any event structure and in consequence disallows the incoherent behavior in the extracted execution.

\[ [X = Y = 0] \]

65
B. Causality Test Cases

\[ r_3 = X; \]
if \((r_3 \neq 4)\)
\[ X = 4; \]
\[ r_2 = Y; \]
\[ X = r_2; \]
\[ r_1 = X; \]
\[ Y = r_1; \]
\[ r_3 = X; \]
if \((r_3 = 0)\)
\[ X = 4; \]
\[ r_2 = Y; \]
\[ X = r_2; \]
\[ r_1 = X; \]
\[ Y = r_1; \]

\[ A = B = X = Y = 0 \]

Figure B.18.: Case 17 and 18. Allows \( r_1 == r_2 == r_3 == 4 \).

\[ r_1 = X; \]
\[ Y = r_1; \]
\[ r_2 = Y; \]
\[ X = r_2; \]
\[ r_3 = X; \]
if \((r_3 \neq 4)\)
\[ X = 4; \]
\[ r_2 = Y; \]
\[ X = r_2; \]
\[ r_1 = X; \]
\[ Y = r_1; \]
\[ r_3 = X; \]
if \((r_3 = 0)\)
\[ X = 4; \]

\[ A = B = X = Y = 0 \]

Figure B.19.: Case 19 and 20. Event Structure \textbf{Forbids} \( r_1 == r_2 == r_3 == 4 \).
B.1. Allowing Forbidden Behaviors

Now we see certain behaviors which are disallowed by Manson et al. [45] and our proposed scheme but are possible after a number of program transformations.

**Testcase 5** The \( r_1 == r_2 == 1, r_3 == 0 \) outcome is possible after a sequence of transformations as follows.

\[
\begin{align*}
  r_1 &= X; \\
  Y &= r_1; \\
  r_2 &= Y; \\
  X &= r_2; \\
  r_3 &= Z; \\
  X &= r_3; \\
  Z &= 1;
\end{align*}
\]

\[
\sim
\begin{align*}
  r_1 &= X; \\
  Y &= r_1; \\
  r_2 &= Y; \\
  \text{if}(r_2 == 1) X &= 1; \text{else} X &= r_2; \\
  r_3 &= Z; \\
  X &= r_3; \\
  Z &= 1;
\end{align*}
\]

\[
\sim
\begin{align*}
  r_1 &= X; \\
  Y &= r_1; \\
  r_2 &= Y; \\
  \text{if}(r_2 == 1) X &= 1; \text{else} X &= r_2; \\
  \{r_3 = Z; X = r_3;\} \parallel \{Z = 1;\}
\end{align*}
\]

\[
\sim
\begin{align*}
  r_1 &= X; \\
  Y &= r_1; \\
  r_2 &= Y; \\
  \text{if}(r_2 == 1) \{\text{X} = 1; \text{r}_3 = Z; \text{X} = \text{r}_3; \text{Z} = 1;\} \\
  \text{else} \{\text{X} = \text{r}_2; \text{Z} = 1; \text{r}_3 = \text{Z}; \text{X} = \text{r}_3;\}
\end{align*}
\]

\[
\sim
\begin{align*}
  r_1 &= X; \\
  Y &= r_1; \\
  r_2 &= Y; \\
  \text{if}(r_2 == 1) \{\text{X} = 1; \text{r}_3 = Z; \text{X} = \text{r}_3; \text{Z} = 1;\} \\
  \text{else} \{\text{X} = \text{r}_2; \text{Z} = 1; \text{r}_3 = 1; \text{X} = 1;\}
\end{align*}
\]

\[
\sim
\begin{align*}
  a : r_1 &= X; \\
  b : Y &= r_1; \\
  c : X &= 1; \\
  d : r_2 &= Y; \\
  \text{if}(r_2 == 1) \{e : r_3 = Z; X = r_3; Z = 1;\} \\
  \text{else} \{Z = 1; r_3 = 1;\}
\end{align*}
\]

Now it is possible to have an interleaving \(c, a, b, d, e\) which results in \( r_1 == r_2 == 1, r_3 == 0 \).

**Testcase 10** Similar to test case 5 the \( r_1 == r_2 == 1, r_3 == 0 \) outcome is possible after a sequence of transformations as follows.

\[
\begin{align*}
  r_1 &= X; \\
  \text{if}(r_1 == 1) Y &= 1; \\
  r_2 &= Y; \\
  \text{if}(r_2 == 1) X &= 1; \\
  r_3 &= Z; \\
  \text{if}(r_3 == 1) X &= 1; \\
  Z &= 1; \\
\end{align*}
\]

\[
\sim
\begin{align*}
  r_1 &= X; \\
  \text{if}(r_1 == 1) Y &= 1; \\
  r_2 &= Y; \\
  \text{if}(r_2 == 1) X &= 1; \\
  r_3 &= Z; \\
  \text{if}(r_3 == 1) X &= 1; \\
  Z &= 1; \\
\end{align*}
\]
B. Causality Test Cases

\[
\begin{align*}
  r_1 &= X; \\
  \text{if}(r_1 == 1) &\quad \begin{cases} 
    r_2 &= Y; \\
    \text{if}(r_2 == 1) &\quad \begin{cases} 
      r_3 &= Z; \\
      \text{if}(r_3 == 1) &\quad X = 1; \\
      \text{else} &\quad X = 0; \\
    \end{cases} \\
  \end{cases} \\
  Y &= 1; \\
\end{align*}
\]

\[
\begin{align*}
  r_2 &= Y; \\
  \text{if}(r_2 == 1) &\quad \begin{cases} 
    r_3 &= Z; \\
    \text{if}(r_3 == 1) &\quad X = 1; \\
    \end{cases} \\
  \end{cases} \\
\]

\[
\begin{align*}
  r_3 &= Z; \\
  \text{if}(r_3 == 1) &\quad X = 1; \\
  \end{cases} \\
\]

Now we can have an interleaving \( d, a, b, c, e, f \) which results in \( r_1 == r_2 == 1, r_3 == 0 \).
C. Monotonicity of WEAKESTMO

The Weaken transformation is as follows:

- \( \tau \cdot Ld_o(x, v) \cdot \tau' \xrightarrow{\text{Weaken}} \tau \cdot Ld'_o(x, v) \cdot \tau' \) where \( o' \subseteq o \)
- \( \tau \cdot St_o(x, v) \cdot \tau' \xrightarrow{\text{Weaken}} \tau \cdot St'_o(x, v) \cdot \tau' \) where \( o' \subseteq o \)
- \( \tau \cdot U_o(x, v, v') \cdot \tau' \xrightarrow{\text{Weaken}} \tau \cdot U'_o(x, v, v') \cdot \tau' \) where \( o' \subseteq o \)
- \( \tau \cdot F_o \cdot \tau' \xrightarrow{\text{Weaken}} \tau \cdot F'_o \cdot \tau' \) where \( o' \subseteq o \)

We prove that the WEAKESTMO is a monotonic memory model.

Theorem 9. Given a program \( P_{\text{src}} \) if we Weaken a program \( P_{\text{src}} \) to \( P_{\text{tgt}} \) then
1. For each consistent event structure of \( P_{\text{src}} \) there exists a consistent event structure of \( P_{\text{tgt}} \).
2. For each consistent execution extracted from a consistent event structure of \( P_{\text{src}} \) there exists a consistent execution extracted from a consistent event structure of \( P_{\text{tgt}} \).

Formal statement

\[
\forall P_{\text{src}}: \text{Weaken}(P_{\text{src}}, P_{\text{tgt}}) \implies \\
\forall G_{\text{src}}, G_{\text{init}} \rightarrow P_{\text{src}}, \text{WEAKESTMO}^* G_{\text{src}}, \exists G_{\text{tgt}}, G_{\text{init}} \rightarrow P_{\text{tgt}}, \text{WEAKESTMO}^* G_{\text{tgt}} \land \\
\forall X_s \in \text{ex}_{\text{WEAKESTMO}}(G_{\text{src}}), \exists X_t \in \text{ex}_{\text{WEAKESTMO}}(G_{\text{tgt}}). \text{Behavior}(X_t) = \text{Behavior}(X_s)
\]

Proof. (1) Given a target event structure \( G_{\text{init}} \rightarrow P_{\text{src}}, \text{WEAKESTMO}^* G_{\text{src}}, \) we follow the construction steps of \( G_{\text{src}} \) and construct \( G_{\text{tgt}} \). In this construction, we can follow the write steps similar to that of \( G_{\text{tgt}} \). We can also follow the \( G_{\text{src}} \) fence step unless the fence is deleted. Hence we can append the reads with same labels by justifying from same writes compared to that of \( G_{\text{src}} \). Thus, \( G_{\text{tgt}} \cdot E \subseteq G_{\text{src}} \cdot E, G_{\text{tgt}} \cdot RW'_o \equiv G_{\text{tgt}} \cdot RW_o, G_{\text{tgt}} \cdot po \subseteq G_{\text{src}} \cdot po, G_{\text{tgt}} \cdot jf = G_{\text{src}} \cdot jf, \) and \( G_{\text{tgt}} \cdot ew = G_{\text{src}} \cdot ew \). While constructing \( G_{\text{tgt}} \) from \( G_{\text{src}} \), essentially we remove po edges to/from fences along with certain sw edges due to the removal of fences or replacing the Rel or Acq events with events with weaker or same memory order. As a result, we in turn remove certain hb relations and the relations between the SC accesses.

As a result, the \( G_{\text{tgt}} \) is less restrictive than \( G_{\text{src}} \) in terms of the relations involved in the WEAKEST or WEAKESTMO consistency conditions and \( G_{\text{tgt}} \) remains consistent.

(2) For each execution \( X_s \in \text{ex}_{\text{WEAKESTMO}}(G_{\text{src}}) \), we find an execution \( X_t \) such that \( X_t \cdot E \subseteq X_s \cdot E, X_t \cdot RW'_o \equiv X_s \cdot RW_o, X_t \cdot po \subseteq X_s \cdot po, X_t \cdot rf = X_s \cdot rf, X_t \cdot mo = X_s \cdot mo \).
C. Monotonicity of \textsc{weakestmo}

Similar to the event structures, the $X_t$ is less restrictive than $X_s$ in terms of the relations involved in the execution consistency conditions. Hence $X_t$ remains consistent and $X_t \in \text{ex}_{\textsc{weakestmo}}(G_{tgt})$ holds. Moreover, in this case $\text{Behavior}(X_s) = \text{Behavior}(X_t)$ holds following the definitions of $X_s$ and $X_t$.

\textbf{Remark 3.} Consider we append a read $r$ to consistent event structure $G$ by justifying from a write $w \in G.W$ from $(G'.\text{hb} \cup G'.\text{jf})$-prefix and create $G'$ such that $G'$ is consistent when $\exists W(G',w,r)$ holds where

$$\exists W(G',w,r) \triangleq (w,r) \in (G'.\text{jf}^2; G'.\text{hb}^2 \setminus G'.\text{ecf}) \land \exists w'. \exists W(G',w',r) \land G'.\text{mo}(w,w')$$

Note that there exists some write $w \in G.W$ such that $\exists W(G,w,r)$ holds as all locations are initialized.
D. Proofs of Correctness of Reorderings

We start with definitions and a lemma on $h_b$ in the WEAKEMO model.

We first define unique predecessor and unique successor.

**Definition 9.** Unique-pred($R, a, b$) $\triangleq R(a, b) \land \forall c. \ G.R(c, b) \implies c = a$

**Definition 10.** Unique-succ($R, a, b$) $\triangleq R(a, b) \land \forall c. \ G.R(a, c) \implies c = b$.

We derive the following lemma.

**Lemma 8.** if Unique-pred($R, b, a$) and Unique-succ($R, b, a$) holds then ($R \setminus \{(b, a)\} \cup \{(a, b)\})^+ \subseteq R^+ \setminus \{(b, a)\} \cup \{(a, b)\}$ also holds.

**Proof.** We assume Unique-pred($R, b, a$) and Unique-succ($R, b, a$) holds.

Now we show ($R \setminus \{(b, a)\} \cup \{(a, b)\})^+ \subseteq R^+ \setminus \{(b, a)\} \cup \{(a, b)\}$.

We prove by induction on transitive closure.

**Base Case:** ($R \setminus \{(b, a)\} \cup \{(a, b)\}) \subseteq ($R^+ \setminus \{(b, a)\} \cup \{(a, b)\}$).

The base case holds trivially by monotonicity.

The induction step:

**case 1.** ($R \setminus \{(b, a)\}) \circ (R^+ \setminus \{(b, a)\}) \subseteq (R^+ \setminus \{(b, a)\}) \cup \{(a, b)\}$.

It is sufficient to show:

(R \setminus \{(b, a)\}) \circ (R^+ \setminus \{(b, a)\}) \subseteq R^+ \setminus \{(b, a)\}

Therefore it is sufficient to show,

(R \setminus \{(b, a)\}) \circ (R^+ \setminus \{(b, a)\}) \subseteq R^+ \land (b, a) \notin (R \setminus \{(b, a)\}) \circ (R^+ \setminus \{(b, a)\})

Now

(i) By monotonicity we know that (R \setminus \{(b, a)\}) \circ (R^+ \setminus \{(b, a)\}) \subseteq R^+.

therefore it is sufficient to show

(ii) (b, a) \notin (R \setminus \{(b, a)\}) \circ (R^+ \setminus \{(a, b)\}).

Assume (b, a) \in (R \setminus \{(b, a)\}) \circ (R^+ \setminus \{(b, a)\}).

By unfolding the definition of \circ, it is sufficient to show
D. Proofs of Correctness of Reorderings

\[ \exists c. (b, c) \in (R \setminus \{(b, a)\}) \land (c, a) \in (R^+ \setminus \{(b, a)\}) \]

Assume \( \exists c. (b, c) \in R \setminus \{(b, a)\} \).

Therefore \((b, c) \in R \land c \neq a \land (c, a) \in R^+ \land c \neq b \).

From Unique-succ\((R, b, a)\) we know \( c = a \) which is a contradiction.

Hence \( \exists c. (b, c) \in (R \setminus \{(b, a)\}) \).

case 2. \((R \setminus \{(b, a)\}) \circ \{(a, b)\} \subseteq (R^+ \setminus \{(b, a)\}) \cup \{(a, b)\}) \).

We know Unique-pred\((R, a, b)\) holds and hence \( \exists a, b, c. R(b, a) \land R(c, a) \land b \neq c \).

Hence, \((R \setminus \{(b, a)\}) \circ \{(a, b)\} = \emptyset \).

As a result, \((R \setminus \{(b, a)\}) \circ \{(a, b)\} \subseteq (R^+ \setminus \{(b, a)\}) \cup \{(a, b)\}) \).

case 3. \{(a, b)\} \circ (R^+ \setminus \{(b, a)\}) \subseteq (R^+ \setminus \{(b, a)\}) \cup \{(a, b)\}) \).

We know \{(a, b)\} \circ R \setminus \{(b, a)\} = \emptyset \) because Unique-succ\((R, a, b)\) holds, that is, \( \exists a, b, c. R(a, b) \land R(a, c) \land b \neq c \).

As a result, \{(a, b)\} \circ R \setminus \{(b, a)\} \subseteq (R^+ \setminus \{(b, a)\}) \cup \{(a, b)\}) \).

case 4. \{(a, b)\} \circ \{(a, b)\} \subseteq (R^+ \setminus \{(b, a)\}) \cup \{(a, b)\}) \).

\{(a, b)\} \circ \{(a, b)\} = \emptyset \) and hence \{(a, b)\} \circ \{(a, b)\} \subseteq (R^+ \setminus \{(b, a)\}) \cup \{(a, b)\}) \).

\[ \square \]

Now we relate the happens-before relations between the source and target executions. The safe reorderings from Table 7.1 as follows:

\text{reord}(\mathbb{P}_{\text{src}}, \mathbb{P}_{\text{tgt}}) \) such that

\[ \mathbb{P}_{\text{tgt}}(i) \subseteq \mathbb{P}_{\text{src}}(i) \cup \{\tau \cdot \beta \cdot \tau' | \tau \cdot \alpha \cdot \tau' \in \mathbb{P}_{\text{src}}(i) \} \land \forall j \neq i. \mathbb{P}_{\text{tgt}}(j) = \mathbb{P}_{\text{src}}(j) \]

where \( \alpha = a-b, \beta = b-a \), and \( a, b \) are labels of shared memory accesses or fences..

**Lemma 9.** Suppose

1. \( \text{reord}(\mathbb{P}_{\text{src}}, \mathbb{P}_{\text{tgt}}) \) where the reordering is \( a; b \leadsto b; a \) and
2. \( X_s \in \text{ex_{weakestmo}}(G_{\text{src}}) \) where \( G_{\text{init}} \to G_{\text{src}, \text{weakestmo}^*} G_{\text{src}} \) and
3. \( X_t \in \text{ex_{weakestmo}}(G_{\text{tgt}}) \) where \( G_{\text{init}} \to G_{\text{tgt}, \text{weakestmo}^*} G_{\text{tgt}} \).

Then \( X_s \cdot \text{hb}_{C11} \subseteq X_t \cdot \text{hb}_{C11} \setminus \{(b, a)\} \cup \{(a, b)\} \).

**Proof.** We know \( X_s \cdot \text{po} = X_t \cdot \text{po} \setminus \{(b, a)\} \cup \{(a, b)\} \). Let \( R = (X_t \cdot \text{po} \cup R') \) where \( R' \) is some other relation independent of \( X_t \cdot \text{po} \). Hence from Lemma 8,

\[ (R \setminus \{(b, a)\}) \cup \{(a, b)\})^+ \subseteq (R^+ \setminus \{(b, a)\}) \cup \{(a, b)\}) \]

\[ \implies ((X_t \cdot \text{po} \cup R') \setminus \{(b, a)\}) \cup \{(a, b)\})^+ \subseteq ((X_t \cdot \text{po} \cup R') \setminus \{(b, a)\}) \cup \{(a, b)\}) \]

\[ \implies ((X_s \cdot \text{po} \setminus \{(b, a)\}) \cup \{(a, b)\}) \cup R')^+ \subseteq ((X_t \cdot \text{po} \cup R') \setminus \{(b, a)\}) \cup \{(a, b)\}) \]

\[ \implies (X_s \cdot \text{po} \cup R')^+ \subseteq ((X_t \cdot \text{po} \cup R') \setminus \{(b, a)\}) \cup \{(a, b)\}) \]

72
D.1. Reordering Theorem

\[ \implies \imm(X_s,po) \cup R' \subseteq ((\imm(X_t,po) \cup R') \setminus \{(b,a)\} \cup \{(a,b)\}) \]

since \((X_s,po \cup R') = (\imm(X_s,po) \cup R') + \)

\(= \imm(X_s,po) \cup R' + \)

\(= \{(b,a)\} \cup \{(a,b)\})\]

substituting \(R' = X_s,sw_{C11} = X_t,sw_{C11}\) we get

\((\imm(X_s,po) \cup X_s,sw_{C11})^+ \subseteq ((X_t,po \cup X_t,sw_{C11})^+ \setminus \{(b,a)\} \cup \{(a,b)\})\)

It implies \(X_s,hb_{C11} \subseteq (X_t,hb_{C11} \setminus \{(b,a)\} \cup \{(a,b)\})\)

as \(X_s,hb_{C11} = (\imm(X_s,po) \cup X_s,sw_{C11})^+ \) and \(X_t,hb_{C11} = (\imm(X_t,po) \cup X_t,sw_{C11})^+. \)

\[ \square \]

D.1. Reordering Theorem

We restate the definition of compilation correctness and the safe reordering theorem.

**Definition 8.** A transformation of program \(P_{src}\) in memory model \(M_{src}\) to program \(P_{tgt}\) in model \(M_{tgt}\) is **correct** if it does not introduce new behaviors:

i.e., \(\text{Behavior}_{M_{tgt}}(P_{tgt}) \subseteq \text{Behavior}_{M_{src}}(P_{src}).\)

**Theorem 6.** The safe reorderings in Table 7.1 are correct in both WEAKESTMO models.

The formal statement is as follows:

\[ \forall P_{src} \cdot \text{reord}(P_{src}, P_{tgt}) \implies \forall G_{tgt}, G_{init} \rightarrow_{tgt,WEAKESTMO^*} G_{tgt}, \exists G_{src}, G_{init} \rightarrow_{src,WEAKESTMO^*} G_{src} \land \]

\[ \forall X_t \in \text{ex}_{WEAKESTMO}(G_{tgt}), \exists X_s \in \text{ex}_{WEAKESTMO}(G_{src}). \text{Behavior}(X_t) = \text{Behavior}(X_s) \land X_t.Race \cap E_{NA} \neq \emptyset \implies X_s.Race \cap E_{NA} \neq \emptyset \]

To prove the theorem, given an extracted consistent target execution \(X_t \in \text{ex}_{WEAKESTMO}(G_{tgt})\) from a consistent target event structure \(G_{tgt}\), we construct a consistent source execution \(X_s\) from \(X_t\). Then we ensure that the behavior of the \(X_s\) and \(X_t\) are same and if \(X_t\) has undefined behavior due to data race then \(X_s\) also has undefined behavior due to data race. Finally, we show that the \(X_s \in \text{ex}_{WEAKESTMO}(G_{src})\) where \(G_{src}\) is a WEAKESTMO consistent source event structure.

**Proof.** In this proof we follow the above mentioned steps as follows.

**Source Execution Consistency.** From target execution \(X_t\) we get source execution \(X_s\) by reordering the respective events. Thus if \(\imm(X_t,po)(b,a)\) then \(\imm(X_t,po)(a,b)\) holds. We know, following the Lemma 9, \(X_s,hb \subseteq X_t \setminus \{(b,a)\} \cup \{(a,b)\}\), that is, \(X_s\) is more relaxed than \(X_t\). We also know that \(X_t\) is consistent. Hence the execution \(X_s\) is consistent.

**Same Behavior.** The behaviors of \(X_s\) and \(X_t\) are same. The reordering does not introduce any new mo relation in \(X_s\) and thus \(X_t.mo = X_s.mo\). Hence the behaviors of \(X_s\) and \(X_t\) are same.

**Race Preservation.**

following the Lemma 9, \(X_s,hb \subseteq X_t,hb \setminus \{(b,a)\} \cup \{(a,b)\}\). Hence if \(X_t\) is racy, then \(X_s\) is also racy. As a result, if the target execution has undefined behavior due to a data race, so does the source execution.

73
D. Proofs of Correctness of Reorderings

Source Event Structure Construction and Execution Extraction

It is left to show that we can construct a source event structure \( G_{\text{init}} \rightarrow_{\text{WEAKESTMO}} G_{\text{src}} \) such that execution \( X_s \) is an extracted execution from \( G_{\text{src}} \), that is, \( X_s \in \text{ex}_{\text{WEAKESTMO}}(G_{\text{src}}) \).

If \( (X_s, \text{po} \cup X_s, \text{rf})^+ \) is acyclic, then we follow the \( (X_s, \text{po} \cup X_s, \text{rf})^+ \) path to construct the source event structure and in this case \( G_{\text{src}} = X_s \). From the definitions we know that \( \text{WEAKESTMO} \) constraints are weaker than the execution constraints. Hence \( G_{\text{src}} \) is consistent as \( X_s \) is consistent. As a result, \( X_s \in \text{ex}_{\text{WEAKESTMO}}(G_{\text{src}}) \).

However, if \( X_s \) has \( (X_s, \text{po} \cup X_s, \text{rf})^+ \) cycle(s), then we construct \( G_{\text{src}} \) and extract \( X_s \) from \( G_{\text{src}} \).

Source Event Structure Construction. To construct \( G_{\text{src}} \), we follow the construction steps of \( G_{\text{tgt}} \). For each target construction step that adds event \( e \) to \( G_{\text{tgt}} \) to get \( G'_{\text{tgt}} \), we perform one or more corresponding steps going from \( G_{\text{src}} \) to \( G'_{\text{src}} \). We do a case analysis on the event \( e \) of the target event structure. For the reordered events the construction is as follows:

![Diagram](image)

Figure D.1.: \( \{(c_s, c_t), (b_s, b_t), (a_s, a_t), (b', b_t), (d_s, d_t)\} \subseteq \mathbb{M} \).

We define \( \text{pc} : \mathbb{N} \rightarrow \mathbb{E} \); a function that maps a thread identifier to an event in the respective thread in the execution.

We use \( \text{pc} \) to keep track of \( X_s \) in \( G_{\text{src}} \).

We define \( \mathbb{M} \) relation which pairs a \( G_{\text{src}} \) and \( G_{\text{tgt}} \) event, that is,

\[
\mathbb{M} \triangleq \{(s, t) \mid s \in G_{\text{src}}.E \land t \in G_{\text{tgt}}.E \land s.\text{lab} = t.\text{lab} \land s.\text{tid} = t.\text{tid}\}
\]

Let \( A \subseteq G_{\text{tgt}}.E, B \subseteq G_{\text{tgt}}.E \) denote the pair of sets of events which are created for the reordered access pairs.

We call \( A \cup B \) as reordered events and \( G_{\text{tgt}}.E \setminus (A \cup B) \) as non-reordered events.

Also let \( C \subseteq G_{\text{tgt}}.E \setminus (A \cup B) \) be the immediate \( G_{\text{tgt}}.\text{po} \)-predecessors of the \( B \) events.

We say \( G_{\text{src}} \sim G_{\text{tgt}} \) holds iff

1. \( G_{\text{src}}, G_{\text{tgt}} \) are consistent.

2. there exists \( \mathbb{M} \) such that \( G_{\text{src}} \) and \( G_{\text{tgt}} \) preserves invariant which is a conjunction of following clauses.

   a) The non-reordered events in the target event structures are mapped to some non-reordered events in the source event structure.

\[
\forall c_t \in G_{\text{tgt}}.E \setminus (A \cup B), \exists c_s \in G_{\text{src}}.E, \mathbb{M}(c_s, c_t)
\]
b) If \( b_t \) is po-successor of some event \( c_t \) in the target event structure then there exists \( a', b_s, c_s \) events in the source event structure such that \( \mathcal{M}(b_s, b_t), \mathcal{M}(c_s, c_t) \) hold. In addition, memory location and memory order of \( a' \) and \( a_t \) match.

\[
\forall c_t \in G_{tgt}.E \setminus (A \cup B), a_t \in A, b_t \in B \land G_{tgt}.po(c_t, b_t) \implies \\
\exists c_s, a_s, b_s \in G_{src}.E. \mathcal{M}(c_s, c_t) \land \mathcal{M}(a_s, a_t) \land \mathcal{M}(b_s, b_t) \\
\land (\exists a' \in G_{src}.E. a_s.\text{loc} = a'.\text{loc} \land a_s.\text{ord} = a'.\text{ord} \land G_{src}.po(c_s, a') \\
\land \text{imm}(G_{src}.po)(a', b_s))
\]

c) If \( a_t \) is po-successor of some event \( c_t \) in the target event structure then there exists \( a_s, c_s \) events in the source event structure such that \( \mathcal{M}(a_s, a_t) \) and \( \mathcal{M}(c_s, c_t) \) hold.

\[
\forall c_t \in G_{tgt}.E \setminus (A \cup B), a_t \in A \land G_{tgt}.po(c_t, a_t) \implies \\
\exists c_s, a_s \in G_{src}.E. \mathcal{M}(c_s, c_t) \land \mathcal{M}(a_s, a_t) \land G_{src}.po(c_s, a_s)
\]

d) If \( a_t \in A \) is immediate-po successor of \( b_t \in B \) in the target event structure then there exist \( a_s, a', b_s, b', c_s, c_t \) such that

i. \( \{(c_s, c_t), (b_s, b_t), (a_s, a_t)\} \subseteq \mathcal{M} \) holds.

ii. \( c_s \) and \( c_t \) are non-reordered events such that if \( c_t \) is immediate-po-predecessor of \( b_t \) then \( c_s \) is immediate-po predecessor of \( a_s \).

iii. \( a' \) and \( a \) are in immediate-conflict relation.

iv. \( b_s \) and \( b' \) are immediate-po successors of \( a' \) and \( a_s \) respectively.

v. \( b' \) and \( b_s \) are equal-writes.

\[
\forall a_t \in A, b_t \in B. \text{imm}(G_{tgt}.po)(b_t, a_t) \implies \\
(\exists c_t \in G_{tgt}.E \setminus (A \cup B), a', a_s, b_s, c_s \in G_{src}.E. \mathcal{M}(c_s, c_t) \land \mathcal{M}(a_s, a_t) \land \mathcal{M}(b_s, b_t) \\
\land \text{imm}(G_{tgt}.po)(c_t, b_t) \land \text{imm}(G_{src}.po)(c_s, a_s) \land \text{imm}(G_{src}.po)(a_s, b') \\
\land \text{imm}(G_{src}.ew)(a_s, a') \land \text{imm}(G_{src}.po)(a', b_s) \\
\land b_s.\text{loc} = b'.\text{loc} \land b_s.\text{ord} = b'.\text{ord} \land G_{src}.ew(b_s, b'))
\]

e) If non-reordered event \( c_t \) is po-successor of \( b_t \) in the target event structure then there exists \( c_s \) in source event structure which maps to \( c_t \) and \( c_s \) is po-successor of \( b' \) or \( b_s \) where \( b' \) and \( b_s \) are equal-writes.

\[
\forall c_t \in G_{tgt}.E \setminus (A \cup B), b_t \in B. G_{tgt}.po(b_t, c_t) \implies \\
\exists b_{s}, b', c_s \in G_{src}.E. \mathcal{M}(c_s, c_t) \land \mathcal{M}(b_s, b_t) \land \mathcal{M}(b', b_t) \\
\land G_{src}.ew(b_s, b') \land (G_{src}.po(b_s, c_s) \lor G_{src}.po(b', c_s))
\]

f) If \( a_t \in A \) is immediate-po successor of \( b_t \in B \) in the target event structure then there is no po relation between \( b_s \) and \( a_s \) in source event structure where \( a_s \) maps to \( a_t \) and \( b_s \) maps to \( b_t \).

\[
\forall a_t \in A, b_t \in B. G_{tgt}.po(b_t, a_t) \implies \\
\exists a_s, b_s \in G_{src}.E. \mathcal{M}(a_s, a_t) \land \mathcal{M}(b_s, b_t) \land \neg G_{src}.po(b_s, a_s)
\]
D. Proofs of Correctness of Reorderings

g) For a pair of non-ordered events in the target event structure which are in po relation, there exists corresponding pair of events in the source event structure such that \( a_s \) is justified from \( c_s \).

\[ \forall c_t \in G_{tgt} \setminus (A \cup B), a_t \in A. G_{tgt}.jf(c_t, a_t) \implies \exists c_s, a_s \in G_{src}.M(a_s, a_t) \land M(c_s, c_t) \land G_{src}.jf(c_s, a_t) \]

h) If \( a_t \) is justified from an event \( c_t \) in the target event structure then there exists corresponding \( a_s, c_s \) events in the source event structure such that \( a_s \) is justified from \( c_s \).

\[ \forall c_t \in G_{tgt} \setminus (A \cup B), a_t \in A. G_{tgt}.jf(c_t, a_t) \implies \exists c_s, a_s \in G_{src}.M(a_s, a_t) \land M(c_s, c_t) \land G_{src}.jf(c_s, a_t) \]

i) If \( a_t \) justifies an event \( c_t \) in the target event structure then there exists corresponding \( a_s, c_s \) events in the source event structure such that \( a_s \) justifies \( c_s \).

\[ \forall c_t \in G_{tgt} \setminus (A \cup B), a_t \in A. G_{tgt}.jf(c_t, a_t) \implies \exists c_s, a_s \in G_{src}.M(a_s, a_t) \land M(c_s, c_t) \land G_{src}.jf(c_s, a_s) \]

j) If \( b_t \) is justified from an event \( c_t \) in the target event structure then there exists corresponding \( b' \) and \( b_s, c_s \) events in the source event structure such that \( c_s \) justifies \( b_s, b' \), and \( b_s, b' \) are equal-writes.

\[ \forall c_t \in G_{tgt} \setminus (A \cup B), b_t \in B. G_{tgt}.jf(c_t, b_t) \implies \exists b_s, c_s \in G_{src}.M(b_s, b_t) \land M(c_s, c_t) \land G_{src}.jf(c_s, b_s) \land (\exists b' \in G_{src}.M(b', b_t) \land G_{src}.ew(b_s, b') \implies G_{src}.jf(c_s, b')) \]

k) If \( b_t \) in the target event structure justifies \( c_t \) then either there exists \( b' \) corresponding to \( b_t \) that \( b' \) justifies \( c_s \) where there is no \( b_s \) that maps to \( b_t \) or source event structure has \( b_s \) which is equal-writes to \( b' \) and justifies \( c_s \).

\[ \forall c_t \in G_{tgt} \setminus (A \cup B), b_t \in B. G_{tgt}.jf(b_t, c_t) \implies ((\exists b_s, c_s \in G_{src}.M(b_s, b_t) \land \exists b' \in G_{src}.M(b', b_t) \land G_{src}.ew(b_s, b')) \implies G_{src}.jf(b_s, c_s)) \land (\exists b', c_s \in G_{src}.M(b', b_t) \land M(c_s, c_t) \land G_{src}.ew(b_s, b') \implies G_{src}.jf(b', c_s))) \]

l) If a pair of non-reordered events are in justified-from relation, then there exists corresponding pair of events in the source event structure in justified-from relation.

\[ \forall c_t, c'_t \in G_{tgt} \setminus (A \cup B). G_{tgt}.jf(c_t, c'_t) \implies \exists c_s, c'_s \in G_{src}.M(c_s, c_t) \land M(c'_s, c'_t) \land G_{src}.jf(c_s, c'_s) \]
m) If there is mo relation from a non-reordered event \( c_t \) to an ordered event \( a_t \) then there exists events \( c_s, a_s \) in mo relation in source event structure where non-reordered event \( c_s \) maps to \( c_t \) and ordered event \( a_s \) maps to \( a_t \).

\[
\forall c_t \in G_{tgt}.E \setminus (A \cup B), a_t \in A, b_t \in B. G_{tgt}.mo(c_t, a_t) \implies \\
\exists c_s, a_s \in G_{src}.E. M(c_s, c_t) \land M(a_s, a_t) \land G_{src}.mo(c_s, a_s)
\]

n) If there is mo relation from an ordered event \( a_t \) to a non-reordered event \( c_t \) then there exists mo relation from event \( a_s \) to \( c_s \) in source event structure where ordered event \( a_s \) maps to \( a_t \) and non-reordered event \( c_s \) maps to \( c_t \).

\[
\forall c_t \in G_{tgt}.E \setminus (A \cup B), a_t \in A. G_{tgt}.mo(a_t, c_t) \implies \\
\exists c_s, a_s \in G_{src}.E. M(c_s, c_t) \land M(a_s, a_t) \land G_{src}.mo(c_s, a_t)
\]

o) If there is mo relation from a non-reordered event \( c_t \) to an ordered event \( b_t \) then there exists events \( c_s, b_s \) in mo relation in source event structure where non-reordered event \( c_s \) maps to \( c_t \) and ordered event \( b_s \) maps to \( b_t \).

\[
\forall c_t \in G_{tgt}.E \setminus (A \cup B), b_t \in B. G_{tgt}.mo(c_t, b_t) \implies \\
\exists c_s, b_s \in G_{src}.E. M(c_s, c_t) \land M(b_s, b_t) \land G_{src}.mo(c_s, b_s)
\]

p) If there is mo relation from an ordered event \( b_t \) to a non-reordered event \( c_t \) then there exists mo relation from event \( b_s \) to \( c_s \) in source event structure where ordered event \( b_s \) maps to \( b_t \) and non-reordered event \( c_s \) maps to \( c_t \).

\[
\forall c_t \in G_{tgt}.E \setminus (A \cup B), b_t \in B. G_{tgt}.mo(b_t, c_t) \implies \\
\exists c_s, b_s \in G_{src}.E. M(c_s, c_t) \land M(b_s, b_t) \land G_{src}.mo(b_s, c_s)
\]

q) If there is mo relation between a pair of non-reordered events \( c_t \) and \( c'_t \) in the target event structure then there exists mo relation from event \( c_s \) to \( c'_s \) in source event structure where \( c_s \) maps to \( c_t \) and \( c'_s \) maps to \( c'_t \).

\[
\forall c, c' \in G_{tgt}.E \setminus (A \cup B). G_{tgt}.mo(c_t, c'_t) \implies \\
\exists c_s, c'_s \in G_{src}.E. M(c_s, c_t) \land M(c'_s, c'_t) \land G_{src}.mo(c_s, c'_s)
\]

r) If an event is unmapped in the source event structure then there is no outgoing mo edge from that event.

\[
\forall e_s \in G_{src}.W. (\#e_t \in G_{tgt}.E. M(e_s, e_t)) \implies \\
\#e'_s \in G_{src}.E. G_{src}.mo(e_s, e'_s)
\]

s) For each equal-writes pair of events in the target event structure, there exists equal-writes pairs in the source event structure.

\[
\forall c_t, c'_t \in G_{tgt}.E. G_{tgt}.ew(c_t, c'_t) \implies \\
\exists c_s, c'_s \in G_{src}.E. M(c_s, c_t) \land M(c'_s, c'_t) \land G_{src}.ew(c_s, c'_s)
\]
D. Proofs of Correctness of Reorderings

3. there exists pc such that
\[ X_s.E = S \]
\[ X_s.po = G_{src}.po \cap (S \times S) \]
\[ X_s.rf = G_{src}.rf \cap (S \times S) \]
\[ X_s.mo = G_{src}.mo \cap (S \times S) \]

where \( S(G_{src}, pc) \triangleq \{ e \mid e \in G_{src}.E \wedge G_{src}.po^3(e, pc(tid)) \} \).

To prove the simulation we show the followings.
\[ G_{src} \sim G_{tgt} \wedge G_{tgt} \xrightarrow{WEAKESTMO} G_{tgt}' \implies \exists G_{src}'. G_{src} \xrightarrow{WEAKESTMO} G_{src}' \wedge G_{src}' \sim G_{tgt}' \]

At each construction step, we extend \( G_{tgt} \) to \( G_{src}' \) by po-extending from an event \( e_t \in G_{tgt}.E \) with a new event \( e'_t \in G_{tg}'t'.E \). We consider following cases:

**Case** \( e'_t \in B' \) **where** \( B' = B \uplus \{ e'_t \} \):

In this case \( A' = A \), and \( G_{src}'t'.E = G_{tg}'.E \uplus \{ e'_t \} \).
We also append corresponding event(s) in \( G_{src} \) and construct \( G_{src}' \).

1. Condition to show: \( G_{src}' \) is consistent.

The construction has two steps: \( G_{src} \rightarrow G_{src}'' \rightarrow G_{src}' \). In \( G_{src}'' \) we introduce \( a' \) and in \( G_{src}' \) we introduce \( e'_s \).

**case.** event \( e_s \) has an immediate po successor \( a'' \) such that \( a.loc = a''.loc \) and \( a\text{.ord} = a''.\text{ord} \). In this case \( a' = a'' \) and \( G_{src}' = G_{src}'' \).

**otherwise.**

We append an event \( a' \) in \( G_{src} \) and create \( G_{src}'' \) such that
\[ G_{src}.E = G_{src}'.E \uplus \{ a' \} \]
\[ G_{src}'p_0 = (G_{src}.p_0 \uplus \{ (e_s, a') \mid M(e_s, e_t) \})^+ \]
\[ G_{src}'f = G_{src}.f \]
\[ \uplus \{ (w, a') \mid (w, a') \in (G_{src}''.w \times G_{src}''.R) \]
\[ \wedge \exists w' \in G_{tg}'.E. M(w, w') \wedge G_{tg}''.f(w', a) \} \]
\[ \uplus \{ (w, a') \mid (w, a') \in (G_{src}''w \times G_{src}''R) \]
\[ \wedge \# w' \in G_{tg}'.E. M(w, w') \wedge G_{tg}''.f(w', a) \wedge \exists w(G_{src}''w, a') \} \]
\[ G_{src}'.m_0 = G_{src}.m_0 \uplus \{ (w, a') \mid (w, a') \in (G_{src}''w \times G_{src}''W) \} \]
\[ G_{src}'.e = G_{src}.e \]

Also in this case \( M'' = M \).

Now we check whether \( G_{src}'' \) is consistent.
We know that \( G_{tg} \sim G_{src} \). Hence \( G_{src} \) and \( G_{tg} \) are consistent.
If \( G''_\text{src} = G'_\text{src} \) then \( G''_\text{src} \) is consistent as \( G'_\text{src} \) is consistent.

Otherwise, from definition of \( G''_\text{src} \) and observation from Remark 3 we know that \( G''_\text{src} \) satisfies (CF), (CFJ), (VIS), (ICF), (ICFJ).

There is no outgoing edge from \( a' \) and hence it does not result in any \( (G''_\text{src}.\text{hb}; G''_\text{src}.\text{eco'}) \) cycle. Hence \( G''_\text{src} \) satisfies (COH').

We show that (NCFU) constraint holds on \( G''_\text{src} \).

From Table 7.1, we consider two cases:

1. \( a'.\text{lab} \neq U_{\text{ACQ}} \). In this case (NCFU) holds from the definition of \( G''_\text{src} \).
2. \( a'.\text{lab} = U_{\text{ACQ}} \). In this case \( b.\text{lab} = F_{\text{ACQ}} \) and \( a' \) is justified-from a write \( w' \in G'_\text{tgt}.E \) such that \( M'(w, w') \land G'_\text{tgt}.\text{if}(w', a) \) holds. Hence \( G''_\text{src} \) satisfies (NCFU) as we know that (NCFU) holds on \( G'_\text{src} \) and \( G''_\text{tgt} \).

We show that the (NCFSC) constraint holds on \( G''_\text{src} \).

We consider two cases on \( a' : \) **case** \( G''_\text{src}.\text{if}(w, a) \) where \( \exists w' \in G'_\text{tgt}.E \) \( M'(w, w') \land G'_\text{tgt}.\text{if}(w', a) \land \exists w W_{\text{src}} (w', a') \).

In this case \( a'.\text{lab} = L_{\text{IF}} \) or \( \text{a'.lab} = U_{\text{REL}} \) and \( G''_\text{src}.\text{if}(w, a) \) does not create any \( G''.\text{pscb} \) or \( G''.\text{pscf} \) relations. Hence \( G''_\text{src} \) satisfies (NCFSC) as \( G'_\text{tgt} \) and \( G_\text{src} \) satisfy (NCFSC).

**otherwise** Hence \( G''_\text{src} \) satisfies (NCFSC) as \( G'_\text{tgt} \) and \( G_\text{src} \) satisfy (NCFSC).

As a result, \( G''_\text{src} \) remains consistent.

Next, we construct \( G'_\text{src} \) from \( G''_\text{src} \).

**case.** There exists \( e'_s \) where \( e'_s.\text{lab} = e'_t.\text{lab} \) and if \( e'_s.e'_t \in R \) then \( G''_\text{src}.\text{if}(w_s, e'_s), G''_\text{src}.\text{if}(w_t, e'_t), M'(w_s, w_t) \) hold.

In this case \( G'_\text{src} = G''_\text{src} \) and \( b_s = e'_s \).

**otherwise.** We append such an event \( e'_s \) and thus

\[
G'_\text{src}.E = G''_\text{src}.E \cup \{ (w_s, e'_s) | (w_s, e'_t).\text{lab} = e'_t.\text{lab} \} \\
G'_\text{src}.\text{po} = (G''_\text{src}.\text{po} \cup \{ (a', e'_s) \})^+ \\
G'_\text{src}.\text{if} = G''_\text{src}.\text{if} \cup \{ (w_s, e'_s) | (w_s, e'_t) \in (G'_\text{src}.W \times G'_\text{src}.R) \land G'_\text{tgt}.\text{if}(w_t, e'_t) \land M''(w_s, w_t) \} \\
G'_\text{src}.\text{mo} = G''_\text{src}.\text{mo} \cup \{ (w_s, e'_s) | (w_s, e'_t) \in (G'_\text{src}.W \times G'_\text{src}.W) \land M''(w_s, w_t) \land G'_\text{tgt}.\text{mo}(w_t, e'_t) \} \\
G'_\text{src}.\text{ew} = G''_\text{src}.\text{ew} \cup \{ (w_s, e'_s), (e'_t, w_s) \land (w_s, e'_s) \in (G''_\text{src}.W_{\text{RLX}} \times G'_\text{src}.W_{\text{RLX}}) \land M''(w_s, w_t) \land G'_\text{tgt}.\text{ew}(w_t, e'_t) \} \\
\]
D. Proofs of Correctness of Reorderings

Also in this case $M' = M'' \cup \{(e'_s, e'_t)\}$.

Now we check whether $G'_{src}$ is consistent.

If $G'_{src} = G''_{src}$ then $G_{src}$ is consistent as $G''_{src}$ is consistent.

Otherwise, we check whether $G'_{src}$ is consistent.

We know $G'_{src}$ and $G'_{tgt}$ preserve (CF). As a result, from the construction $(e'_s, e'_t) \notin G'_{src}.ecf$. Hence $G'_{src}$ preserves (CF).

We know $G''_{src}$ preserves (CFJ). Moreover, $G'_{tgt}.jf(w_t, e'_t)$ implies $\neg G'_{src}.ecf(w_t, e'_s)$. As a result, from the construction $\neg G'_{src}.ecf(w_s, e'_s)$ where $M''(w_s, w_t)$ holds. Hence $G'_{src}$ preserves (CFJ).

We know $G''_{src}$ preserves (VISJ). Moreover, $G'_{tgt}.jf(w_t, e'_t)$ implies $w_t \in \text{vis}(G'_{tgt})$. As a result, from the construction $w_s \in \text{vis}(G''_{src})$ where $M''(w_s, w_t)$ holds. Hence $G'_{src}$ preserves (VISJ).

We know $G''_{src}$ and $G'_{tgt}$ preserves (ICF). Hence following the construction we know if $e'_s \notin G'_{src}.R$ then there exists no event $e_1$ such that $G'_{src}.e \sim (e'_s, e_1)$. Hence $G'_{src}$ preserves (ICF).

We know $G''_{src}$ preserves (ICFJ). Moreover, following the construction of $G'_{src}$ from $G''_{src}, (w_s, w_s) \notin G'_{src}.jf; \text{imm}(cf); G_{src}.rf^{-1}$. Hence $G'_{src}$ preserves (ICFJ).

We know $G''_{src}$ preserves (COH') and consider there is a $(G'_{src}.hb; G'_{src}.eco)$ cycle. In that case $e'_s$ is part of the $(G'_{src}.hb; G'_{src}.eco)$ cycle. However, following the construction of $G'_{src}$, in this case, there exists a $(G'_{tgt}.hb; G'_{tgt}.eco)$ cycle. This is not possible as $G'_{tgt}$ is consistent. Hence a contradiction and $G'_{src}$ preserves (COH').

We know $G''_{src}$ preserves (NCFU) and (NCFSC). Consider $G'_{src}$ violates (NCFU) or (NCFSC). In that case $G'_{src}$ violates (NCFU) or (NCFSC) due to $e'_t$. However, following the construction of $G'_{src}$, in this case, $G''_{src}$ also violates (NCFU) or (NCFSC) due to $e'_t$. This is not possible as $G'_{src}$ is consistent. Hence a contradiction and $G'_{src}$ preserves (NCFU) and (NCFSC).

As a result, $G'_{src}$ is consistent.

Thus finally $M' = M \cup \{(e'_s, e'_t)\}$ and $pc' = pc$.

2. Condition to show: the simulation invariant holds between $G'_{src}$ and $G'_{tgt}$

a) $\forall c_t \in G'_{tgt}.E \setminus (A' \cup B')$. $\exists c_s \in G'_{src}.E. M'(c_s, c_t)$

We know this condition holds between $G_{src}$ and $G_{tgt}$. Hence the condition holds between $G'_{src}$ and $G'_{tgt}$. Hence the condition holds between $G'_{src}$ and $G'_{tgt}$ as $e'_t \notin G'_{tgt}.E \setminus (A' \cup B')$. 

D.1. Reordering Theorem

b) \[ \forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), a_t \in A', b_t \in B' \land G'_{tgt}.po(c_t, b_t) \implies \]
\[ \exists c_s, a_s, b_s \in G'_{src}.E. M'(c_s, c_t) \land M'(a_s, a_t) \land M'(b_s, b_t) \]
\[ \land (\exists a'' \in G'_{src}.E. a_s.loc = a''.loc \land a_s.ord = a''.ord \land G'_{src}.po(c_s, a'') \land \text{imm}(G'_{src}.po)(a'', b_s)) \]

We know this condition holds between \( G_{src} \) and \( G_{tgt} \). Considering the definitions of \( G'_{src} \) and \( G'_{tgt} \), the condition holds between \( G'_{src} \) and \( G'_{tgt} \) where \( b_t = e'_t \), \( a_s = e'_s \), and \( a'' = a' \).

c) \[ \forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), a_t \in A'. \land G'_{tgt}.po(c_t, a_t) \implies \]
\[ \exists c_s, a_s \in G'_{src}.E. M'(c_s, c_t) \land M'(a_s, a_t) \land G'_{src}.po(c_s, a_s) \]

We know this condition holds between \( G_{src} \) and \( G_{tgt} \). Considering the definitions of \( G'_{src} \) and \( G'_{tgt} \), \( M' \) this condition holds between \( G'_{src} \) and \( G'_{tgt} \) for all \( e'_t \), \( e'_s \), \( a' \).

d) \[ \forall a_t \in A', b_t \in B'. \text{imm}(G'_{tgt}.po)(b_t, a_t) \implies \]
\[ (\exists c_t \in G'_{tgt}.E \setminus (A' \cup B'), a'_t, b_s, c_s \in G'_{src}.E. M'(c_s, c_t) \land M'(a_s, a_t) \land M'(b_s, b_t) \]
\[ \land \text{imm}(G'_{tgt}.po)(c_t, b_t) \land \text{imm}(G'_{src}.po)(c_s, a_s) \land \text{imm}(G'_{src}.po)(a'_s, b'_t) \]
\[ \land \text{imm}(G'_{src}.e)(a'_s, a'') \land \text{imm}(G'_{src}.po)(a''_s, b'_s) \land b_s.loc = b'.loc \land b_s.ord = b'.ord \]
\[ \land G'_{src}.ew(b_s, b'_s)) \]

We know this condition holds between \( G_{src} \) and \( G_{tgt} \). The event \( e'_t \) is \( G'_{tgt}.po \)-maximal and hence \( \text{imm}(G'_{tgt}.po)(b_t, a_t) \) does not hold when \( b_t = e'_t \). Hence the condition holds between \( G'_{src} \) and \( G'_{tgt} \).

e) \[ \forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), b_t \in B'. G'_{tgt}.po(b_t, c_t) \implies \]
\[ \exists b_s, b'_t, c_s \in G'_{src}.E. M'(c_s, c_t) \land M'(b_s, b_t) \land M'(b'_t, b_t) \]
\[ \land G'_{src}.ew(b_s, b'_t) \land (G'_{src}.po(b_s, c_s) \lor G'_{src}.po(b'_t, c_s)) \]

We know this condition holds between \( G_{src} \) and \( G_{tgt} \). The event \( e'_t \) is \( G'_{tgt}.po \)-maximal and hence \( G'_{tgt}.po(b_t, c_t) \) does not hold when \( b_t = e'_t \). Hence the condition holds between \( G'_{src} \) and \( G'_{tgt} \).

f) \[ \forall a_t \in A', b_t \in B'. G'_{tgt}.po(b_t, a_t) \implies \]
\[ \exists a_s, b_s \in G'_{src}.E. M'(a_s, a_t) \land M'(b_s, b_t) \land \neg G'_{src}.po(b_s, a_s) \]

We know this condition holds between \( G_{src} \) and \( G_{tgt} \). The event \( e'_t \) is \( G'_{tgt}.po \)-maximal and hence \( G'_{tgt}.po(b_t, a_t) \) does not hold when \( b_t = e'_t \). Hence the condition holds between \( G'_{src} \) and \( G'_{tgt} \).
D. Proofs of Correctness of Reorderings

\[ \forall c_t, c'_t \in G'_\text{tgt}.E \setminus (A' \cup B'), G'_\text{tgt}.\text{po}(c_t, c'_t) \implies \exists c_s, c'_s \in G'_\text{src}.E. M'(c_s, c_t) \land M'(c'_s, c'_t) \land G'_\text{src}.\text{po}(c_s, c'_s) \]

We know the condition holds between \( G'_\text{src} \) and \( G'_\text{tgt} \). Considering the definitions of \( G'_\text{src}, G'_\text{tgt}, M' \), the condition holds between \( G'_\text{src} \) and \( G'_\text{tgt} \) as \( c'_t \notin G'_\text{tgt}.E \setminus (A' \cup B') \).

\[ \forall c_t \in G'_\text{tgt}.E \setminus (A' \cup B'), a_t \in A', G'_\text{tgt}.\text{jf}(c_t, a_t) \implies \exists c_s, a_s \in G'_\text{src}.E. M'(a_s, a_t) \land M'(c_s, c_t) \land G'_\text{src}.\text{jf}(c_s, a_s) \]

We know the condition holds between \( G'_\text{src} \) and \( G'_\text{tgt} \). Considering the definitions of \( G'_\text{src}, G'_\text{tgt}, M' \), the condition holds between \( G'_\text{src} \) and \( G'_\text{tgt} \) as \( c'_t \notin G'_\text{tgt}.E \setminus (A' \cup B') \) or \( c'_t \notin A \).

\[ \forall c_t \in G'_\text{tgt}.E \setminus (A' \cup B'), a_t \in A', G'_\text{tgt}.\text{jf}(a_t, c_t) \implies \exists c_s, a_s \in G'_\text{src}.E. M'(a_s, a_t) \land M'(c_s, c_t) \land G'_\text{src}.\text{jf}(a_s, c_s) \]

We know the condition holds between \( G'_\text{src} \) and \( G'_\text{tgt} \). Considering the definitions of \( G'_\text{src}, G'_\text{tgt}, M' \), the condition holds between \( G'_\text{src} \) and \( G'_\text{tgt} \) as \( c'_t \notin G'_\text{tgt}.E \setminus (A' \cup B') \) and \( c'_t \notin A \).

\[ \forall c_t \in G'_\text{tgt}.E \setminus (A' \cup B'), b_t \in B', G'_\text{tgt}.\text{jf}(c_t, b_t) \implies \exists b_s, c_s \in G'_\text{src}.E. M'(b_s, b_t) \land M'(c_s, c_t) \land G'_\text{src}.\text{jf}(c_s, b_s) \land G'_\text{src}.\text{ew}(b_s, b_t) \implies G'_\text{src}.\text{jf}(c_s, b') \]

We know the condition holds between \( G'_\text{src} \) and \( G'_\text{tgt} \). Considering the definitions of \( G'_\text{src}, G'_\text{tgt}, M' \), the condition holds between \( G'_\text{src} \) and \( G'_\text{tgt} \) where \( b_s = c'_t \) and there exists no \( b' \) such that \( M'(b_s, b') \).

\[ \forall c_t \in G'_\text{tgt}.E \setminus (A' \cup B'), b_t \in B', G'_\text{tgt}.\text{jf}(b_t, c_t) \implies \exists b_s, c_s \in G'_\text{src}.E. (M'(b_s, b_t) \land \exists b' \in G'_\text{src}.E. M'(b', b_t) \land G'_\text{src}.\text{ew}(b_s, b')) \implies G'_\text{src}.\text{jf}(b_s, c_s) \]

We know the condition holds between \( G'_\text{src} \) and \( G'_\text{tgt} \). Considering the definitions of \( G'_\text{src}, G'_\text{tgt}, M' \), the condition holds between \( G'_\text{src} \) and \( G'_\text{tgt} \) where \( b_s = c'_t \) and there exists no \( b' \in G'_\text{src}.E \) such that \( M'(b', b_t) \) and \( G'_\text{src}.\text{ew}(b_s, b') \) holds.
D.1. Reordering Theorem

l) \[ \forall c_t, c'_t \in G'_{tgt}.E \setminus (A' \cup B'). G'_{tgt}.jf(c_t, c'_t) \implies \exists c_s, c'_s \in G'_{src}.E. M'(c_s, c_t) \land M'(c'_s, c'_t) \land G'_{src}.jf(c_s, c'_s) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). Considering the definitions of \( G'_{src} \), \( G'_{tgt} \), \( M' \), the condition holds between \( G'_{src} \) and \( G'_{tgt} \) as \( e'_t \notin G'_{tgt}.E \setminus (A' \cup B') \).

m) \[ \forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), a_t \in A', b_t \in B'. G'_{tgt}.mo(c_t, a_t) \implies \exists c_s, a_s \in G'_{src}.E. M'(c_s, c_t) \land M'(a_s, a_t) \land G'_{src}.mo(a_s, c_s) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). Considering the definitions of \( G'_{src} \), \( G'_{tgt} \), \( M' \), the condition holds between \( G'_{src} \) and \( G'_{tgt} \) as \( e'_t \notin G'_{tgt}.E \setminus (A' \cup B') \) and for all \( a_t \in A' \), \( \neg M'(a', a_t) \) holds.

n) \[ \forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), a_t \in A', G'_{tgt}.mo(a_t, c_t) \implies \exists c_s, a_s \in G'_{src}.E. M'(c_s, c_t) \land M'(a_s, a_t) \land G'_{src}.mo(a_s, c_s) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). Considering the definitions of \( G'_{src} \), \( G'_{tgt} \), \( M' \), the condition holds between \( G'_{src} \) and \( G'_{tgt} \) as \( e'_t \notin G'_{tgt}.E \setminus (A' \cup B') \) and \( e'_t \notin A' \).

o) \[ \forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), b_t \in B'. G'_{tgt}.mo(c_t, b_t) \implies \exists c_s, b_s \in G'_{src}.E. M'(c_s, c_t) \land M'(b_s, b_t) \land G'_{src}.mo(c_s, b_s) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). Following the definitions of \( G'_{src} \) and \( G'_{tgt} \), \( M' \), the condition holds between \( G'_{src} \) and \( G'_{tgt} \) where \( b_t = e'_t \) and \( b_s = e'_s \).

p) \[ \forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), b_t \in B'. G'_{tgt}.mo(b_t, c_t) \implies \exists c_s, b_s \in G'_{src}.E. M'(c_s, c_t) \land M'(b_s, b_t) \land G'_{src}.mo(b_s, c_s) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). Following the definitions of \( G'_{src} \), \( G'_{tgt} \), \( M' \), the condition holds between \( G'_{src} \) and \( G'_{tgt} \) where \( b_t = e'_t \) and \( b_s = e'_s \).

q) \[ \forall c_t, c'_t \in G'_{tgt}.E \setminus (A' \cup B'). G'_{tgt}.mo(c_t, c'_t) \implies \exists c_s, c'_s \in G'_{src}.E. M'(c_s, c_t) \land M'(c'_s, c'_t) \land G'_{src}.mo(c_s, c'_s) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). Following the definitions of \( G'_{src} \), \( G'_{tgt} \), \( M' \), the condition holds between \( G'_{src} \) and \( G'_{tgt} \) as \( e'_t \notin G'_{tgt}.E \setminus (A' \cup B') \).
D. Proofs of Correctness of Reorderings

r)
∀os ∈G′src.
\forall\exists ot ∈G′tgt. E. M′(os, ot) \implies \exists o′ \in G′src. E. G′src.mo(os, o′)

We know the condition holds between Gsrc and Gtgt. Following the definitions of G′src, G′tgt, M′, the condition holds where o_s = a′.

s)
∀ct, c′t ∈ G′tgt. E. G′tgt.ew(ct, c′t) \implies \exists c_s, c′s \in G′src. E. M′(c_s, c_t) \land M′(c′_s, c′_t) \land G′src.ew(c_s, c′_s)

We know the condition holds between Gsrc and Gtgt. Following the definitions of G′src, G′tgt, M′, the condition holds between G′src and G′tgt where c_t = e′_t or c′_t = e′_t and c_s = e′_s and c′_s = e′_s.

Hence the invariant holds between G′src and G′tgt.

3. Condition to show:

**there exists pc′ such that**

\begin{align*}
X′_s, E &= S' \\
X′_s, po &= G′src.po \cap (S' \times S') \\
X′_s, rf &= G′src.rf \cap (S' \times S') \\
X′_s, mo &= G′src.mo \cap (S' \times S') \\
\end{align*}

where \(S'(G′src, pc′) \triangleq \{ e | e \in G′src.E \land G′src.po^+(e, pc′.tid) \}\).

We know there exists pc such that

\begin{align*}
X_s, E &= S \\
X_s, po &= Gsrc.po \cap (S \times S) \\
X_s, rf &= Gsrc.rf \cap (S \times S) \\
X_s, mo &= Gsrc.mo \cap (S \times S) \\
\end{align*}

where \(S(Gsrc, pc) \triangleq \{ e | e \in Gsrc.E \land Gsrc.po^+(e, pc(e.tid)) \}\) and pc′ = pc holds.

In this case \(X′_s = X_s\).

As a result, G′src \sim G′tgt holds.

**Case e′_t \in A where A′ = A \uplus \{e′_t\}:**

The construction has two steps: Gsrc → G″src → G′src. In G″src we introduce e′_s and in G′src we introduce b′.

In this case B′ = B, and G′tg.t E = Gtg.t E \uplus \{e′_t\}.

Let \(c_t \in C\) be the immediate Gtg.t.po-predecessor of e_t, that is, imm(Gtg.t.po)(c_t, e_t).

In Gsrc the event c_s is the corresponding event of c_t, that is, M(c_s, c_t).

We also append corresponding event(s) in Gsrc and construct G′src.
D.1. Reordering Theorem

1. Condition to show: $G''_{src}$ is consistent.

**case.** event $e_s$ has an immediate po successor $a''$ such that $e'_s, \text{lab} = a'' \text{lab}$ and if $e'_t \in \mathcal{R}$ and $G''_{tgt}.jf(w_t, e'_t)$ then there exists $w_s$ such that $M_l(w_s, w_t)$ and $G_{src}.jf(w_s, a'')$.

In this case $e'_s = a''$ and $G''_{src} = G_{src}$.

**otherwise.**

We append an event $e'_s$ in $G_{src}$ by po-extending from $e_s$ and create $G''_{src}$ such that

$$G''_{src}.E = G_{src}.E \uplus \{ e'_s \}$$

$$G''_{src}.po = (G_{src}.po \uplus \{ (e_s, e'_s) | M_l(e_s, e_t) \})^+$$

$$G''_{src}.jf = G_{src}.jf \uplus \{ (w_s, e'_s) | (w_s, e'_s) \in (G''_{src}.W \times G''_{src}.R) \land G'_{tgt}.jf(w_t, e'_t) \land M_l(w_s, w_t) \}$$

$$G''_{src}.mo = G_{src}.mo \uplus \{ (w_s, e'_s) | (w_s, e'_s) \in (G''_{src}.W \times G''_{src}.W) \land M_l(w_s, w_t) \land G'_{tgt}.mo(w_t, e'_t) \}$$

$$G''_{src}.ew = G_{src}.ew \uplus \{ (w_s, e'_s), (w'_s, w_s) | (w_s, e'_s) \in (G''_{src}.W \times G''_{src}.W) \land M_l(w_s, w_t) \land G'_{tgt}.ew(w_t, e'_t) \}$$

Also in this case $M''_l = M \uplus \{ (e'_s, e'_t) \}$.

Now we check whether $G''_{src}$ is consistent.

We know that $G_{tgt} \sim G_{src}$ and hence $G_{src}$ and $G_{tgt}$ are consistent. Now we check whether $G''_{src}$ is consistent.

If $G''_{src} = G_{src}$ then $G''_{src}$ is consistent as $G_{src}$ is consistent.

Otherwise.

We know that $G_{src}$ preserves (ICFJ). Also from the construction of $G''_{src}$, we know there is no $G''_{src}.jf(e'_s, \_$. Hence $G''_{src}$ preserves (ICFJ).

We know that $G_{src}$ preserves (CF), (CFJ), (VISJ), (CFJ). Also $G'_{tgt}.jf(w_t, e'_t)$ implies $e'_s \in \mathcal{R}$, $w_t \in \text{vis}(G'_{tgt})$ and $\neg G'_{tgt}.ecf(w_t, e'_t)$, and $M_l(w_s, w_t)$ holds. Following the construction, $w_s \in \text{vis}(G''_{src})$, $\neg G''_{src}.ecf(w_s, e'_s)$ holds. Hence $G''_{src}$ preserves (CF), (CFJ), (VISJ), (ICF).

We know $G_{src}$ preserves (COH'). Consider there is $(G''_{src}.hb; G''_{src}.eco')$ cycle in $G''_{src}$ and $e'_s$ is a part of this cycle. In that case there is a $(G'_{tgt}.hb; G'_{tgt}.eco')$ cycle in $G'_{tgt}$ and $e'_t$ is a part of the cycle. However, $G'_{tgt}$ preserves (COH') and hence there is no $(G'_{tgt}.hb; G'_{tgt}.eco')$ cycle. Hence a contradiction and $G''_{src}$ preserves (COH').

We know $G''_{src}$ preserves (NCFU) and (NCFSC). Consider $G'_{src}$ violates (NCFU) or (NCFSC). In that case $G''_{src}$ violates (NCFU) or (NCFSC) due to $e'_s$. However, following the construction of $G'_{src}$, in this case, $G''_{src}$ also violates (NCFU) or (NCFSC) due
D. Proofs of Correctness of Reorderings

to $e'_t$. This is not possible as $G'_{\text{tgt}}$ is consistent. Hence a contradiction and $G''_{\text{src}}$ preserves (NCFU) and (NCFSC).

As a result, $G''_{\text{src}}$ is consistent.

Next, we construct $G''_{\text{src}}$ from $G''_{\text{src}}$ where we identify or create $e'_s$.

case. There exists $e'_s$ where $e'_s, \text{lab} = e', \text{lab}$ and if $e'_s, e'_t \in \mathcal{R}$, then $G''_{\text{src}}.jf(w_s, e'_s)$ and $G''_{\text{src}}.jf(w_t, e'_s)$ and $M''(w, w_t)$ hold.

In this case $G''_{\text{src}} = G''_{\text{src}}$.

Otherwise. We append such an $e'_s = b'$ and thus

$$G'_{\text{src}}.E = G''_{\text{src}}.E \cup \{ b' \mid b'.\text{lab} = e_t.\text{lab} \}$$

$$G'_{\text{src}}.po = (G''_{\text{src}}.po \cup \{(e'_s, b')\})^+$$

$$G'_{\text{src}}.jf = (G''_{\text{src}}.jf \cup \{(w_s, b') \mid (w_s, b') \in (G''_{\text{src}}.W \times G''_{\text{src}}.R) \land G'_{\text{tgt}}.jf(w_t, e_t) \land M''(w, w_t) \land \neg G''_{\text{src}}.cf(w_s, e'_s)\})$$

$$G'_{\text{src}}.mo = (G''_{\text{src}}.mo \cup \{(w_s, b') \mid (w_s, b') \in (G''_{\text{src}}.W \times G''_{\text{src}}.W) \land M''(w, w_t) \land G'_{\text{src}}.mo(w_t, e_t) \land \neg G''_{\text{src}}.cf(w_s, b')\})$$

$$G'_{\text{src}}.ew = G''_{\text{src}}.ew \cup \{(w_s, b'), (b', w_s) \mid (w_s, b') \in (G''_{\text{src}}.W_{RLX} \times G''_{\text{src}}.W_{RLX}) \land M''(w, e_t)\}$$

Also in this case $M' = M'' \cup \{(e'_s, e'_t)\}$.

Now we check whether $G''_{\text{src}}$ is consistent.

If $G''_{\text{src}} = G''_{\text{src}}$ then $G''_{\text{src}}$ is consistent as $G''_{\text{src}}$ is consistent.

Otherwise we check the consistency of $G''_{\text{src}}$.

We know $G''_{\text{src}}$ and $G'_{\text{tgt}}$ preserve (CF). As a result, from the construction $(e'_s, e'_t) \notin G'_{\text{src}}.ecf$. Hence $G''_{\text{src}}$ preserves (CF).

We know $G''_{\text{src}}$ preserves (CFJ). Moreover, $G'_{\text{tgt}}.jf(w_t, e'_t)$ implies $\neg G'_{\text{src}}.ecf(w_t, e'_t)$. As a result, from the construction $\neg G'_{\text{src}}.ecf(w_s, e'_t)$ where $M''(w, w_t)$ holds. Hence $G''_{\text{src}}$ preserves (CFJ).

We know $G''_{\text{src}}$ preserves (CFJ). Moreover, $G'_{\text{tgt}}.jf(w_t, e_t)$ implies $\neg G'_{\text{src}}.cf(w_t, e_t)$. As a result, from the construction $\neg G'_{\text{src}}.cf(w_s, b')$ where $M''(w, w_t)$ holds. Hence $G''_{\text{src}}$ preserves (CFJ).

We know $G''_{\text{src}}$ preserves (VISJ). Moreover, $G'_{\text{tgt}}.jf(w_t, e_t)$ implies $w_t \in \text{vis}(G'_{\text{tgt}})$ As a result, from the construction $w_s \in \text{vis}(G'_{\text{src}}) \text{ where } M''(w, w_t)$ holds. Hence $G''_{\text{src}}$ preserves (VISJ).
We know $G''_{\text{src}}$ and $G'_{\text{tgt}}$ preserves (ICF). Hence following the construction we know that $G''_{\text{src}}$ preserves (ICF).

We know that $G''_{\text{src}}$ preserves (ICFJ). Also from the construction of $G'_{\text{src}}$, we know there is no $G'_{\text{src}}$, $j(f(e',-))$. Hence $G'_{\text{src}}$ preserves (ICF).

We know $G''_{\text{src}}$ preserves (COH') and consider there is a $(G'_{\text{src}}, \text{hb}; G'_{\text{src}}, \text{eco}^?)$ cycle. In that case $b'$ is part of the $(G'_{\text{src}}, \text{hb}; G'_{\text{src}}, \text{eco}^?)$ cycle. However, following the construction of $G'_{\text{src}}$, in this case, there exists a $(G'_{\text{tgt}}, \text{hb}; G'_{\text{tgt}}, \text{eco}^?)$ cycle. This is not possible as $G'_{\text{tgt}}$ is consistent. Hence a contradiction and $G'_{\text{src}}$ preserves (COH').

We know $G''_{\text{src}}$ preserves (NCFU) and (NCFSC). Consider $G'_{\text{src}}$ violates (NCFU) or (NCFSC). In that case $G'_{\text{src}}$ violates (NCFU) or (NCFSC) due to $b'$. However, following the construction of $G'_{\text{src}}$, in this case, $G'_{\text{tgt}}$ also violates (NCFU) or (NCFSC). This is not possible as $G'_{\text{tgt}}$ is consistent. Hence a contradiction and $G'_{\text{src}}$ preserves (NCFU) and (NCFSC).

As a result, $G'_{\text{src}}$ is consistent.

Thus finally $M' = M \cup \{(e'_s, e'_t), (b', e_t)\}$ and $pc' = pc|_{e_s,tid \mapsto b'}$.

2. Condition to show: the simulation invariant holds between $G'_{\text{src}}$ and $G'_{\text{tgt}}$

a)  
\[ \forall c_t \in G'_{\text{tgt}}, E \setminus (A' \cup B'). \exists c_s \in G'_{\text{src}}, E. M'(c_s, c_t) \]

In this case $e'_t, e_t \notin G'_{\text{tgt}}, E \setminus (A' \cup B')$. Hence the condition holds.

b)  
\[ \forall c_t \in G'_{\text{tgt}}, E \setminus (A' \cup B'), a_t \in A', b_t \in B' \land G'_{\text{tgt}}, \text{po}(c_t, b_t) \implies \exists c_s, a_s, b_s \in G'_{\text{src}}, E. M'(c_s, c_t) \land M'(a_s, a_t) \land M'(b_s, b_t) \land (\exists a'' \in G'_{\text{src}}, E. a_s.\text{loc} = a''.\text{loc} \land a_s.\text{ord} = a''.\text{ord} \land G'_{\text{src}}, \text{po}(c_s, a'') \land \text{imm}(G'_{\text{src}}, \text{po})(a'', b_s)) \]

We know this condition holds in $G'_{\text{src}}$ and $G'_{\text{tgt}}$. Considering the definitions of $G'_{\text{src}}, G'_{\text{tgt}}$, and $M'$ the condition holds between $G'_{\text{src}}$ and $G'_{\text{tgt}}$ where $e_t \notin G'_{\text{tgt}}, E \setminus (A' \cup B')$ and $e'_t \notin B'$.

c)  
\[ \forall c_t \in G'_{\text{tgt}}, E \setminus (A' \cup B'), a_t \in A' \land G'_{\text{tgt}}, \text{po}(c_t, a_t) \implies \exists c_s, a_s \in G'_{\text{src}}, E. M'(c_s, c_t) \land M'(a_s, a_t) \land G'_{\text{src}}, \text{po}(c_s, a_s) \]

We know this condition holds in $G'_{\text{src}}$ and $G'_{\text{tgt}}$. Considering the definitions of $G'_{\text{src}}, G'_{\text{tgt}}, M'$ this condition holds between $G'_{\text{src}}$ and $G'_{\text{tgt}}$ for $a_t = e'_t$ and $a_s = e'_s$. 

87
D. Proofs of Correctness of Reorderings

d)
\[ \forall a_t \in A', b_t \in B'. \text{imm}(G'_{tgt}.po)(b_t, a_t) \implies \]
\[ (\exists c_t \in G'_{tgt}.E \setminus (A' \cup B'), a_t, b_t, c_t \in G'_{src}.E. M'(c_t, c_t) \land M'(a_t, a_t) \land M'(b_t, b_t) \]
\[ \land \text{imm}(G'_{tgt}.po)(c_t, b_t) \land \text{imm}(G'_{src}.po, c_t, a_t) \land \text{imm}(G'_{src}.po, a_t, b_t) \]
\[ \land G'_{src}.cf(a_t, a') \land \text{imm}(G'_{src}.po)(a', b_t) \]
\[ \land b_s.\text{loc} = b'_s.\text{loc} \land b_s.\text{ord} = b'_s.\text{ord} \land G'_{src}.ew(b_s, b'_s) \]

We know this condition holds in \( G_{src} \) and \( G_{tgt} \). Considering the definitions of \( G'_{src} \), \( G'_{tgt} \), \( M' \) we have \( b_t = e_t, a_t = e'_t, a_s = e'_s, b_s = e_s \) and from the construction we know there exists such an \( a' \in G_{src}.E \) so that \( \text{imm}(G_{src}.po)(a', b_s) \) holds. In this case \( M'(e_s, e_t), M'(b'_s, e_t) \) and \( G'_{tg}.ew(e_s, b'_s) \) hold.

As a result, this condition holds between \( G'_{src} \) and \( G'_{tgt} \).

e)
\[ \forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), b_t \in B', G'_{tgt}.po(b_t, c_t) \implies \]
\[ \exists s, b', c_s \in G'_{src}.E. M'(c_s, c_s) \land M'(b_s, b_t) \land M'(b'_s, b_t) \]
\[ \land G'_{src}.ew(b_s, b'_s) \land (G'_{src}.po(b_s, c_s) \lor G'_{src}.po(b'_s, c_s)) \]

We know this condition holds in \( G_{src} \) and \( G_{tgt} \).

Considering the definitions of \( G'_{src} \), \( G'_{tgt} \), \( M' \) we know \( b', e_t \notin G'_{tgt}.E \setminus (A' \cup B') \). Hence the condition holds between \( G'_{src} \) and \( G'_{tgt} \).

f)
\[ \forall a_t \in A', b_t \in B'. G'_{tgt}.po(b_t, a_t) \implies \]
\[ \exists a_s, b_s \in G'_{src}.E. M'(a_s, a_t) \land M'(b_s, b_t) \land \neg G'_{src}.po(b_s, a_s) \]

We know the condition holds between \( G'_{src} \) and \( G'_{tgt} \).

Considering the definitions of \( G'_{src} \), \( G'_{tgt} \), \( M' \) for \( b_t = e_t, a_t = e'_t, a_s = e'_s, b_s = b' \) the condition holds between \( G'_{src} \) and \( G'_{tgt} \).

g)
\[ \forall c_t, c'_t \in G'_{tgt}.E \setminus (A' \cup B'). G'_{tgt}.po(c_t, c'_t) \implies \]
\[ \exists c_s, c'_s \in G'_{src}.E. M'(c_s, c_t) \land M'(c'_s, c'_t) \land G'_{src}.po(c_s, c'_s) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). In this case \( c'_t \notin G'_{tgt}.E \setminus (A' \cup B') \). Hence the condition holds between \( G'_{src} \) and \( G'_{tgt} \).

h)
\[ \forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), a_t \in A'. G'_{tgt}.jf(c_t, a_t) \implies \]
\[ \exists c_s, a_s \in G'_{src}.E. M'(a_s, a_t) \land M'(c_s, c_t) \land G'_{src}.jf(c_s, a_s) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). Considering the definitions of \( G'_{src} \), \( G'_{tgt} \), \( M' \), the condition holds for \( a_t = c'_t, a_s = e'_s \) between \( G'_{src} \) and \( G'_{tgt} \).
D.1. Reordering Theorem

i) \[
\forall c_t \in G'_t \setminus (A' \cup B'), \ a_t \in A', \ G'_t.jf(a_t, c_t) \implies \\
\exists c_s, a_s \in G'_s. E. M'(a_t, a_t) \land M'(c_s, c_t) \land G'_s.jf(a_s, c_s)
\]

We know the condition holds between \(G'_s\) and \(G'_t\). Considering the definitions of \(G'_s\), \(G'_t\), \(M'_s\), for \(a_t = e'_t\) there is no outgoing edge from \(e'_t\). Hence the condition holds between \(G'_s\) and \(G'_t\).

j) \[
\forall c_t \in G'_t \setminus (A' \cup B'), \ b_t \in B', \ G'_t.jf(c_t, b_t) \implies \\
\exists b_s, c_s \in G'_s. E. M'(b_t, b_t) \land M'(c_s, c_t) \land G'_s.jf(c_s, b_s) \\
\land (\exists b' \in G'_s. E. M'(b', b_t) \land G'_s.ew(b_s, b') \implies G'_s.jf(c_s, b'))
\]

We know the condition holds between \(G'_s\) and \(G'_t\). In this case the condition holds between \(G'_s\) and \(G'_t\) as \(e'_t \notin B'\).

k) \[
\forall c_t \in G'_t \setminus (A' \cup B'), \ b_t \in B', \ G'_t.jf(b_t, c_t) \implies \\
(\exists b_s, c_s \in G'_s. E. (M'(b_t, b_t) \land \exists b' \in G'_s. E. M'(b', b_t) \land G'_s.ew(b_s, b')) \implies \]
\[
G'_s.jf(b_s, c_s) \\
\land (\exists b', b_s, c_s \in G'_s. E. (M'(b_s, b_t) \land M'(b', b_t) \land M'(c_s, c_t) \land G'_s.ew(b_s, b')) \implies \]
\[
G'_s.jf(b', c_s))
\]

We know the condition holds between \(G'_s\) and \(G'_t\). In this case the condition holds between \(G'_s\) and \(G'_t\) as \(e'_t \notin B'\) and \(e'_t \notin G'_t \setminus (A' \cup B')\).

l) \[
\forall c_t, c'_t \in G'_t \setminus (A' \cup B'). \ G'_t.jf(c_t, c'_t) \implies \\
\exists c_s, c'_s \in G'_s. E. M'(c_s, c_t) \land M'(c'_s, c'_t) \land G'_s.jf(c_s, c'_s)
\]

We know the condition holds between \(G'_s\) and \(G'_t\). In this case \(e'_t \notin G'_t \setminus (A' \cup B')\). Hence the condition holds between \(G'_s\) and \(G'_t\).

m) \[
\forall c_t \in G'_t \setminus (A' \cup B'), \ a_t \in A', \ b_t \in B', \ G'_t.mo(a_t, c_t) \implies \\
\exists a_s, a'_s \in G'_s. E. M'(c_s, c_t) \land M'(a_s, a_t) \land G'_s.mo(a'_s, c_s)
\]

We know the condition holds between \(G'_s\) and \(G'_t\). Considering the definitions of \(G'_s\), \(G'_t\), \(M'_s\), for \(a_t = e'_t\) and \(a_s = e'_s\) the condition holds between \(G'_s\) and \(G'_t\).

n) \[
\forall c_t \in G'_t \setminus (A' \cup B'), \ a_t \in A', \ G'_t.mo(a_t, c_t) \implies \\
\exists a_s, a'_s \in G'_s. E. M'(c_s, c_t) \land M'(a_s, a_t) \land G'_s.mo(a'_s, c_s)
\]

We know the condition holds between \(G'_s\) and \(G'_t\). Considering the definitions of \(G'_s\), \(G'_t\), \(M'_s\), for \(a_t = e'_t\) and \(a_s = e'_s\) the condition holds between \(G'_s\) and \(G'_t\).
D. Proofs of Correctness of Reorderings

o) \[ \forall c_t \in G_{tgt} \setminus (A' \cup B'), b_t \in B', G_{tgt}.mo(c_t, b_t) \implies \exists c_s, b_s \in G_{src} \setminus (A' \cup B'), M'(c_s, c_t) \land M'(b_s, b_t) \land G_{src}.mo(c_s, b_s) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). Following the definitions of \( G'_{src} \) and \( G'_{tgt} \), \( M' \), the condition holds between \( G'_{src} \) and \( G'_{tgt} \) as \( e'_t \notin B' \) and \( e'_t \notin G'_{tgt} \setminus (A' \cup B') \).

p) \[ \forall c_t \in G_{tgt} \setminus (A' \cup B'), b_t \in B', G_{tgt}.mo(b_t, c_t) \implies \exists c_s, b_s \in G_{src} \setminus (A' \cup B'), M'(c_s, c_t) \land M'(b_s, b_t) \land G_{src}.mo(b_s, c_s) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). Following the definitions of \( G'_{src} \) and \( G'_{tgt} \), \( M' \), the condition holds between \( G'_{src} \) and \( G'_{tgt} \) as \( e'_t \notin B' \) and \( e'_t \notin G'_{tgt} \setminus (A' \cup B') \).

q) \[ \forall c, c'_t \in G_{tgt} \setminus (A' \cup B'), G_{tgt}.mo(c_t, c'_t) \implies \exists c, c'_s \in G_{src} \setminus (A' \cup B'), M'(c_s, c_t) \land M'(c'_t, c'_t) \land G_{src}.mo(c'_s, c'_t) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). In this case \( e'_t \notin G'_{tgt} \setminus (A' \cup B') \). Hence the condition holds between \( G'_{src} \) and \( G'_{tgt} \).

r) \[ \forall o_s \in G_{src}, \forall (\exists o_t \in G_{tgt} \setminus (A' \cup B'), M'(o_s, o_t)) \implies \exists o'_s \in G''_{src} \setminus (A' \cup B'), G_{src}.mo(o_s, o'_t) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). Following the definitions of \( G'_{src} \) and \( G'_{tgt} \), \( M' \), \( (c'_s, e_t), (b'_t, e_s) \in M' \). Hence the condition holds between \( G''_{src} \) and \( G''_{tgt} \).

s) \[ \forall c_t, c'_t \in G_{tgt} \setminus (A' \cup B'), G_{tgt}.ew(c_t, c'_t) \implies \exists c_s, c'_s \in G_{src} \setminus (A' \cup B'), M'(c_s, c_t) \land M'(c'_s, c'_t) \land G_{src}.ew(c_s, c'_s) \]

We know the condition holds between \( G_{src} \) and \( G_{tgt} \). Following the definitions of \( G'_{src} \) and \( G'_{tgt} \), \( M' \) the condition holds between \( G''_{src} \) and \( G''_{tgt} \) as \( G''_{tgt}.ew = G''_{tgt}.ew \).

Hence the invariant holds between \( G''_{src} \) and \( G''_{tgt} \).

3. Condition to show:

there exists pc' such that

\( X_{s'} E = S' \)

\( X_{s'} po = G'_{src}.po \cap (S' \times S') \)
D.1. Reordering Theorem

\[ X'_s.\text{rf} = G'_{\text{src}.\text{rf}} \cap (S' \times S') \]
\[ X'_s.\text{mo} = G'_{\text{src}.\text{mo}} \cap (S' \times S') \]

where \( S'(G'_{\text{src}.\text{pc}'} \triangleq \{ e \mid e \in G'_{\text{src}.\text{E}} \land G'_{\text{src}.\text{po}'}(e, \text{pc'}(e.\text{tid})) \}) \).

If \( e'_t \notin X'_t \) then \( X'_t = X_t \). In this case \( \text{pc'} = \text{pc}, S' = S, \) and \( X'_s = X_s \).

Otherwise, when \( e'_t \in X'_t \) then \( X'_t \) is an extension of \( X_t \), that is,

\[
X'_t.E = X_t.E \cup \{ e_t, e'_t \} \\
X'_t.\text{po} = (X_t.\text{po} \cup \{ (a, e_t) \mid a \in X_t.E \land G'_{\text{tgt}.\text{po}}(a, e_t) \}) \cup \{ (a, e'_t) \mid a \in X_t.E \land G'_{\text{tgt}.\text{po}}(a, e'_t) \} \cup \{ (e_t, e'_t) \mid a \in X_t.E \land G'_{\text{tgt}.\text{po}}(e_t, a) \} \\
X'_t.\text{rf} = X_t.\text{rf} \cup \{ (a, e_t) \mid a \in X_t.E \land G'_{\text{tgt}.\text{rf}}(a, e_t) \} \cup \{ (a, e'_t) \mid a \in X_t.E \land G'_{\text{tgt}.\text{rf}}(a, e'_t) \} \cup \{ (e'_t, a) \mid a \in X_t.E \land G'_{\text{tgt}.\text{rf}}(e'_t, a) \} \\
X'_t.\text{mo} = X_t.\text{mo} \cup \{ (a, e_t) \mid a \in X_t.E \land G'_{\text{tgt}.\text{mo}}(a, e_t) \} \cup \{ (a, e'_t) \mid a \in X_t.E \land G'_{\text{tgt}.\text{mo}}(a, e'_t) \} \cup \{ (e_t, e'_t) \mid a \in X_t.E \land G'_{\text{tgt}.\text{mo}}(e_t, a) \} \cup \{ (e'_t, a) \mid a \in X_t.E \land G'_{\text{tgt}.\text{mo}}(e'_t, a) \} \\
\]

We also know that the \( X_t \) and \( X_s \) are related as follows.

\[ X_s.E = X_t.E \]
\[ X_s.\text{po} = \{ (a_s, b_s) \mid M(a_s, a_t) \land M(b_s, b_t) \land X_t.\text{po}(a_t, b_t) \} \]
\[ X_s.\text{rf} = \{ (a_s, b_s) \mid M(a_s, a_t) \land M(b_s, b_t) \land X_t.\text{rf}(a_t, b_t) \} \]
\[ X_s.\text{mo} = \{ (a_s, b_s) \mid M(a_s, a_t) \land M(b_s, b_t) \land X_t.\text{mo}(a_t, b_t) \} \]

**Source Execution Extraction.**

From \( X'_t \) we derive \( X'_s \) and relate \( X'_s \) to \( X_s \)

\[ X'_s.E = X'_t.E = X_t.E \cup \{ e_t, e'_t \} = X_s.E \cup \{ e_t, e'_t \} \]
\[ X'_s.\text{po} = \{ (a_s, b_s) \mid X'_t.\text{po}(a_t, b_t) \land M'(a_s, a_t) \land M'(b_s, b_t) \} \]
\[ \cup \{ (a_s, e'_s) \mid X'_t.\text{po}(a_t, e'_t) \land M'(a_s, a_t) \land M'(e'_s, e'_t) \} \]
\[ \cup \{ (e'_s, b'_s) \mid X'_t.\text{po}(e_t, a_t) \land M'(a_s, a_t) \land M'(e'_s, e'_t) \} \]
\[ \cup \{ (a'_s, b'_s) \mid X'_t.\text{po}(a_t, a_t) \land M'(a'_s, a'_t) \land M'(b'_s, b'_t) \} \]
\[ \cup \{ (a'_s, e'_s) \mid X'_t.\text{po}(a_t, e'_t) \land M'(a'_s, a'_t) \land M'(e'_s, e'_t) \} \]
\[ \cup \{ (e'_s, b'_s) \mid X'_t.\text{po}(e_t, a'_t) \land M'(a'_s, a'_t) \land M'(e'_s, e'_t) \} \]
\[ X'_s.\text{rf} = \{ (a_s, a_s) \mid X'_t.\text{rf}(a_t, b_t) \land M'(a_s, a_t) \land M'(b_s, b_t) \} \]
D. Proofs of Correctness of Reorderings

\[ \implies X'_s, rf = \{(a_s, b_s) \mid X_t, rf(a_t, b_t) \land M'(a_s, b_s) \land M'(b_s, b_t)\} \]
\[ \cup \{(a_s, e'_s) \mid X'_t, rf(a_t, e'_t) \land M'(a_s, a_t) \land M'(e'_s, e'_t)\} \]
\[ \cup \{(a_s, b') \mid X'_t, rf(a_t, b_t) \land M'(a_s, a_t) \land M'(b', a_t)\} \]
\[ \cup \{(e'_s, a_s) \mid X'_t, rf(e'_t, a_t) \land M'(e'_s, e'_t) \land M'(a_s, a_t)\} \]
\[ \cup \{(b', a_s) \mid X'_t, rf(e'_t, b_t) \land M'(b', a_t) \land M'(a_s, a_t)\} \]

\[ \implies X'_s, mo = \{(a_s, b_s) \mid X'_t, mo(a_t, b_t) \land M'(a_s, a_t) \land M'(b_s, b_t)\} \]
\[ \cup \{(a_s, e'_s) \mid X'_t, mo(a_t, e'_t) \land M'(a_s, a_t) \land M'(e'_s, e'_t)\} \]
\[ \cup \{(a_s, b') \mid X'_t, mo(a_t, b_t) \land M'(a_s, a_t) \land M'(b', a_t)\} \]
\[ \cup \{(e'_s, a_s) \mid X'_t, mo(e'_t, a_t) \land M'(e'_s, e'_t) \land M'(a_s, a_t)\} \]
\[ \cup \{(b', a_s) \mid X'_t, mo(e'_t, b_t) \land M'(b', a_t) \land M'(a_s, a_t)\} \]

In this case pc' = pc'[b'.tid \mapsto b'] and hence

\[ S' = S \cup \{e'_s, b'\}. \]

Now we relate \( X'_s \) and \( S' \).

\[ X'_s, E = X_s, E \cup \{e'_s, b'\} = S \cup \{e'_s, b'\} = S' \]

We already have

\[ X'_s, po = X_s, po \]
\[ \cup \{(a_s, e'_s) \mid X'_t, po(a_t, e'_t) \land M'(a_s, a_t) \land M'(e'_s, e'_t)\} \]
\[ \cup \{(a_s, b') \mid X'_t, po(a_t, b_t) \land M'(a_s, a_t) \land M'(e_s, e_t)\} \]
\[ \cup \{(e'_s, b') \mid X'_t, po(e'_t, b_t) \land M'(e'_s, e'_t) \land M'(b', e_t)\} \]

\[ \implies X'_s, po = G_{src}.po \cap (S \times S) \cup \{G'_{src}.po(a_s, e'_s) \mid a_s, e_s \in S'\} \]
\[ \cup \{(a_s, b') \mid a_s, b' \in S'\} \cup \{(e'_s, b') \mid e'_s, b' \in S'\} \]

\[ \implies X'_s, po = G'_{src}.po \cap (S' \times S') \]

We already have

\[ X'_s, rf = X_s, rf \]
\[ \cup \{(a_s, e'_s) \mid X'_t, rf(a_t, e'_t) \land M'(a_s, a_t) \land M'(e'_s, e'_t)\} \]
\[ \cup \{(a_s, b') \mid X'_t, rf(a_t, b_t) \land M'(a_s, a_t) \land M'(b', e_t)\} \]
D.1. Reordering Theorem

\[\{ (e'_s, a_s) \mid X'_s.\text{rf}(e'_s, a_s) \land M'(e'_s, a_s) \land M'(a_s, a_t) \}\]

\[\{ (b', a_s) \mid X'_s.\text{rf}(b', a_s) \land M'(b', a_t) \land M'(a_s, a_t) \}\]

\[\implies X'_s.\text{rf} = G_{src}.\text{rf} \cap (S \times S) \cup \{ G'_{src}.\text{rf}(a_s, e'_s) \mid a_s, e_s \in S' \}\]

\[\cup \{ G'_{src}.\text{rf}(a_s, b') \mid a_s, b' \in S' \}\]

\[\cup \{ G'_{src}.\text{rf}(e'_s, a_s) \mid a_s, e_s \in S' \}\cup \{ G'_{src}.\text{rf}(b', a_s) \mid a_s, b' \in S' \}\]

\[\implies X'_s.\text{rf} = G'_{src}.\text{rf} \cap (S' \times S')\]

We already have

\[X'_{s,\text{mo}} = X_{s,\text{mo}}\]

\[\cup \{ (a_s, e'_s) \mid X'_{s,\text{mo}}(a_s, e'_s) \land M'(a_s, a_t) \land M'(e'_s, e'_t) \}\]

\[\cup \{ (a_s, b') \mid X'_{s,\text{mo}}(a_s, e'_t) \land M'(a_s, a_t) \land M'(b', e'_t) \}\]

\[\cup \{ (e'_s, a_s) \mid X'_{s,\text{mo}}(e'_s, a_s) \land M'(e'_s, e'_t) \land M'(a_s, a_t) \}\]

\[\cup \{ (b', a_s) \mid X'_{s,\text{mo}}(b', e'_t) \land M'(b', e'_t) \land M'(a_s, a_t) \}\]

\[\implies X'_{s,\text{mo}} = G'_{src,\text{mo}} \cap (S \times S)\]

\[\cup \{ G'_{src,\text{mo}}(a_s, e'_s) \mid a_s, e_s \in S' \}\]

\[\cup \{ G'_{src,\text{mo}}(a_s, b') \mid a_s, b' \in S' \}\]

\[\cup \{ G'_{src,\text{mo}}(e'_s, a_s) \mid a_s, e_s \in S' \}\]

\[\cup \{ G'_{src,\text{mo}}(b', a_s) \mid a_s, b' \in S' \}\]

\[\implies X'_{s,\text{mo}} = G'_{src,\text{mo}} \cap (S' \times S')\]

As a result, \(G'_{src} \sim G'_{tgt}\).

**Case** \(e'_t \in G'_{tgt}.E \setminus (A', B')\) **where** \(A' = A\) **and** \(B' = B\):

In this case \(G'_{tgt}.E = G_{tgt}.E \cup \{ e'_t \}\).

In \(G_{src}\) \(e'_s\) is the corresponding event of \(e_t\), that is, \(M(e_s, e_t)\).

We also append corresponding event in \(G_{src}\) and construct \(G'_{src}\).

1. **Condition to show:** \(G'_{src}\) is consistent.

Two possibilities: (1) either \(e_s\) is po-maximal or (2) there exists an event \(e''_s\)

such that \(\text{imm}(G_{src,\text{po}})(e_s, e''_s)\) and \(e''_s\) is \(G_{src,\text{po}}\) maximal.

Let the maximal event be \(e_m\).

We append an event \(e'_t\) in \(G_{src}\) by po-extending from \(e_m\) and create \(G'_{src}\) such that
D. Proofs of Correctness of Reorderings

\[ G'_{\text{src}} \cdot E = G_{\text{src}} \cdot E \uplus \{ e' \} \]
\[ G'_{\text{src}} \cdot \text{po} = (G_{\text{src}} \cdot \text{po} \uplus \{ (e_m, e'_s) \})^+ \]
\[ G'_{\text{src}} \cdot \text{jf} = G_{\text{src}} \cdot \text{jf} \uplus \{ (w_s, e'_s) | (w_s, e'_s) \in (G'_{\text{src}} \cdot W \times G'_{\text{src}} \cdot R) \}
\quad \land M(w_s, w_t) \land G'_{\text{tgt}} \cdot \text{jf}(w_t, e'_t) \land \neg G'_{\text{src}} \cdot \text{cf}(w_s, e'_s) \}
\]
\[ G'_{\text{src}} \cdot \text{mo} = G_{\text{src}} \cdot \text{mo} \uplus \{ (w_s, e'_s) | (w_s, e'_s) \in (G'_{\text{src}} \cdot W \times G'_{\text{src}} \cdot W) \}
\quad \land M(w_s, w_t) \land G'_{\text{tgt}} \cdot \text{mo}(w_t, e_t) \land \neg G'_{\text{src}} \cdot \text{cf}(w_s, e'_s) \}
\]
\[ G'_{\text{src}} \cdot \text{ew} = G_{\text{src}} \cdot \text{ew} \uplus \{ (w_s, e'_s), (e'_s, w_s) | (w_s, e'_s) \in (G'_{\text{src}} \cdot W \times G'_{\text{src}} \cdot W) \}
\quad \land M(w_s, w_t) \land G'_{\text{tgt}} \cdot \text{ew}(w_t, e_t) \}
\]

Also in this case \( M' = M \uplus \{ (e'_s, e'_t) \} \).

Now we check whether \( G'_{\text{src}} \) is consistent.

We know \( G_{\text{src}}, G'_{\text{tgt}} \) are consistent hence satisfy (ICFJ). Hence from definition of \( G'_{\text{src}} \) and \( M' \) we know that \( G'_{\text{src}} \) satisfies (ICF).

We know \( G_{\text{src}}, G'_{\text{tgt}} \) are consistent hence satisfy (ICF). Hence following the definition of \( G'_{\text{src}} \) and \( M' \) we know \( G'_{\text{src}} \) preserves (ICF).

We know that \( G_{\text{src}} \) preserves (CF), (CFJ), (VISJ). Also \( G'_{\text{tgt}} \cdot \text{jf}(w_t, e'_t) \) implies \( w_t \in \text{vis}(G'_{\text{tgt}}) \) and \( \neg G'_{\text{tgt}} \cdot \text{ef}(w_t, e'_t) \), and \( M(w_s, w_t) \) holds. Following the construction, \( w_s \in \text{vis}(G'_{\text{src}}) \) as well as \( \neg G'_{\text{src}} \cdot \text{ef}(w_s, e'_s) \) hold. Hence \( G'_{\text{src}} \) preserves (CF), (CFJ), (VISJ).

We know \( G_{\text{src}} \) preserves (COH'). Consider there is \( (G'_{\text{src}} \cdot \text{hb}; G'_{\text{src}} \cdot \text{eco}) \) cycle in \( G'_{\text{src}} \) and \( e'_s \) is a part of this cycle. In that case there is a \( (G'_{\text{tgt}} \cdot \text{hb}; G'_{\text{tgt}} \cdot \text{eco}) \) cycle in \( G'_{\text{tgt}} \) and \( e'_t \) is a part of the cycle. However, \( G'_{\text{tg})) \) preserves (COH) and hence there is no \( (G'_{\text{tg)}, \text{hb}; G'_{\text{tg}} \cdot \text{eco}) \) cycle. Hence a contradiction and \( G'_{\text{src}} \) preserves (COH').

We know \( G'_{\text{src}} \) preserves (NCFU) and (NCFSC). Consider \( G'_{\text{src}} \) violates (NCFU) or (NCFSC) in that case \( G'_{\text{src}} \) violates (NCFU) or (NCFSC) due to \( e'_s \). However, following the construction of \( G'_{\text{src}} \), in this case, \( G'_{\text{tgt}} \) also violates (NCFU) or (NCFSC) due to \( e'_t \). This is not possible as \( G'_{\text{tgt}} \) is consistent. Hence a contradiction and \( G'_{\text{src}} \) preserves (NCFU) and (NCFSC).

As a result, \( G'_{\text{src}} \) is consistent.

Thus finally \( M' = M \uplus \{ (e'_s, e'_t) \} \) and \( \text{pc}' = \text{pc}[e_s, \text{tid} \mapsto e'_s] \).

2. Condition to show: the simulation invariant holds between \( G'_{\text{src}} \) and \( G'_{\text{tgt}} \)

a) \( \forall e_t \in G'_{\text{tg}}, E \setminus (A' \cup B'), \exists e_s \in G'_{\text{src}}, E, M'(e_s, e_t) \)
D.1. Reordering Theorem

We know this condition holds in $G_{\text{src}}$ and $G_{\text{tgt}}$. Considering the definitions of $G'_{\text{src}}$, $G'_{\text{tgt}}$, and $M'$, the condition holds between $G'_{\text{src}}$ and $G'_{\text{tgt}}$ as $M'(e', e'_t)$ holds.

b) $\forall c_t \in G'_{\text{tgt}}.E \setminus (A' \cup B'), a_t \in A', b_t \in B' \land G'_{\text{tgt}}.po(c_t, b_t) \implies \\
\exists c_s, a_s, b_s \in G'_{\text{src}}.E, M'(c_s, c_t) \land M'(a_s, a_t) \land M'(b_s, b_t) \\
\land (\exists a'' \in G'_{\text{src}}.E, a_s.\text{loc} = a''.\text{loc} \land a_s.\text{ord} = a''.\text{ord} \\
\land G'_{\text{src}}.po(c_s, a'') \land \text{imm}(G'_{\text{src}}.po)(a'', b_s))$

We know this condition holds in $G_{\text{src}}$ and $G_{\text{tgt}}$. Considering the definitions of $G'_{\text{src}}$, $G'_{\text{tgt}}$, and $M'$, when $c_t = e'_t$ then $c_t$ is $G'_{\text{tgt}}.po$-maximal and there is no $G'_{\text{tgt}}.po(c_t, b_t)$. Hence the condition holds between $G'_{\text{src}}$ and $G'_{\text{tgt}}$.

c) $\forall c_t \in G'_{\text{tgt}}.E \setminus (A' \cup B'), a_t \in A' \land G'_{\text{tgt}}.po(c_t, a_t) \implies \\
\exists c_s, a_s \in G'_{\text{src}}.E, M'(c_s, c_t) \land M'(a_s, a_t) \land G'_{\text{src}}.po(c_s, a_s)$

We know this condition holds in $G_{\text{src}}$ and $G_{\text{tgt}}$. Considering the definitions of $G'_{\text{src}}$, $G'_{\text{tgt}}$, and $M'$, when $c_t = e'_t$ then $c_t$ is $G'_{\text{tgt}}.po$-maximal and there is no $G'_{\text{tgt}}.po(c_t, a_t)$. Hence the condition holds between $G'_{\text{src}}$ and $G'_{\text{tgt}}$.

d) $\forall a_t \in A', b_t \in B', \text{imm}(G'_{\text{tgt}}.po)(b_t, a_t) \implies \\
(\exists c_t \in G'_{\text{tgt}}.E \setminus (A' \cup B'), a', b_s, c_s \in G'_{\text{src}}.E, M'(c_s, c_t) \land M'(a_s, a_t) \land M'(b_s, b_t) \\
\land \text{imm}(G'_{\text{tgt}}.po)(c_t, b_t) \land \text{imm}(G'_{\text{src}}.po)(c_s, a_s) \land \text{imm}(G'_{\text{src}}.po)(a_s, b') \\
\land G'_{\text{src}}.cf(a_s, a') \land \text{imm}(G'_{\text{src}}.po)(a', b_s) \land b_s.\text{loc} = b'.\text{loc} \land b_s.\text{ord} = b'.\text{ord} \\
\land G''_{\text{src}}.e\text{w}(b_s, b'))$

We know this condition holds in $G_{\text{src}}$ and $G_{\text{tgt}}$. Considering the definitions of $G'_{\text{src}}$, $G'_{\text{tgt}}$, $M'$, $e'_t \notin (A' \cup B')$. As a result, this condition holds between $G'_{\text{src}}$ and $G'_{\text{tgt}}$.

e) $\forall c_t \in G'_{\text{tgt}}.E \setminus (A' \cup B'), b_t \in B', G'_{\text{tgt}}.po(b_t, c_t) \implies \\
\exists b_s, b'_s, c_s \in G'_{\text{src}}.E, M'(c_s, c_t) \land M'(b_s, b_t) \land M'(b'_s, b_t) \\
\land G'_{\text{src}}.e\text{w}(b_s, b'_s) \land (G'_{\text{src}}.po(b_s, c_s) \lor G'_{\text{src}}.po(b'_s, c_s))$

We know this condition holds in $G_{\text{src}}$ and $G_{\text{tgt}}$. We consider two cases for $e_t$.

case $e_t \in G'_{\text{tgt}}.E \setminus (A' \cup B')$: In this case there exists $b_t$ such that $G_{\text{tgt}}.po(b_t, e_t)$. Hence $G_{\text{tgt}}.po(e, e'_t)$ implies $G_{\text{tgt}}.po(b_t, e'_t)$ and the condition holds.

case $e_t \in A'$: In this case there exists an event $e''_s$ such that $\text{imm}(G'_{\text{src}}.po)(e_s, e''_s)$ where $M'(e''_s, b_t)$ and $b_t \in B'$ and $\text{imm}(G_{\text{tgt}}.po)(b_t, e_t)$. Thus the condition holds between $G'_{\text{src}}$ and $G'_{\text{tgt}}$. 

95
D. Proofs of Correctness of Reorderings

f) \[
\forall a_t \in A', b_t \in B', G'_{tgt}.po(b_t, a_t) \implies \\
\exists a_s, b_s \in G'_{src}.E. M'(a_s, a_t) \land M'(b_s, b_t) \land -G'_{src}.po(b_s, a_s)
\]

We know this condition holds in \(G_{src}\) and \(G_{tgt}\). Considering the definitions of \(G'_{src}\), \(G'_{tgt}\), \(M'\), \(G'_{tgt}\), \(G'_{src}\), \(G'_{tgt}\) and \(G'_{src}\), we see that this condition holds between \(G'_{src}\) and \(G'_{tgt}\).

g) \[
\forall c_t, c'_t \in G'_{tgt}.E \setminus (A' \cup B'), G'_{tgt}.po(c_t, c'_t) \implies \\
\exists c_s, c'_s \in G'_{src}.E. M'(c_s, c_t) \land M'(c'_s, c'_t) \land G'_{src}.po(c_s, c'_s)
\]

We know the condition holds between \(G_{src}\) and \(G_{tgt}\). Considering the definitions of \(G'_{src}\), \(G'_{tgt}\), \(M'\), this condition holds between \(G'_{src}\) and \(G'_{tgt}\) where \(c'_t = e'_t\).

h) \[
\forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), a_t \in A', G'_{tgt}.jf(c_t, a_t) \implies \\
\exists c_s, a_s \in G'_{src}.E. M'(a_s, a_t) \land M'(c_s, c_t) \land G'_{src}.jf(c_s, a_s)
\]

We know the condition holds between \(G_{src}\) and \(G_{tgt}\). Considering the definitions of \(G'_{src}\), \(G'_{tgt}\), \(M'\), the condition holds between \(G'_{src}\) and \(G'_{tgt}\) for \(c_t = e'_t\) where there is no outgoing \(G'_{tgt}.jf\) edge from \(e'_t\).

i) \[
\forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), a_t \in A', G'_{tgt}.jf(a_t, c_t) \implies \\
\exists c_s, a_s \in G'_{src}.E. M'(a_s, a_t) \land M'(c_s, c_t) \land G'_{src}.jf(a_s, c_s)
\]

We know the condition holds between \(G_{src}\) and \(G_{tgt}\). Considering the definitions of \(G'_{src}\), \(G'_{tgt}\), \(M'\), the condition holds between \(G'_{src}\) and \(G'_{tgt}\) for \(c_t = e'_t\).

j) \[
\forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), b_t \in B', G'_{tgt}.jf(b_t, c_t) \implies \\
\exists b_s, c_s \in G'_{src}.E. M'(b_s, b_t) \land M'(c_s, c_t) \land G'_{src}.jf(b_s, c_s)
\]

We know the condition holds between \(G_{src}\) and \(G_{tgt}\). Considering the definitions of \(G'_{src}\), \(G'_{tgt}\), \(M'\), the condition holds between \(G'_{src}\) and \(G'_{tgt}\) for \(c_t = e'_t\) where there is no outgoing \(G'_{tgt}.jf\) edge from \(e'_t\).

k) \[
\forall c_t \in G'_{tgt}.E \setminus (A' \cup B'), b_t \in B', G'_{tgt}.jf(b_t, c_t) \implies \\
((\exists b_s, c_s \in G'_{src}.E. M'(b_s, b_t) \land \exists b'_t \in G'_{src}.E. M'(b'_t, b_t) \land G'_{src}.ew(b_s, b'_t) \implies \\
G'_{src}.jf(b_s, c_s)) \\
\land (\exists b'_t, b_s, c_s \in G'_{src}.E. (M'(b'_t, b_t) \land M'(b_s, b'_t) \land M'(c_s, c_t) \land G'_{src}.ew(b_s, b'_t)) \implies \\
G'_{src}.jf(b'_t, c_s)))
\]

We know the condition holds between \(G_{src}\) and \(G_{tgt}\). Considering the definitions of \(G'_{src}\), \(G'_{tgt}\), \(M'\), the condition holds between \(G'_{src}\) and \(G'_{tgt}\) for \(c_t = e'_t\).
D.1. Reordering Theorem

1) \[ \forall c_t, c'_t \in G'_t.g.t. E \setminus (A' \cup B'), G'_t.g.t. jf(c_t, c'_t) \implies \exists c_s, c'_s \in G'_s.g.s. M'(c_s, c_t) \land M'(c'_s, c'_t) \land G'_s.g.s. jf(c_s, c'_s) \]

We know the condition holds between \( G'_s \) and \( G'_t \). Considering the definitions of \( G'_s, G'_t, M' \), (1) this condition holds between \( G'_s \) and \( G'_t \) where \( c'_t = e'_t \). (2) the condition also holds when \( c_t = e'_t \) as in that case there is no outgoing edge from \( e'_t \).

m) \[ \forall c_t \in G'_t.g.t. E \setminus (A' \cup B'), a_t \in A', b_t \in B'. G'_t.g.t. mo(c_t, a_t) \implies \exists c_s, a_s \in G'_s.g.s. E. M'(c_s, c_t) \land M'(a_s, a_t) \land G'_s.g.s. mo(c_s, a_s) \]

We know the condition holds between \( G'_s \) and \( G'_t \). Considering the definitions of \( G'_s, G'_t, M' \), for \( c_t = e'_t \) the condition holds between \( G'_s \) and \( G'_t \).

n) \[ \forall c_t \in G'_t.g.t. E \setminus (A' \cup B'), a_t \in A', b_t \in B'. G'_t.g.t. mo(a_t, c_t) \implies \exists c_s, a_s \in G'_s.g.s. E. M'(c_s, c_t) \land M'(a_s, a_t) \land G'_s.g.s. mo(c_s, a_s) \]

We know the condition holds between \( G'_s \) and \( G'_t \). Considering the definitions of \( G'_s, G'_t, M' \), for \( c_t = e'_t \) the condition holds between \( G'_s \) and \( G'_t \).

o) \[ \forall c_t \in G'_t.g.t. E \setminus (A' \cup B'), b_t \in B'. G'_t.g.t. mo(c_t, b_t) \implies \exists c_s, b_s \in G'_s.g.s. E. M'(c_s, c_t) \land M'(b_s, b_t) \land G'_s.g.s. mo(c_s, b_s) \]

We know the condition holds between \( G'_s \) and \( G'_t \). Considering the definitions of \( G'_s, G'_t, M' \), for \( c_t = e'_t \) the condition holds between \( G'_s \) and \( G'_t \).

p) \[ \forall c_t \in G'_t.g.t. E \setminus (A' \cup B'), b_t \in B'. G'_t.g.t. mo(b_t, c_t) \implies \exists c_s, b_s \in G'_s.g.s. E. M'(c_s, c_t) \land M'(b_s, b_t) \land G'_s.g.s. mo(b_s, c_s) \]

We know the condition holds between \( G'_s \) and \( G'_t \). Considering the definitions of \( G'_s, G'_t, M' \), for \( c_t = e'_t \) the condition holds between \( G'_s \) and \( G'_t \).

q) \[ \forall c, c' \in G'_t.g.t. E \setminus (A' \cup B'), G'_t.g.t. mo(c_t, c'_t) \implies \exists c_s, c'_s \in G'_s.g.s. E. M'(c_s, c_t) \land M'(c'_s, c'_t) \land G'_s.g.s. mo(c_s, c'_s) \]

We know the condition holds between \( G'_s \) and \( G'_t \). Considering the definitions of \( G'_s, G'_t, M' \), for \( c_t = e'_t \) or \( c'_t = e'_t \) the condition holds between \( G'_s \) and \( G'_t \).
D. Proofs of Correctness of Reorderings

\[ \forall o_s \in G'_s, \forall o_t \in G'_t. (\exists a_s, b_t \in G'_s, e. \ M'(o_s, o_t)) \implies \exists o'_s, o'_t \in G'_s, G'_t. m'o(o_s, o'_t) \]

We know the condition holds between \( G'_s \) and \( G'_t \). Following the definitions of \( G'_s \) and \( G'_t \), \( M', M'(o'_s, o'_t) \) holds. Hence the condition holds between \( G'_s \) and \( G'_t \).

3. Condition to show:

there exists \( pc' \) such that

\[ X'_t, E = S' \]
\[ X'_s, po = G'_s, po \cap (S' \times S') \]
\[ X'_s, rf = G'_s, rf \cap (S' \times S') \]
\[ X'_s, mo = G'_s, mo \cap (S' \times S') \]

where \( S'(G'_s, pc') \triangleq \{ e | e \in G'_s, E \land G'_s, po(e, pc'(e, tid)) \} \).

If \( e'_t \notin X'_t \) then \( X'_t = X_t \). In this case \( pc' = pc', S' = S, \) and \( X'_s = X_s \).

Otherwise, when \( e'_t \in X'_t \) then \( X'_t \) is an extension of \( X_t \), that is,

\[ X'_t, E = X_t, E \cup \{ e'_t \} \]
\[ X'_s, po = (X_t, po \cup \{ (a, e'_t) | a \in X_t, E \land G'_t, po(a, e'_t) \} \cup \{ (e_t, e'_t) \})^+ \]
\[ X'_s, rf = X_t, rf \cup \{ (a, e'_t) | a \in X_t, E \land G'_t, rf(a, e'_t) \} \]
\[ \cup \{ (e'_t, e_t) | a \in X_t, E \land G'_t, rf(e'_t, a) \} \]
\[ X'_s, mo = X_t, mo \cup \{ (a, e'_t) | a \in X_t, E \land G'_t, mo(a, e'_t) \} \]
\[ \cup \{ (e'_t, a) | a \in X_t, E \land G'_t, mo(e'_t, a) \} \]

We also know that the \( X_t \) and \( X_s \) are related as follows.

\[ X_t, E = X_s, E \]
\[ X_s, po = \{ (a_s, b_t) | M(a_s, a_t) \land M(b_s, b_t) \land X_t, po(a_t, b_t) \land \neg (a_t \in A \land b_t \in B) \} \]
\[ \cup \{ (a_s, b_t) | M(a_s, a_t) \land M(b_s, b_t) \land X_t, po(b_t, a_t) \land (a_t \in A \land b_t \in B) \} \]
\[ X_t, rf = \{ (a_s, b_t) | M(a_s, a_t) \land M(b_s, b_t) \land X_t, rf(a_t, b_t) \} \]
\[ X_s, mo = \{ (a_s, b_s) | M(a_s, a_t) \land M(b_s, b_t) \land X_t, mo(a_t, b_t) \}\]
D.1. Reordering Theorem

Source Execution Extraction.

From $X'_t$ we derive $X'_s$ and relate $X'_s$ to $X_s$

$$X'_s.E = X'_t.E = X_t.E \cup \{e_t, e'_t\} = X_s.E \cup \{e_t, e'_t\}$$

$$X'_s.po = \{(a_s, b_s) \mid X'_s.po(a_t, b_t) \land M'(a_s, a_t) \land M'(b_s, b_t)$$

$$\land \neg (a_t \in A' \land b_t \in B')\}$$

$$\cup \{(a_s, b_s) \mid M(a_s, a_t) \land M(b_s, b_t) \land X'_t.po(b_t, a_t) \land (a_t \in A' \land b_t \in B')\}$$

$$\implies X'_s.po = X_s.po \cup \{(a_s, e'_s) \mid X'_t.po(a_t, e'_s) \land M'(a_s, a_t) \land M'(e'_s, e'_t)\}$$

$$X'_s.rf = \{(a_s, b_s) \mid X'_t.rf(a_t, b_t) \land M'(a_s, a_t) \land M'(b_s, b_t)\}$$

$$\implies X'_s.rf = \{(a_s, b_s) \mid X'_t.rf(a_t, b_t) \land M'(a_s, a_t) \land M'(b_s, b_t)\}$$

$$\cup \{(a_s, e'_s) \mid X'_t.rf(a_t, e'_s) \land M'(a_s, a_t) \land M'(e'_s, e'_t)\}$$

$$\cup \{(e'_s, a_s) \mid X'_t.rf(e'_s, a_t) \land M'(e'_s, e'_t) \land M'(a_s, a_t)\}$$

$$\implies X'_s.rf = X_s.rf$$

$$\cup \{(a_s, e'_s) \mid X'_t.rf(a_t, e'_s) \land M'(a_s, a_t) \land M'(e'_s, e'_t)\}$$

$$\cup \{(e'_s, a_s) \mid X'_t.rf(e'_s, a_t) \land M'(e'_s, e'_t) \land M'(a_s, a_t)\}$$

$$\implies X'_s.mo = X'_s.mo$$

$$\cup \{(a_s, e'_s) \mid X'_t.mo(a_t, e'_s) \land M'(a_s, a_t) \land M'(e'_s, e'_t)\}$$

$$\cup \{(e'_s, a_s) \mid X'_t.mo(e'_s, a_t) \land M'(e'_s, e'_t) \land M'(a_s, a_t)\}$$

In this case $pc' = pc[e'_s,tid \mapsto e'_t]$ and hence $S' = S \cup \{e'_s\}$.

Now we relate $X'_s$ and $S'$.

$$X'_s.E = X_s.E \cup \{e'_s\} = S \cup \{e'_s\} = S'$$

We already have

$$X'_s.po = (X_s.po \cup \{(a_s, e'_s) \mid X'_t.po(a_t, e'_s) \land M'(a_s, a_t) \land M'(e'_s, e'_t)\})^+$$

$$\implies X'_s.po = G_{src}.po \cap (S \times S) \cup \{G'_{src}.po(a_s, e'_s) \mid a_s, e_s \in S'\}$$

$$\implies X'_s.po = G'_{src}.po \cap (S' \times S')$$

We already have

$$X'_s.rf = X_s.rf$$

$$\cup \{(a_s, e'_s) \mid X'_t.rf(a_t, e'_s) \land M'(a_s, a_t) \land M'(e'_s, e'_t)\}$$

$$\cup \{(e'_s, a_s) \mid X'_t.rf(e'_s, a_t) \land M'(e'_s, e'_t) \land M'(a_s, a_t)\}$$

$$\implies X'_s.rf = G_{src}.rf \cap (S \times S) \cup \{G'_{src}.rf(a_s, e'_s) \mid a_s, e_s \in S'\}$$

$$\cup \{G'_{src}.rf(e'_s, a_s) \mid a_s, e_s \in S'\}$$

$$\implies X'_s.rf = G'_{src}.rf \cap (S' \times S')$$

99
D. Proofs of Correctness of Reorderings

We already have

\[ X'_{s}.mo = X_{s}.mo \]
\[ \cup \{(a_{s}, e'_{s}) | X'_{t}.mo(a_{t}, e'_{t}) \land M'(a_{s}, a_{t}) \land M'(e_{s}, e'_{t})\} \]
\[ \cup \{(e'_{s}, a_{s}) | X'_{t}.mo(e'_{t}, a_{t}) \land M'(e_{s}, e'_{t}) \land M'(a_{s}, a_{t})\} \]

\[ \Rightarrow X'_{s}.mo = G_{src}.mo \cap (S \times S) \cup \{G'_{src}.mo(a_{s}, e'_{s}) | a_{s}, e_{s} \in S'\} \]
\[ \cup \{G'_{src}.mo(e'_{s}, a_{s}) | a_{s}, e_{s} \in S'\} \]

\[ \Rightarrow X'_{s}.mo = G'_{src}.mo \cap (S' \times S') \]

As a result, \( G'_{src} \sim G'_{tgt} \).

Thus we complete the construction of the source event structure \( G_{src} \) and the source execution \( X_{s} \) can be extracted from \( G_{src} \), that is, \( X_{s} \in \text{ex}_{\text{WEAKESTMO}}(G_{src}) \).
E. Proofs of Correctness of Eliminations

We restate the definition of compilation correctness and the safe elimination theorem.

Definition 8. A transformation of program $P_{src}$ in memory model $M_{src}$ to program $P_{tgt}$ in model $M_{tgt}$ is correct if it does not introduce new behaviors:

i.e., $\text{Behavior}_{M_{tgt}}(P_{tgt}) \subseteq \text{Behavior}_{M_{src}}(P_{src})$.

Theorem 7. The eliminations in Figure 7.1 are correct in both WEAKESTMO models.

The safe eliminations from Figure 7.1 are

Definition 11. $\text{elim}(P_{src}, P_{tgt})$ such that $P_{tgt}(i) \subseteq P_{src}(i) \cup \{\tau \cdot \alpha \cdot \tau' | \tau \cdot \alpha \cdot \tau' \in P_{src}(i)\} \land \forall j \neq i. P_{tgt}(j) = P_{src}(j)$

where $\alpha$ is a label of shared memory accesses or fences.

Then The formal statement is as follows:

$\forall P_{src}. \text{elim}(P_{src}, P_{tgt}) \implies$

$\forall G_{tgt}. G_{init} \rightarrow P_{tgt.\text{WEAKESTMO}}^\ast G_{tgt}. \exists G_{src}. G_{init} \rightarrow P_{src.\text{WEAKESTMO}}^\ast G_{src} \land$

$\forall X_t \in \text{ex}_{\text{WEAKESTMO}}(G_{tgt}). \exists X_s \in \text{ex}_{\text{WEAKESTMO}}(G_{src}). \text{Behavior}(X_t) = \text{Behavior}(X_s)$

$\land X_t.\text{Race} \cap E_{\text{NA}} \neq \emptyset \implies X_s.\text{Race} \cap E_{\text{NA}} \neq \emptyset$

To prove the theorem, we construct a source event structure following a given target event structure. Then, for an extracted consistent target execution we extract a source execution from the source event structure. Then we show that the source execution is consistent and source and target execution has same behavior. Finally, we show race preservation: if target is racy, then the source execution is also racy. As a result, if the target execution has undefined behavior due to a data race, so does the source execution.

Now we study various safe eliminations.

E.1. Overwritten Write (OW)

Proof. Recall the relationship between the two programs for the thread $i$ affected by the transformation:

$P_{tgt}(i) \subseteq P_{src}(i) \cup \{\tau \cdot St_o(x, v) \cdot \tau' | \tau \cdot St'_o(x, v') \cdot St_o(x, v) \cdot \tau' \in P_{src}(i) \land \sigma' \subseteq \sigma\}$

For all other threads $j \neq i$, we have $P_{tgt}(j) = P_{src}(j)$. Assume we have a target event structure, $G_{tgt}$, and an execution, $X_t \in \text{ex}_{\text{WEAKESTMO}}(G_{tgt})$, extracted from it.
E. Proofs of Correctness of Eliminations

Let \( W \) be the set of stores of thread \( i \) of \( G_{tgt} \) with label \( St_o(x, v) \), and whose po-prefix has some sequence of labels \( \tau \) such that \( \tau \cdot St_o(x, v) \notin P_{src}(i) \). Then, because of the relationship between the two programs, we know that for each such \( w \in W \), \( \tau \cdot St_o(x, v') \cdot St_o(x, v) \in P_{src}(i) \) for the appropriate \( \tau \). Let \( C \) be the immediate \( G_{tgt} \) po-predecessors of the events in \( W \).

**Source Event Structure Construction.** To construct \( G_{src} \), we follow the construction steps of \( G_{tgt} \). For each target construction step that adds event \( e \) to \( G_{tgt} \) to get \( G_{tgt}' \), we perform one or more corresponding steps going from \( G_{src} \) to \( G_{src}' \). We do a case analysis on the event \( e \) of the target event structure.

**Case** \( e \notin W \): In this case, we append event \( e \) to the source event structure as follows:

\[
\begin{align*}
G_{src}' & = G_{src} \cup \{e\} \\
G_{src}.po & = (G_{src}.po \cup \{(a, e) \mid a \in \text{dom}(C'_{tgt}.po; [e])\})^+ \\
G_{src}.jf & = C'_{tgt}.jf \\
G_{src}' .mo & = G'_{tgt}.mo \cup \text{imm}(G_{src}.po); [W]; G_{tgt}.mo \cup G'_{tgt}.mo; [W]; \text{imm}(G_{src}.po^{-1}) \\
G_{src}.ew & = C'_{tgt}.ew
\end{align*}
\]

Now we check the consistency of \( G_{src}' \). We already know that \( G_{src} \) and \( G_{tgt}' \) are consistent. Following the construction of \( G_{src}' \), the (CF), (CFJ), (VISJ), (ICF), (ICFJ) constraints immediately hold.

Now we show \( G_{src}' \) satisfies (COH'). From the definition, there is no \( G_{src}.hb; G_{src}.eco \) as well as \( G_{tgt}.hb; G_{tgt}.eco \) cycle. Compared to \( G_{src} \) and \( G_{tgt}' \), the additional \( G_{src}.mo \) edges are from and to events the deleted events.

Let \( d \in (G_{src}' \setminus G_{tgt}.E) \) be such a deleted event. Assume the \( mo \) edges to or from \( d \) creates a \( G_{src}.mo \) cycle. However, for each \( G_{src}.mo(d, e) \) or \( G_{src}.mo(e, d) \) already there exists \( G_{src}.mo(w, e) \) or \( G_{src}.mo(e, w) \) respectively where \( w \in W \) and \( \text{imm}(G_{src}.po(d, w)) \). Thus event \( e \) results no new \( G_{src}.hb; G_{src}.eco \) cycle and hence \( G_{src}' \) satisfies (COH').

We know \( G_{src} \) preserves (NCFU) and (NCFSC). Consider \( G_{src}' \) violates (NCFU) or (NCFSC). In that case \( G_{src}' \) violates (NCFU) or (NCFSC) due to \( e \). However, following the construction of \( G_{src}' \), this case, \( G_{tgt}' \) also violates (NCFU) or (NCFSC). This is not possible as \( G_{tgt}' \) is consistent. Hence a contradiction and \( G_{src}' \) preserves (NCFU) and (NCFSC).

Hence \( G_{src}' \) is consistent.

**Case** \( e \in W \): In this case, we first append a new event \( d \) with \( d.\text{lab} = St_o(x, v') \) and then the event \( e \) to \( G_{src} \) as follows:

\[
\begin{align*}
G_{src}' & = G_{src} \cup \{d, e\} \quad \text{where} \quad d.\text{lab} = St_o(x, v') \\
G_{src}.po & = (G_{src}.po \cup \{(d, e) \mid (c, d) \in G_{tgt}.po\})^+ \\
G_{src}.jf & = G_{tgt}.jf \\
G_{src}' .mo & = G'_{tgt}.mo \cup \{(d, a) \mid G_{tgt}.mo(e, a)\} \cup \{(a, d) \mid G_{tgt}.mo(a, e)\} \cup \{(d, e)\} \\
G_{src}.ew & = G_{tgt}.ew
\end{align*}
\]
E.1. Overwritten Write (OW)

Now we check the consistency of $G'_{src}$. We already know that $G_{src}$ and $G'_{tgt}$ is consistent. Following the construction of $G'_{src}$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ) constraints immediately hold.

Now we show $G'_{src}$ satisfies (COH'). From the definition, there is no $G_{src}.hb; G_{src}.eco'$ as well as $G'_{tgt}.hb; G'_{tgt}.eco'$ cycle. Compared to $G_{src}$ and $G'_{tgt}$, the additional $G'_{src}.mo$ edges are from and to the event $d$. Assume the $mo$ edges to or from $d$ creates a $G'_{src}.hb; G'_{src}.eco'$ cycle.

However, for each $G'_{src}.mo(d, a)$ or $G'_{src}.mo(a, d)$ already there exists $G''_{src}.mo(w, e)$ or $G'_{src}.mo(e, w)$ respectively where $a \neq e$. Thus event $e$ results no new $G'_{src}.hb; G'_{src}.eco'$ cycle and hence $G'_{src}$ satisfies (COH').

We know $G_{src}$ preserves (NCFU) and (NCFSC). Consider $G'_{src}$ violates (NCFU) or (NCFSC). In that case $G'_{src}$ violates (NCFU) or (NCFSC) due to $d$ or $e$. However, following the construction of $G'_{src}$, in this case, $G'_{tgt}$ also violates (NCFU) or (NCFSC). This is not possible as $G'_{tgt}$ is consistent. Hence a contradiction and $G'_{src}$ preserves (NCFU) and (NCFSC).

Hence $G'_{src}$ is consistent.

Source Execution Construction. Next, we construct an execution $X_t \in \text{ex}_\text{WEAKESTMO}(G_{tgt})$.

If $W \subseteq (G_{tgt}.E \setminus X_t, E)$, then we find the corresponding execution $X_s \in \text{ex}_\text{WEAKESTMO}(G_{src})$ such that $X_s$ contains no event created for $S_{tr}(x, v')$. Else if an event $w_t \in W$ is in $X_t$, then we know that we can find an execution with $w_s \in X_s, E$ and $X_s, E$ also contains an event $w'$ corresponding to $store_{tr}(x, v')$. Thus $X_s$ is as follows.

$$X_s.E = X_t.E \uplus \{d \mid X_s.E \cap W \neq \emptyset\}$$

$$X_s.po = (X_t.po \uplus \{(c, d), (d, w) \mid (c, w) \in \text{imm}(X_t.po) \cap (C \times W) \land d \in (G_{src}.E \setminus G_{tgt}.E))\}^+$$

$$X_s.rf = X_t.rf$$

$$X_s.mo = (X_t.mo \uplus \{(d, w) \mid (d, w) \in ((G_{src}.E \setminus G_{tgt}.E) \times W)\}$$

$$\uplus \{(a, d) \mid X_t.mo(a, w) \land (d, w) \in ((G_{src}.E \setminus G_{tgt}.E) \times W) \cap \text{imm}(G_{src}.po)\}$$

$$\uplus \{(d, a) \mid X_t.mo(a, w) \land (d, w) \in ((G_{src}.E \setminus G_{tgt}.E) \times W) \cap \text{imm}(G_{src}.po)\}$$

Source Execution Consistency. Now we check the consistency of $X_s$.

Since $X_t$ is consistent, the (Well-formed), (total-MO), (Coherence), (Atomicity) constraints also hold for $X_s$. The (SC) constraint is affected only when $o = o' = sc$, in which case the new events introduce some [SC], $X_s.po_z$. [SC] edges. These edges, however, can create a $(X_s.psc_{base} \cup X_s.psc_F)$ cycle only when there is a $(X_t.psc_{base} \cup X_t.psc_F)$ cycle. Since $X_t$ is consistent there is no $(X_t.psc_{base} \cup X_t.psc_F)$ cycle. Hence, $X_s$ satisfies (SC) and, as a result, $X_s$ is consistent.

Same Behavior. For locations $y \neq x$, we have $X_s.E_y = X.E_y$ and as a result $\text{Behavior}(X_s)|_y = \text{Behavior}(X_t)|_y$ trivially holds. Now we check whether $\text{Behavior}(X_s)|_x = \text{Behavior}(X_t)|_x$ holds. Note that any newly introduced event $d \in X_s.E \setminus X_t.E$ is not $X_s.mo$ maximal, because in that case there exists $w \in W$ such that $X_s.mo(d, w)$. Hence $\text{Behavior}(X_s) = \text{Behavior}(X_t)$ holds.
Race Preservation. Moreover, if $X_i$ is racy, then the new write $d$ does not introduce any $X_{sw_{C11}}$ edge in $X_s$. Hence $X_s$ is also racy. As a result, if the target execution has undefined behavior due to a data race, so does the source execution. 

E.2. Read after Write (RAW) / Read after Update (RAU)

Proof. Recall the relationship between the two programs for the thread $i$ affected by the transformation:

$$P_{tgt}(i) \subseteq P_{src}(i) \cup \{ \tau \cdot St_o(x,v) \cdot \tau' \mid \tau \cdot St_o(x,v) \cdot Ld_o(x,\_)
\cdot \tau' \in P_{src}(i) \land o' \subseteq o \}$$

or

$$P_{tgt}(i) \subseteq P_{src}(i) \cup \{ \tau \cdot U_o(x, v', v) \cdot \tau' \mid \tau \cdot U_o(x, v', v) \cdot Ld_o(x,\_)
\cdot \tau' \in P_{src}(i) \land o' \subseteq o \}$$

For all other threads $j \neq i$, we have $P_{tgt}(j) = P_{src}(j)$. Assume we have a target event structure, $G_{tgt}$, and an execution, $X_t \in \text{exWEAKESTMO}(G_{tgt})$, extracted from it.

Let $W$ be the set of writes with label $St_o(x,v)$ or $U_o(x,v',v)$ in the target event structure $G_{tgt}$ for the respective accesses and whose po-suffix has some sequence of labels $\tau'$ such that $St_o(x,v) \cdot \tau' \notin P_{src}(i)$ or $U_o(x,v',v) \cdot \tau' \notin P_{src}(i)$ respectively. Then, because of the relationship between the two programs, we know that for each such $w \in W$, $St_o(x,v) \cdot Ld_o(x,\_) \cdot \tau' \in P_{src}(i)$ or $U_o(x,v',v) \cdot Ld_o(x,\_) \cdot \tau' \in P_{src}(i)$ respectively for the appropriate $\tau'$. Let $C$ be the immediate $G_{tgt}$-po-successors of the events in $W$.

Source Event Structure Construction.

To construct $G_{src}$, we follow the construction steps of $G_{tgt}$. For each target construction step that adds event $e$ to $G_{tgt}$ to get $G_{tgt}'$, we perform one or more corresponding steps going from $G_{src}$ to $G_{src}'$. We do a case analysis on the event $e$ of the target event structure.

Case $e \notin W$: In this case we append event $e$ to the source event structure as follows:

$$G_{src}' \cdot E = G_{src} \cdot E \uplus \{ e \}$$

$$G_{src}' \cdot po = (G_{src} \cdot po) \uplus \{ (a, e) \mid a \notin W \land \text{imm}(G_{tgt}' \cdot po)(a,e) \}$$

$$G_{src}' \cdot jf = G_{src} \cdot jf \uplus \{ (a, e) \mid G_{tgt}' \cdot jf(a,e) \}$$

$$G_{src}' \cdot mo = G_{tgt} \cdot mo$$

$$G_{src}' \cdot ew = G_{tgt} \cdot ew$$

Now we check the consistency of $G_{src}'$ event structure. We already know that $G_{src}$ and $G_{tgt}'$ are consistent.

Following the definition of $G_{src}'$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (COH'), (NCFU), (NCFSC) constraints immediately hold and hence $G_{src}'$ is also consistent.
E.2. Read after Write (RAW) / Read after Update (RAU)

Case $e \in W$: In this case we first append event $e$ and then event $r$ with $r.lab = Ld_{e}(x, v)$ to $G_{src}$ as follows:

$$G'_{src}.E = G_{src}.E \cup \{r, e\}$$

where $r.lab = Ld_{e}(x, v)$

$$G'_{src}.po = (G_{src}.po \cup \{(e, r), (a, e) \mid \text{imm}(G'_{tgt}.po)(a, e)\})^+$$

$$G'_{src}.jf = G_{src}.jf \cup \{(e, r)\}$$

$G'_{src}.mo = G'_tg$.mo

$G'_{src}.ew = G'_{tg}.ew$

Now we check the consistency of $G'_{src}$.
We already know that $G_{src}$ and $G'_{tg}$ is consistent. Following the construction of $G'_{src}$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (NCFU) constraints immediately hold.

Now we show that $G'_{src}$ satisfies (COH'). The outgoing edges from $r$ are $G'_{src}.fr$. Hence for an outgoing edge $G'_{src}.fr(r, a)$, there is $G_{src}.mo(e, a)$ edge. If $G'_{src}.fr(r, a)$ results in a $G'_{src}.hb; G'_{src}.eco'$ cycle, then $G_{src}.hb; G_{src}.eco'$ cycle is already there in $G_{src}$. But we know that $G_{src}$ is consistent and hence $G_{src}.hb; G_{src}.eco'$ is not possible. Hence a contradiction and $G'_{src}.hb; G'_{src}.eco'$ is also not possible. Thus $G'_{src}$ preserves (COH').

We know $G_{src}$ preserves (NCFSC). Consider $G'_{src}$ violates (NCFSC). In that case $G'_{src}$ violates (NCFU) or (NCFSC) due to $r$ or $e$.

Let $G'_{src}.psc = G_{src}.pscb \cup G'_{src}.psc$. Following the construction if $G'_{src}.psc(r, a)$ then $G'_{src}.psc(e, a)$ holds and when $G'_{src}.psc(a, r)$ where $a \neq e$, then $G'_{src}.psc(a, e)$. However, following the construction of $G'_{src}$, in this case, $G'_{tg}$ also violates (NCFU) or (NCFSC) due to $e$. This is not possible as $G'_{tg}$ is consistent. Hence a contradiction and $G'_{src}$ preserves (NCFU) and (NCFSC).

As a result, $G'_{src}$ is consistent.

**Source Execution Construction.** Next, we construct an execution $X_t \in \text{ex}_{\text{WEAKESTMO}}(G_{tg})$.

If $W \subseteq (G_{tg} \setminus X_t.E)$, then we find the corresponding execution $X_s \in \text{ex}_{\text{WEAKESTMO}}(G_{src})$ such that $X_s$ contains no event from $W$. In that case $X_s$ also does not contain any event created for $Ld_{e}(x, v)$ access.

Else if an event $w \in W$ is in $X_t$, then we know that we can find a source execution $X_s$ which contains both $w$ and $r$. Thus $X_s$ is as follows.

Thus $X_s$ is as follows.

$$X_s.E = X_{t}.E \cup \{r \mid X_{t}.E \cap W \neq \emptyset\}$$

$$X_s.po = (X_{t}.po \cup \{(w, r), (w, c) \mid (w, c) \in \text{imm}(X_{t}.po) \cap (W \times C) \land r \in (G_{src}.E \setminus G_{tg}.E))\}^+$$

$$X_s.rf = X_{t}.rf \cup \{(w, r) \mid w \in X_{t}.E \cap W\}$$

$$X_s.mo = X_{t}.mo$$

**Source Execution Consistency.** Now we check the consistency of $X_s$.

We know that $X_t$ is consistent. The (Well-formed), (total-MO), (Coherence), (Atomicity) constraints hold as they hold for $X_t$. Considering the (SC) constraint we observe that if $o =
E. Proofs of Correctness of Eliminations

If \( o' = \text{SC}, \) then \( r' \) introduces a \([\text{SC}], X_s.po_x; [\text{SC}]\) edge. This edge can create an \( (X_s, \text{psc}_{\text{base}} \cup X_s, \text{psc}_{\text{F}}) \) cycle only when there is a \( (X_t, \text{psc}_{\text{base}} \cup X_t, \text{psc}_{\text{F}}) \) cycle. Since \( X_t \) is consistent there is no \( (X_t, \text{psc}_{\text{base}} \cup X_t, \text{psc}_{\text{F}}) \) cycle. Hence there is no \( (X_s, \text{psc}_{\text{base}} \cup X_s, \text{psc}_{\text{F}}) \) cycle and \( X_s \) satisfies (SC). As a result, \( X_s \) is consistent.

**Same Behavior.**

Now we check whether \( \text{Behavior}(X_s) = \text{Behavior}(X_t) \) holds.

For locations \( y \neq x \), \( \text{Behavior}|_y(X_s) = \text{Behavior}|_y(X_t) \) holds.

For \( x \) load \( r \) does not introduce any new \( \text{mo} \) edge and hence does not affect behavior of \( X_s \).

Hence \( \text{Behavior}(X_s) = \text{Behavior}(X_t) \) holds.

**Race Preservation.**

Moreover, if \( X_t \) is racy, then the new read \( r \) does not introduce any new \( (X_s, \text{sw}_{\text{C11}} \setminus X_s, \text{po}) \) edge in \( X_s \). Hence \( X_s \) is also racy. As a result, if the target execution has undefined behavior due to data race then the source execution also has undefined behavior due to data race.

E.3. Read after Read (RAR)

**Proof.** Recall the relationship between the two programs for the thread \( i \) affected by the transformation:

\[
P_{\text{tgt}}(i) \subseteq P_{\text{src}}(i) \cup \{ \tau \cdot \text{Ld}_o(x, v) \cdot \tau' | \tau \cdot \text{Ld}_o(x, v) \cdot \text{Ld}_o(x, \_ \_ \_ ) \cdot \tau' \in P_{\text{src}}(i) \land o' \subseteq o \}
\]

For all other threads \( j \neq i \), we have \( P_{\text{tgt}}(j) = P_{\text{src}}(j) \). Assume we have a target event structure, \( G_{\text{tgt}} \), and an execution, \( X_t \in \text{ex}_{\text{WEAKESTMO}}(G_{\text{tgt}}) \), extracted from it.

Let \( R \) be the set of loads with label \( \text{Ld}_o(x, v) \) in the target event structure \( G_{\text{tgt}} \) whose po-suffix has some sequence of labels \( \tau' \) such that \( \text{Ld}_o(x, v) \cdot \tau' \notin P_{\text{src}}(i) \). Then, because of the relationship between the two programs, we know that for each such \( r \in W \), for the appropriate \( \tau' \), \( \text{Ld}_o(x, v) \cdot \text{Ld}_o(x, \_ \_ \_ ) \cdot \tau' \in P_{\text{src}}(i) \) holds. Let \( C \) be the immediate \( G_{\text{tgt}.po} \)-successors of the events in \( R \).

**Source Event Structure Construction.**

To construct \( G_{\text{src}} \), we follow the construction steps of \( G_{\text{tgt}} \). For each target construction step that adds event \( e \) to \( G_{\text{tgt}} \) to get \( G'_{\text{tgt}} \), we perform one or more corresponding steps going from \( G_{\text{src}} \) to \( G'_{\text{src}} \). We do a case analysis on the event \( e \) of the target event structure.

**Case \( e \notin R \):** In this case we append event \( e \) to the source event structure as follows:

\[
G'_{\text{src}.E} = G_{\text{src}.E} \uplus \{ e \}
\]
\[
G'_{\text{src}.po} = (G_{\text{src}.po} \uplus \{(a, e) \mid a \notin R \land \text{imm}(G'_{\text{tgt}.po})(a, e)\})^+
\]
\[
G'_{\text{src}.jf} = G'_{\text{tgt}.jf}
\]
\[
G'_{\text{src}.mo} = G'_{\text{tgt}.mo}
\]
\[
G'_{\text{src}.ew} = G'_{\text{tgt}.ew}
\]
E.3. Read after Read (RAR)

Now we check the consistency of $G'_{\text{src}}$ event structure. We already know that $G_{\text{src}}$ and $G'_{\text{tgt}}$ are consistent.

Following the definition of $G'_{\text{src}}$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (COH''), (NCFU), (NCFSC) constraints immediately hold and hence $G'_{\text{src}}$ is also consistent.

**Case $e \in R$:** In this case we first append event $e$ and then event $r$ with $r.\text{lab} = \text{Ld}_{e}(x, v)$ to $G_{\text{src}}$ as follows:

\[
G'_{\text{src}}.E = G_{\text{src}}.E \uplus \{d, e\} \quad \text{where } d.\text{lab} = \text{Ld}_{e}(x, v)
\]

\[
G'_{\text{src}}.po = (G_{\text{src}}.po \uplus \{(e, d), (a, e) \mid \text{imm}(G'_{\text{tgt}}.po)(a, e)\})^+
\]

\[
G'_{\text{src}}.jf = G_{\text{src}}.jf \uplus \{(a, e), (a, d) \mid G'_{\text{tgt}}.jf(a, e)\}
\]

\[
G'_{\text{src}}.mo = G'_{\text{tgt}}.mo
\]

\[
G'_{\text{src}}.ew = G'_{\text{tgt}}.ew
\]

Now we check the consistency of $G'_{\text{src}}$.

We already know that $G_{\text{src}}$ and $G'_{\text{tgt}}$ is consistent. Following the construction of $G'_{\text{src}}$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ) constraints immediately hold.

We now show that $G'_{\text{src}}$ satisfies (COH''). The outgoing edges from $d$ are $G'_{\text{src}}.\text{fr}$. Hence for an outgoing edge $G'_{\text{src}}.\text{fr}(d, a)$ there is $G'_{\text{src}}.\text{fr}(e, a)$ as well as $G'_{\text{tgt}}.\text{fr}(e, a)$ edges. Hence if $G'_{\text{src}}.\text{fr}(d, a)$ results in a $G'_{\text{src}}.\text{hb} \cup G'_{\text{src}}.\text{eco}'$ cycle, then there is also $G'_{\text{tgt}}.\text{hb}; G'_{\text{tgt}}.\text{eco}'$ cycle. But we know that $G'_{\text{tgt}}$ is consistent and hence $G'_{\text{tgt}}.\text{hb}; G'_{\text{tgt}}.\text{eco}'$ cycle is not possible. Hence a contradiction and $G'_{\text{src}}.\text{hb}; G'_{\text{src}}.\text{eco}'$ cycle is also not possible. Thus $G'_{\text{src}}$ preserves (COH'').

We know $G_{\text{src}}$ preserves (NCFSC). Consider $G'_{\text{src}}$ violates (NCFSC). In that case $G'_{\text{src}}$ violates (NCFU) or (NCFSC) due to $r$ or $e$.

Let $G'_{\text{src}}.\text{psc} = G'_{\text{src}}.\text{pscb} \cup G'_{\text{src}}.\text{pscf}$. Following the construction if $G'_{\text{src}}.\text{psc}(d, e'')$ then we know $G'_{\text{src}}.\text{psc}(d, e')$ holds and when $G'_{\text{src}}.\text{psc}(e'', d)$ where $e'' \neq e$, then $G'_{\text{src}}.\text{psc}(e'', e)$.

However, following the construction of $G'_{\text{src}}$, in this case, $G'_{\text{tgt}}$ also violates (NCFU) or (NCFSC) due to $e$. This is not possible as $G'_{\text{tgt}}$ is consistent. Hence a contradiction and $G'_{\text{src}}$ preserves (NCFU) and (NCFSC).

As a result, $G'_{\text{src}}$ is consistent.

**Source Execution Construction.** Next, we construct an execution $X_{t} \in \text{ex}_{\text{WEAKSTEMO}}(G_{\text{tgt}})$.

If $R \subseteq (G_{\text{tgt}} \setminus X_{t}.E)$, then we find the corresponding execution $X_{s} \in \text{ex}_{\text{WEAKSTEMO}}(G_{\text{src}})$ such that $X_{s}$ contains no $\text{Ld}_{e}(x, v)$. In that case $X_{s}$ also does not contain any event created for $\text{Ld}_{e}(x, v)$.

Else if an event $r \in R$ is in $X_{t}$, then we know that we can find a source execution $X_{s}$ which contains both $r$ and $d$. Thus $X_{s}$ is as follows.

Thus $X_{s}$ is as follows.

\[
X_{s}.E = X_{t}.E \uplus \{d \mid X_{t}.E \cap R \neq \emptyset\}
\]

\[
X_{s}.po = (X_{t}.po \uplus \{(r, d), (d, c) \mid (r, c) \in \text{imm}(X_{t}.po) \cap (R \times C) \cap d \in (G_{\text{src}}.E \setminus G_{\text{tgt}}.E)\})^+
\]

\[
X_{s}.rf = X_{t}.rf \uplus \{(a, d) \mid a \in \text{dom}(X_{t}.rf; [R])\}
\]

\[
X_{s}.mo = X_{t}.mo
\]
E. Proofs of Correctness of Eliminations

Source Execution Consistency. Now we check the consistency of $X_s$.
We know that $X_t$ is consistent. The (Well-formed), (total-MO), (Coherence), (Atomicity)
constraints hold as they hold for $X_t$. Considering the (SC) constraint we observe that if $o =
\tau' = sc$, then $\tau'$ introduces a $[SC], X_s, po_o; [SC]$ edge. This edge can create a $(X_s, psc_{base} \cup \nolimits X_s, psc_F)$ cycle only when there is a $(X_t, psc_{base} \cup X_t, psc_F)$ cycle. Since $X_t$ is consistent there is no $(X_t, psc_{base} \cup X_t, psc_F)$ cycle. Hence there is no $(X_s, psc_{base} \cup X_s, psc_F)$ cycle and $X_s$ satisfies (SC). As a result, $X_s$ is consistent.

Same Behavior.
Now we check whether $\text{Behavior}(X_s) = \text{Behavior}(X_t)$ holds.
For locations $y \neq x$, $\text{Behavior}|_y (X_s) = \text{Behavior}|_y (X_t)$ holds.
For $x$, load $d$ does not introduce any new mo edge and hence does not affect behavior of $X_s$.
Hence $\text{Behavior}(X_s) = \text{Behavior}(X_t)$ holds.

Race Preservation.
Moreover, if $X_t$ is racy, then the new read $d$ does not introduce any new $(X_s, \text{hb}_{C11} \setminus X_s, po)$ relation in $X_s$. Hence $X_s$ is also racy. As a result, if the target execution has undefined behavior due to data race then the source execution also has undefined behavior due to data race. □

E.4. Non-Atomic Read-Write (naRW)

Proof. Recall the relationship between the two programs for the thread $i$ affected by the transformation:

$$\mathbb{P}_{tgt}(i) \subseteq \mathbb{P}_{src}(i) \cup \{ \tau \cdot \tau' \mid \tau \cdot \text{Ld}_{NA}(x, v) \cdot \text{St}_{NA}(x, v) \cdot \tau' \in \mathbb{P}_{src}(i) \}$$

For all other threads $j \neq i$, we have $\mathbb{P}_{tgt}(j) = \mathbb{P}_{src}(j)$. Assume we have a target event structure, $G_{tgt}$, and an execution, $X_t \in \text{ex}_{\text{WEAKESTMO}}(G_{tgt})$, extracted from it.

Let $C$ be the set of events the target event structure $G_{tgt}$ whose po-suffix has some sequence of labels $\tau'$ such that $c \cdot \tau' \notin \mathbb{P}_{src}(i)$ where $c \in C$. Also let $D$ be the set of events which are immediate po-successors of events in $C$. Then, because of the relationship between the two programs, we know that for each such $c \in C$ and $c \in \tau$, $c \cdot \text{Ld}_{NA}(x, v) \cdot \text{St}_{NA}(x, v) \cdot \tau' \in \mathbb{P}_{src}(i)$ for the appropriate $\tau'$.

Source Event Structure Construction.
To construct $G_{src}$, we follow the construction steps of $G_{tgt}$. For each target construction step that adds event $e$ to $G_{tgt}$ to get $G'_{tgt}$, we perform one or more corresponding steps going from $G_{src}$ to $G'_{src}$. We do a case analysis on the event $e$ of the target event structure.

Case $e \in C$: In this case we append event $e$ followed by $\text{Ld}_{NA}(x, s.wval)$ justified from a write
E.4. Non-Atomic Read-Write (naRW)

sand StNA\((x, s, wval)\) to the source event structure as follows:

\(G'_{src} E = G_{src} \cup \{(e, r, w)\} \quad \text{where} \quad r.\text{lab} = Ld_{\text{NA}}(x, _) \quad \text{and} \quad w = St_{\text{NA}}(x, _)\)

\(G'_{src}.po = (G_{src}.po \cup \{(a, e), (e, r), (r, w) \mid G_{tgt}.po(a, e)\})^+\)

\(G'_{src}.jf = G_{src}.jf \cup \{(a, e), (s, r) \mid G_{tgt}.jf(a, e) \land \exists \text{w}(G'_{src}, s, r)\}\)

\(G'_{src}.mo = G_{src}.mo \cup \{(a, w) \mid a \in (G_{src}.V_{E} \setminus \text{WA}) \cup \{(w, a) \mid a \in \text{WA} \}\}

\quad \text{where} \quad \text{WA} = \{a \mid (G_{tgt}.ew^2; G_{tgt}.mo)(s, a)\}\)

\(G'_{src}.ew = G_{src}.ew \cup \{(a, e) \mid G_{tgt}.ew(a, e)\}\)

Now we check the consistency of \(G'_{src}\).

We already know that \(G_{src}\) and \(G'_{tgt}\) is consistent. Following the construction of \(G'_{src}\) and considering the definition of Remark 3, the (CF), (CFJ), (NCFJ), (ICF), (ICFJ), (NCFU) constraints immediately hold. It remains to show that \(G'_{src}\) satisfies (COH') and (NCFSC).

Again following the Remark 3 definition, additional events \(r\) and \(w\) do not create any \(G'_{src}.\text{hb}; G'_{src}.\text{eco}^2\) cycle. Moreover, \(r\) and \(w\) do not create any new \(G'_{src}.\text{pscb} \cup G'_{src}.\text{pscf}\) cycle. Hence \(G'_{src}\) satisfies (COH') and (NCFSC). As a result, \(G'_{src}\) is consistent.

Case \(e \not\in C\): In this case we append event \(e\) to the source event structure. However, if \(e\) is justified-from \(s\) in \(G'_{tgt}\) and happens-after the newly newly appended non-atomic store from \((G_{src}.E \setminus G_{tgt}.E)\) in \(G'_{src}\), then \(e\) is justified-from the new store \(St_{\text{NA}}(X, s, wval)\). Let \(W \subseteq (G_{src}.E \setminus G_{tgt}.E)\) be the set of such store events. Note that id event \(e\) happens-after event \(w\), then there exists an intermediate event \(d \in D\). Thus we construct \(G'_{src}\) as follows:

\(G'_{src}.E = G_{src}.E \cup \{e\}\)

\(G'_{src}.po = (G_{src}.po \cup \{(a, e) \mid G'_{tgt}.po(a, e)\})\)

\( \quad \cup \{(w, e) \mid w \in W \land e \in \text{codom}(\{C]; \text{imm}(G'_{ tgt}.po); [D])\}^+\)

\(G'_{src}.jf = G_{src}.jf \cup \{(a, e) \mid G'_{ tgt}.jf(a, e) \land e \not\in \text{codom}(\{D]; G_{src}.\text{hb})\}\)

\( \quad \cup \{(a, e) \mid G'_{ tgt}.jf(a, e) \land e \in \text{codom}(\{D]; G_{src}.\text{hb})\}\)

\(G'_{src}.mo = G_{src}.mo \cup \{(a, e) \mid G'_{ tgt}.mo(a, e)\} \cup \{(e, a) \mid G'_{tgt}.mo(e, a)\}\)

\(G'_{src}.ew = G_{src}.ew \cup \{(a, e) \mid G'_{ tgt}.ew(a, e)\}\)

Now we check the consistency of \(G'_{src}\).

We already know that \(G_{src}\) and \(G'_{tgt}\) is consistent. Following the construction of \(G'_{src}\), the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (NCFU) constraints immediately hold. It remains to show that \(G'_{src}\) satisfies (COH') and (NCFSC).

Assume there is a \(G'_{src}.\text{hb}; G'_{src}.\text{eco}^2\) cycle. We know there is no \(G_{src}.\text{hb}; G_{src}.\text{eco}^2\) cycle. Hence the cycle involves event \(e\). However, if event \(e\) introduces a \(G'_{src}.\text{hb}; G'_{src}.\text{eco}^2\), then from the definition, there is a \(G'_{tgt}.\text{hb}; G'_{tgt}.\text{eco}^2\) cycle which is a contradiction. Hence \(G'_{src}\) satisfies (COH'). Similarly if event \(e\) creates any new \(G'_{src}.\text{pscb} \cup G'_{src}.\text{pscf}\) cycle then there is already a \(G'_{tgt}.\text{pscb} \cup G'_{tgt}.\text{pscf}\) cycle which is a contradiction. Hence \(G'_{src}\) satisfies (NCFSC).

As a result, \(G'_{src}\) is consistent.
Source Execution Construction. Next, we construct an execution $X_t \in \text{ex}_{\text{WEAKESTMO}}(G_{\text{tgt}})$. If $X_t.E$ does not contain any event in $C'$ then we find the corresponding execution $X_s$ such that $X_s \in \text{ex}_{\text{WEAKESTMO}}(G_{\text{src}})$ and $X_s.E$ contains no corresponding $\text{St}_{\text{NA}}(x, v)$ and $\text{Ld}_{\text{NA}}(x, v)$ events.

Else if an event $c \in C'$ is in $X_t$, then we know that we can find an execution with $r, w \in X_s.E$ where $r.\text{lab} = \text{Ld}_{\text{NA}}(x, \_)$ and $w.\text{lab} = \text{St}_{\text{NA}}(x, \_)$. Thus $X_s$ is as follows.

$$X_s.E = X_t \cup \{r, w \mid X_t.E \cap C' \neq \emptyset\}$$

$$X_s.po = (X_t.po \\cup \{(c, r), (r, w), (w, d) \mid (c, d) \in \\text{imm}(X_t.po) \cap (C \times D) \land r \in (G_{\text{src}}.E \setminus G_{\text{tgt}}.E)\})^+$$

$$X_s.rf = X_t.rf \cup \{(s, r) \mid r \in (G_{\text{src}}.E \setminus G_{\text{tgt}}.E) \cap \text{codom}([C]; \text{imm}(G_{\text{src}}.po)) \land G_{\text{src}}.rf(s, r)\}$$

$$X_s.mo = X_t.mo \cup \{(a, w) \mid (a, w) \in (G_{\text{src}}.mo \cup G_{\text{src}}.mo^{-1}) \cap (X_t.E \times W)\}$$

Now we check the consistency of $X_s$.

We already know that $X_t$ is consistent. We also know either $X_s = X_t$ or $X_s$ has newly introduced $r, w$ events. In that case, following the definition of $X_s$, the (Well-formed), (total-MO), (Coherence), (Atomicity) constraints also hold for $X_s$ and hence $X_s$ is consistent.

Same Behavior.

Now we check whether $\text{Behavior}(X_s) = \text{Behavior}(X_t)$ holds. We consider the case where $w$ is not in $X_s$.

- In this case either $s$ or $s'$ is in $X_s$ where $G_{\text{src}}.ew(s, s')$. In this case $s.\text{wval} = s'.\text{wval} = v$. If $s$ or $s'$ is $X_t.mo$ maximal on $x$ then $(x, v) \in \text{Behavior}(X_t)$.

As a result, $\text{Behavior}_{|_x}(X_s) = \text{Behavior}_{|_x}(X_t)$ holds in both cases. For locations $y \neq x$, $\text{Behavior}_{|_y}(X_s) = \text{Behavior}_{|_y}(X_t)$ holds. As a result, $\text{Behavior}(X_s) = \text{Behavior}(X_t)$ holds.

Race Preservation. Moreover, if $X_t$ is racy, then the new write $d$ does not introduce any $X_{s,\text{SWC11}}$ edge in $X_t$. Hence $X_s$ is also racy. As a result, if the target execution has undefined behavior due to a data race, so does the source execution.

### E.5. Non-Adjacent Access Elimination (NA-OW)

A trace $\tau$ satisfies the intermediate condition for a location, $x$, which is written as $\text{GoodInterm}_x(\tau)$, if:

- it contains no $x$-accesses, i.e., $\tau \neq \tau_1.\text{RW}_x.\tau_2$ for all $\tau_1$ and $\tau_2$; and

110
E.5. Non-Adjacent Access Elimination (NA-OW)

\[
\begin{align*}
X &= 2; \\
X_{NA} &= 1; \\
Y_{REL} &= 1; \\
t &= Z_{RLX}; \\
X_{NA} &= 3;
\end{align*}
\]

(a) (NA-OW)

\[
\begin{align*}
[X = Y = Z = 0] \\
St(X, 2) &\quad \quad Ld_{ACQ}(Y, 1) \\
St_{NA}(X, 1) &\quad \quad Ld(X, 2) \\
St_{REL}(Y, 1) &\quad \quad St(Z, 1) \\
Ld(Z, 1) &\quad \quad St_{NA}(X, 3)
\end{align*}
\]

(b) Execution

\[
\begin{align*}
[X = Y = Z = 0] \\
St(X, 2) &\quad \quad Ld_{ACQ}(Y, 1) \\
St_{NA}(X, 1) &\quad \quad Ld(X, 2) \\
St_{REL}(Y, 1) &\quad \quad St(Z, 1) \\
Ld(Z, 1) &\quad \quad St_{NA}(X, 3) \\
StNA(X, 3) &\quad \quad Ld_{ACQ}(Y, 1) \\
St(X, 1) &\quad \quad Ld(X, u) \quad \quad \text{ // 2} \\
St_{REL}(Y, 1) &\quad \quad St(Z, 1) \\
Ld(Z, 0) &\quad \quad Ld(Z, 1) \\
St(X, 3) &\quad \quad St(X, 3)
\end{align*}
\]

(c) WEAKESTMO-LLVM target event structure

(d) WEAKESTMO-LLVM source event structure

Figure E.1.: NA-OW example executions and WEAKESTMO-LLVM event structures.

- it contains no rel-acq pairs, i.e., \( \tau \neq \tau_1 \cdot [\text{Rel}] \cdot \tau_2 \cdot [\text{Acq}] \cdot \tau_3 \) for all traces \( \tau_1, \tau_2, \) and \( \tau_3. \)

Let \( \mathcal{E}_\tau \) be the events corresponding to \( \tau \). If \( \mathcal{E}_\tau \) has no release access then \( St_{NA}(x, v') \) could reorder with \( \mathcal{E}_\tau \) and placed in adjacency with \( St_{NA}(x, v) \). Then \( St_{NA}(x, v') \) could be deleted by overwritten write (OW) transformation. But if \( \mathcal{E}_\tau \) contains a release operation then \( St_{NA}(x, v') \) cannot be reordered with \( \mathcal{E}_\tau \). Hence in this proof we consider the cases where \( C \) contains release access. Before going to the proof we discuss a special case for WEAKESTMO-LLVM model.

**Special Case** Given the program in consider the transformation deletes the \( X_{NA} = 1 \) access and hence results in an taget execution as shown in . This execution has a defined behavior according to the WEAKESTMO-LLVM model as there is no write-write race in this execution.

The execution can be extracted from the target event structure in Figure E.1c.

Given this target event structure we cannot contruct the source event structure as once we introduce \( St_{NA}(X, 1) \), we cannot create \( Ld(X, 2) \) directly.
E. Proofs of Correctness of Eliminations

However, note that, $Ld(X, 2)$ is in read-write race with $St_{NA}(X, 3)$. Hence the program has undefined behavior in WEAKESTMO-C11 and in WEAKESTMO-LLVM the respective event may return $u$ which can be evaluated to 2.

However, if $St_{NA}(X, 3)$ is appended after $Ld(X, 2)$, then we cannot create $Ld(X, u)$ in the source event structure directly. Hence $G_{src}$ requires to create a $St_{NA}(X, _)$ before $Ld(X, u)$ as shown in .

Proof. Let $W$ be the set of stores of thread $i$ of $G_{tgt}$ with label $St_o(x, v)$, and whose po-prefix has some sequence of labels $\tau$ such that $St_{NA}(x, v') \cdot \tau \cdot St_{NA}(x, v) \notin P_{src}(i)$. Then, because of the relationship between the two programs, we know that for each such $w \in W$, $St_{NA}(x, v') \cdot \tau \cdot St_{NA}(x, v) \in P_{src}(i)$ for the appropriate $\tau$.

Let $C$ be the set of first event in the sequence $\tau$.

Let $B$ be the set of immediate $G_{tgt}.po$-predecessor of $C$.

Let $F = G_{tgt}.Rel_{ix}$ be the set of first event in the sequence $\tau$.

Let $W$ be the set of the respective $St_o(x, v)$ labelled events and $W \subseteq codom([F]; G_{tgt}.po)$.

Let $R$ be the set of reads such that $R \subseteq \{ codom([B]; G_{tgt}.po; [F]G_{tgt.swe}; G_{tgt.hb}) \cap G_{tgt.Rx} \}$ and $M : R \mapsto G_{src}.E$ maps a read in $R$ to the corresponding read in source event structure.

Let $P$ be the

$\tau_x$ be the sub-sequence from $f \in F$ to $w \in W$ such that $G_{tgt}.po(f, w)$ holds and there is no $f' \in F$ such that $G_{tgt}.po(f', f)$.

PC($\tau_x$) be the $G_{src}.po$-maximal event appended to the source event structure.

$EW(\tau_x)$ be the set of writes on $x$ with label $St_{NA}(x, v)$ in $G_{src}$. The writes in $EW(\tau_x)$ are equal writes, that is, $\forall w_1, w_2 \in EW(\tau_x).G_{src}.ew(w_1, w_2)$ holds.

$D$ be the set of events deleted from source event structure.

$S$ be the events of $\tau_x$ that is, $S \subseteq codom([F], G_{tgt}.po) \cup dom(G_{tgt}.po; [W])$.

Source Event Structure Construction. To construct $G_{src}$, we follow the construction steps of $G_{tgt}$. For each target construction step that adds event $e$ to $G_{tgt}$ to get $G'_{tgt}$, we perform one or more corresponding steps going from $G_{src}$ to $G'_{src}$. We do a case analysis on the event $e$ of the target event structure.

Case $e \in C$:

We append a $St_{NA}(x, v')$ event $d$ followed by event $e$ as follows. The immediate $G_{tgt}.po$ predecessor of $e$ is $b$.

Let $s$ be the maximal-visible write on $x$ w.r.t $b$, that is, existsW($G_{src}, s, b$) hold. We refer to the event $s$ to create the mo relations to/from $d$.

$$G'_{src} = G_{src} \cup \{ d, e \} \quad \text{where } d.lab = St_{NA}(x, v')$$

$$G'_{src}.po = (G_{src}.po \cup \{ (d, e) \} \cup \{ (b, d) | (b, e) \in G_{tgt}.po \}$$

$$G'_{src}.if = G_{tgt}.if$$

$$G'_{src}.mo = G_{src}.mo \cup \{ (s, d) \} \cup \{ (p, d) | G_{src}.mo(p, s) \} \cup \{ (d, p) | G_{src}.mo(s, p) \}$$

where existsW($G_{src}, s, b$).

$$G'_{src}.ew = G_{src}.ew \cup \{ (a, e) | G_{tgt}.ew(a, e) \}$$
Also we update $D$ to $D \uplus \{d\}$. Now we check the consistency of $G'_{src}$. We already know that $G'_{src}$ and $G'_{tgt}$ is consistent. Following the construction of $G'_{src}$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (NCFU), (NCFSC) constraints immediately hold. It remains to show that $G'_{src}$ satisfies (COH').

From the definition, there is no $G'_{src}.hb;G'_{src}.eco$ as well as $G'_{tgt}.hb;G'_{tgt}.eco$ cycle. Compared to $G_{src}$ and $G'_{tgt}$, the additional $G'_{src}.mo$ edges are from and to the event $d$. Assume the mo edges to or from $d$ creates a $G'_{src}.hb;G'_{src}.eco$ cycle. However, for each $G'_{src}.mo(a,d)$ or $G'_{src}.mo(a,d)$ already there exists $G'_{src}.mo(s,a)$ or $G'_{src}.mo(a,s)$ respectively. Thus event $d$ as well as $e$ results no new $G'_{src}.hb;G'_{src}.eco$ cycle and hence $G'_{src}$ satisfies (COH').

As a result, $G'_{src}$ is consistent.

**Case** $e \in S$: Let $e$ is in sequence $\tau_x$. Two possibilities:

**Subcase** There exists an event $e_s$ such that $imm(G_{src}.po)(pc(\tau_x), e_s): pc' = pc[\tau_x \mapsto e_s]$. In this case $G_{src}' = G_{src}$ and hence $G_{src}'$ is consistent.

**Subcase Otherwise**: We take two steps where we first create an intermediate event structure $G''$ by appending $e$. Next, we append a sequence of events $Q$ where a read $r_e$ reads from a maximal visible write $w_e$ in $G_{src}$, that is, existsW($G_{src}, w_e, r_e$) until we append an event $w_e = St_{NA}(x, v)$. Moreover, $pc' = pc[\tau_x \mapsto e]$.

Next, we append a sequence of events $Q$ where a read $r_e$ reads from a maximal visible write $w_e$ in $G_{src}$, that is, existsW($G_{src}, w_e, r_e$) until we append an event $w_e = St_{NA}(x, v)$.

Thus $G''_{src}$ is as follows:

$G''_{src}.E = G_{src}.E \uplus \{e\} \cup \{Q\}$

$G''_{src}.po = (G_{src}.po \uplus \{(a, e) | G''_{tgt}.po(a, e)\} \uplus \{(p, q) | (p = e \lor p \in Q) \land q \in Q \land p \neq Q\}^+$

for all $q \in Q$

$G''_{src}.jf = G''_{tgt}.jf \uplus \{(a, e) | G''_{src}.jf(a, e)\}$

$\uplus \{(w_v, r_c) | r_c \in Q \land r_c \in R \land \text{existsW}(G_{src}, w_v, r_c)\}$

for all $r_c \in Q$

$G''_{src}.mo = G_{src}.mo \uplus \{(a, e) | G''_{src}.mo(a, e)\}$

$\uplus \{(a, q) | q \in \mathcal{W} \land a.loc = q.loc \land \neg G_{src}.cf(a, q) \land(a \in G_{src}.E \lor G''_{src}.po(a, q))\}$

$G''_{src}.ew = G_{src}.ew \uplus \{(a, e) | G''_{src}.ew(a, e)\} \uplus \{(w', w_e) | w' \in EW(\tau_x)\}$

and finally we update $EW(\tau_x)$, that is, $EW(\tau_x) = EW(\tau_x) \uplus \{w_e\}$.

Now we check the consistency of $G''_{src}$. We already know that $G_{src}$ and $G''_{tgt}$ is consistent. Following the construction of $G''_{src}$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (NCFU), (NCFSC) constraints immediately hold. It remains to show that $G''_{src}$ satisfies (COH').

From the definition, there is no $G''_{src}.hb;G''_{src}.eco$ as well as $G''_{tgt}.hb;G''_{tgt}.eco$ cycle. Compared to $G_{src}$ and $G''_{tgt}$, the additional $G''_{src}.hb$ and $G''_{src}.eco$ edges are from and to the event...
E. Proofs of Correctness of Eliminations

\{e\} \cup Q. The edge from/to e does not create new \(G'_{\text{src}.hb}; G'_{\text{src}.\text{eco}}\) cycle as there is no \(G'_{\text{tgt}.hb}; G'_{\text{tgt}.\text{eco}}\) cycle. Also the outgoing \(G'_{\text{src}.hb}\) and \(G_{\text{src}.\text{eco}}\) edges from events in \(Q\) are only to other events in \(Q\). In consequence, there is no \(G'_{\text{tgt}.hb}; G'_{\text{tgt}.\text{eco}}\) cycle to/from \(Q\) events. Thus \(G'_{\text{src}}\) satisfies (COH') and \(G'_{\text{src}}\) is consistent.

**Case** \(e \in R:\)

In this case event \(e\) reads from a visible write \(w_1\) which is now overwritten. \(w_1\) has a \(G'_{\text{tgt}.po}\)-successor sequence \(\tau\) which includes \(f \in F\) such that \(G'_{\text{tgt}.po}(w_1, f)\). From the construction, \(f\) has a \(G_{\text{src}.po}\) event \(w_c\) such that \(w_c.lab = \text{St}_{\text{NA}}(x, v)\). Consider we append event \(r\) in source event structure corresponding to \(e\).

Following the \textsc{weakestmo-c11} model, if we append an event corresponding to \(e\) it results in race and hence the source has undefined behavior. Hence the transformation is correct.

Now we consider the \textsc{weakestmo-llvm} model. If \(r \in U\), then there is a write-write race and in that case the source program has undefined behavior. Hence the transformation is correct.

The according to \textsc{weakestmo-llvm} read-write race has define behavior. Hence we continue the event structure construction when \(r\) is a load, that is, \(r \in \text{Ld}\).

We append \(r\) to the \(G_{\text{src}}\) as follows:

\[
\begin{align*}
G'_{\text{src}.E} &= G_{\text{src}.E} \uplus \{r\} \quad \text{where } r.lab = \text{Ld}(x, u)\text{ which we evaluate } u \text{ to } w_1.wval. \\
G'_{\text{src}.po} &= (G_{\text{src}.po} \uplus \{(a, r) \mid G'_{\text{tgt}.po}(a, e)\})^+ \\
G'_{\text{src}.if} &= G'_{\text{tgt}.if} \uplus \{(w_c, r)\} \\
G'_{\text{src}.mo} &= G_{\text{src}.mo} \\
G'_{\text{src}.ew} &= G_{\text{src}.ew}
\end{align*}
\]

Also we update the mapping \(M' = M[e \mapsto r]\).

Now we check the consistency of \(G'_{\text{src}}\). We already know that \(G_{\text{src}}\) and \(G'_{\text{tgt}}\) is consistent. Following the construction of \(G'_{\text{src}}\), the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (NCFU), (NCFSC) constraints immediately hold. It remains to show that \(G'_{\text{src}}\) satisfies (COH').

From the definition, there is no \(G'_{\text{src}.hb}; G'_{\text{src}.\text{eco}}\) cycle. So any new \(G'_{\text{src}.hb}; G'_{\text{src}.\text{eco}}\) cycle involves \(r\). The incoming edges to \(r\) is \(G'_{\text{src}.po}\), \(G'_{\text{src}}(w_c, r)\) and the outgoing edges are \(G'_{\text{src}.fr}\) edges when \(w_c \in G'_{\text{tgt}}\). As well. These edges cannot contitute a \(G'_{\text{src}.hb}; G'_{\text{src}.\text{eco}}\) cycle as there is no \(G'_{\text{tgt}.hb}; G'_{\text{tgt}.\text{eco}}\) cycle involving \(w_c\). As a result, \(G'_{\text{src}}\) preserves (COH') and \(G'_{\text{src}}\) is consistent.

**Case** \(e \in W:\)

Either there already exists a write event \(w_c \in EW(\tau_x)\) with \(w_c.lab = \text{St}_{\text{NA}}(x, v)\) such that \(\text{imm}(G_{\text{src}.po})(\text{pc}(\tau_x), w_c)\) or we append event \(e\).

**Subcase** \(\exists w_c \in EW(\tau_x) \text{ such that } w_c.lab = \text{St}_{\text{NA}}(x, v), \text{imm}(G_{\text{src}.po})(\text{pc}(\tau_x), w_c):\)
In this case $pc' = pc[\tau_x \mapsto w_c]$ and $G'_{src}$ is as follows:

$$
\begin{align*}
G'_{src}.E &= G_{src}.E \\
G'_{src}.po &= G_{src}.po \\
G'_{src}.jf &= G_{src}.jf \\
G'_{src}.mo &= G_{src}.mo \\
G'_{src}.ew &= G_{src}.ew \cup \{(a, w_c) | G'_{tgt}.ew(a, e)\}
\end{align*}
$$

Now we check the consistency of $G'_{src}$. We already know that $G_{src}$ and $G'_{tgt}$ is consistent. Following the construction of $G'_{src}$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (NCFU), (NCFSC) constraints immediately hold. It remains to show that $G'_{src}$ satisfies (COH').

From the definition, there is no $G_{src}.hb; G_{src}.eco^2$ cycle. So any new $G_{src}.hb; G'_{src}.eco^2$ cycle involves new outgoing $G'_{src}.rf$ from $w_c$. However, $G'_{tgt}$ also has corresponding outgoing $G'_{tgt}.rf$ edge from $e$ and there is no $G'_{tgt}.hb; G'_{tgt}.eco^2$ cycle involving $e$. Hence there is no $G_{src}.hb; G'_{src}.eco^2$ cycle involving $w_c$. As a result, $G'_{src}$ satisfies (COH') and $G'_{src}$ is consistent.

**Subcase Otherwise:** We append $e$ to $G_{src}$ and construct $G'_{src}$ as follows where $pc'(\tau_x) = e$.

$$
\begin{align*}
G'_{src}.E &= G_{src}.E \cup \{e\} \\
G'_{src}.po &= (G_{src}.po \cup \{(pc(\tau_x), e)\})^+ \\
G'_{src}.jf &= G'_{tgt}.jf \\
G'_{src}.mo &= G_{src}.mo \cup \{(a, e) | G'_{tgt}.mo(a, e)\} \cup \{(e, a) | G'_{tgt}.po(e, a)\} \\
&\quad \cup \{(w, e) | w.lab = St_{NA}(x, v') \land w \in \text{dom}([B]; G_{src}.po) \\
&\quad \land \text{dom}(G_{src}.po); [C]) \land G_{src}.po(w, pc(\tau_x))\} \\
G'_{src}.ew &= G_{src}.ew \cup \{(a, e) | G'_{tgt}.ew(a, e)\}
\end{align*}
$$

Now we check the consistency of $G'_{src}$. We already know that $G_{src}$ and $G'_{tgt}$ is consistent. Following the construction of $G'_{src}$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (NCFU), (NCFSC) constraints immediately hold. It remains to show that $G'_{src}$ satisfies (COH').

From the definition, there is no $G_{src}.hb; G_{src}.eco^2$ cycle. So any new $G_{src}.hb; G'_{src}.eco^2$ cycle involves event $e$. However, if there is any outgoing $G'_{src}.mo$ edge from $e$ then there is a write-write race and hence the source program has undefined behavior. Hence there is no $G'_{src}.hb; G'_{src}.eco^2$ cycle involving $e$. As a result, $G'_{src}$ satisfies (COH') and $G'_{src}$ is consistent.

**Case** $e \in G'_{tgt}.E \setminus (C \cup S \cup R \cup W)$:
E. Proofs of Correctness of Eliminations

We construct the $G'_{\text{src}}$ as follows:

$$G'_{\text{src}}.E = G_{\text{src}}.E \cup \{e\}$$

$$G'_{\text{src}}.po = (G_{\text{src}}.po \cup \{(a, e) \mid G'_{\text{tgt}}.po(a, e)\})^+$$

$$G'_{\text{src}}.jf = G'_{\text{tgt}}.jf \cup \{(a, e) \mid G'_{\text{tgt}}.jf(a, e)\}$$

$$G'_{\text{src}}.mo = G_{\text{src}}.mo \cup \{(a, e) \mid G'_{\text{tgt}}.mo(a, e)\}$$

- $\forall \{(d, e) \mid d \in D \land G'_{\text{tgt}}.mo(s, e) \land \text{existsW}(G'_{\text{src}}, s, d)\}$
- $\forall \{(e, d) \mid d \in D \land G'_{\text{tgt}}.mo(e, s) \land \text{existsW}(G'_{\text{src}}, s, d)\}$
- $\forall \{(e, c) \mid c \in G'_{\text{src}}.E \setminus G'_{\text{tgt}}.E \land c.\text{loc} = e.\text{loc} \land G'_{\text{src}}.cf(e, c)\}$

$$G'_{\text{src}}.ew = G_{\text{src}}.ew \cup \{(a, e) \mid G'_{\text{tgt}}.ew(a, e)\}$$

Now we check the consistency of $G'_{\text{src}}$. We already know that $G_{\text{src}}$ and $G'_{\text{tgt}}$ is consistent. Following the construction of $G'_{\text{src}}$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (NCFU), (NCFSC) constraints immediately hold. It remains to show that $G'_{\text{src}}$ satisfies (COH').

From the definition, there is no $G_{\text{src}}.hb$; $G_{\text{src}}.eco$ cycle. So any new $G'_{\text{src}}.hb$; $G'_{\text{src}}.eco$ cycle involves event $d \in D$ or the events in $G'_{\text{src}}.E \setminus G'_{\text{tgt}}.E$. However, following the definition, if there is any new $G'_{\text{src}}.hb$; $G'_{\text{src}}.eco$ cycle involving event $d$ then there is a cycle involving write event $s$ where existsW($G'_{\text{src}}$, $s$, $d$). In that case there is also $G'_{\text{tgt}}.hb$; $G'_{\text{tgt}}.eco$ cycle which is a contradiction. The writes in $G'_{\text{src}}.E \setminus G'_{\text{tgt}}.E$ have no outgoing $G'_{\text{src}}.mo$ $G'_{\text{src}}.po$ edge and hence cannot create a $G'_{\text{src}}.hb$; $G'_{\text{src}}.eco$ cycle. The reads in $G'_{\text{src}}.E \setminus G'_{\text{tgt}}.E$ may have outgoing $G'_{\text{src}}.fr$ edges. However, if any such $G'_{\text{src}}.fr$ edge creates a cycle then following the definition, there is already a $G'_{\text{src}}.hb$; $G'_{\text{src}}.eco$ cycle which is a contradiction. Hence $G'_{\text{src}}$ satisfies (COH') and $G'_{\text{src}}$ is consistent.

Source Execution Construction. Next, we construct an execution $X_t \in \text{ex}_{\text{WEAKESTMO}}(G_{\text{tgt}})$.

If $W \subseteq (G_{\text{tgt}}.E \setminus X_t.E)$, then we find the corresponding execution $X_s \in \text{ex}_{\text{WEAKESTMO}}(G_{\text{src}})$ such that $X_s$ contains no event created for store$_{\text{w'}}(x, v')$. Else if an event $w \in W$ is in $X_t$, then we know that we can find an execution with $w \in X_s.E$ and $X_s.E$ also contains an event $d \in D$ where $d.\text{lab} = \text{St}_{\text{NA}}(x, v')$. Also let $r \in R \cap X_t.E$. Thus $X_s$ is as follows.

$$X_s.E = X_t.E \cup \{d \mid X_t.E \cap W \neq \emptyset\} \cup \{r \mid r \in R \cap X_t.E\} \cup \{M(r) \mid r \in R \cap X_t.E\}$$

$$X_s.po = (X_t.po \cup \{(b, d), (d, c) \mid (b, c) \in \text{imm}(X_t.po) \cap (B \times C) \land d \in (G_{\text{src}}.E \setminus G_{\text{tgt}}.E)\} \cup \{(p, r) \mid X_t.po(p, r) \land p \notin R \land r \in R \cap X_t.E\} \cup \{(p, M(r)) \mid X_t.po(p, r) \land p \notin R \land r \in R \cap X_t.E\})^+$$

$$X_s.rf = X_t.rf \cup \{(a, r) \mid r \in R\} \cup \{(w, M(r)) \mid G_{\text{src}}.rf(w, M(r)) \land r \in R \land w \in X_s.E\}$$

$$X_s.mo = X_t.mo \cup \{(d, w) \mid d \in D \land w \in \text{codom}([D]; G_{\text{src}.mo}) \cap X_s.E\} \cup \{(w, d) \mid d \in D \land w \in \text{dom}(G_{\text{src}.mo}; [D]) \cap X_s.E\}$$

Source Execution Consistency. Now we check the consistency of $X_s$.

- Following the definition of $X_s$ the (Well-formed) is satisfied.
• We know that $X_t$ follows (total-MO). The additional write $d$ introduced in $X_s$ has the label $S_{NA}(x, v')$. However, from the definition of $G_{src}$ and $X_s$, event $d$ preserves (total-MO).

• Assume (Atomicity) does not hold in $X_s$. We know that (Atomicity) holds in $X_t$. Hence (Atomicity) is violated due to event $d$. In that case there exists $u \in X_s.U_x$ such that $X_s.fr(u, d)$ and $X_s.mo(d, u)$. However, in this case there is a write-write race and hence the source program has undefined behavior which is a contradiction. Hence (Atomicity) holds in $X_s$.

• Now we check if (SC) holds. As $d \not\in SC$, it introduces no new $[SC]; X_s.\text{hb}_{C11}; [SC]$ path compared to $X_t$. We also know that $SC$ holds on $X_t$. As a result, $X_s$ also preserves $SC$.

Thus $X_s$ is consistent and $X \in ex_{WEAKESTMO}(G_{src})$ holds.

**Same Behavior.**

For locations $y \neq x$, we have $X_s.E_y = X_t.E_y$ and so $\text{Behavior}(X_s)|_y = \text{Behavior}(X_t)|_y$ trivially holds. Now we check whether $\text{Behavior}(X_s)|_x = \text{Behavior}(X_t)|_x$ holds. Note that any newly introduced event $d \in X_s.E \setminus X_t.E$ is not $X_s.mo$ maximal, because in that case there exists a store $S_{NA}(x, v)$ which is $X_s.mo$ after $d$. Hence $\text{Behavior}(X_s) = \text{Behavior}(X_t)$ holds.

**Race Preservation.** Moreover, if $X_t$ is racy, then the new write $d$ does not introduce any $X_s.sw_{C11}$ edge in $X_s$. Hence $X_s$ is also racy. As a result, if the target execution has undefined behavior due to a data race, so does the source execution.

---

**E.6. Non-adjacent Read after Write (NA-RAW)**

**Proof.** Recall the relationship between the two programs for the thread $i$ affected by the transformation:

$$P_{tgt}(i) \subseteq P_{src}(i) \cup \{\tau_1.\text{St}_{NA}(x, v) \cdot \tau \cdot \tau' \mid \tau_1.\text{St}_{NA}(x, v) \cdot \tau.\text{Ld}_{NA}(x, v) \cdot \tau' \in P_{src}(i)\}$$

A trace $\tau$ satisfies the intermediate condition for a location, $x$, which is written as $\text{GoodInterm}_x(\tau)$, if:

- it contains no $x$-accesses, i.e., $\tau \neq \tau_1.\text{RW}_x.\tau_2$ for all $\tau_1$ and $\tau_2$; and

- it contains no rel-acq pairs, i.e., $\tau \neq \tau_1.[\text{Rel}].\tau_2.[\text{Acq}].\tau_3$ for all traces $\tau_1$, $\tau_2$, and $\tau_3$.

Let $E_\tau$ be the events corresponding to $\tau$. If $E_\tau$ has no acquire access then $\text{Ld}_{NA}(x, v)$ could reorder with $E_\tau$ and placed in adjacency with $\text{St}_{NA}(x, v)$. Then $\text{St}_{NA}(x, v')$ could be deleted by read after write (RAW) transformation. But if $E_\tau$ contains an acquire operation then $\text{Ld}_{NA}(x, v)$ cannot be reordered with $E_\tau$.

For all other threads $j \neq i$, we have $P_{tgt}(j) = P_{src}(j)$.

Assume we have a target event structure, $G_{tgt}$, and an execution, $X_t \in ex_{WEAKESTMO}(G_{tgt})$, extracted from it.
E. Proofs of Correctness of Eliminations

Let $E_{\tau'}$ be the events corresponding to $\tau'$.
Let $C \in E_{\tau}$ and $D \in E_{\tau'}$ be the set of events such that $(C \times D) \subseteq \text{imm}(G_{\text{tgt}.po})$ holds.
Let $W$ be the corresponding events with label $\text{St}_{\text{NA}}(x, v)$.

Source Event Structure Construction.
To construct $G_{\text{src}}$, we follow the construction steps of $G_{\text{tgt}}$. For each target construction step that adds event $e$ to $G_{\text{tgt}}$ to get $G'_{\text{tgt}}$, we perform one or more corresponding steps going from $G_{\text{src}}$ to $G'_{\text{src}}$. We do a case analysis on the event $e$ of the target event structure.

Case $e \notin D$: In this case we append event $e$ to the source event structure as follows:

\[
\begin{align*}
G'_{\text{src},E} &= G_{\text{src},E} \cup \{e\} \\
G'_{\text{src}.po} &= (G_{\text{src}.po} \cup \{(a, e) | G'_{\text{tgt}.po(a, e)}\})^+ \\
G'_{\text{src}.jf} &= G_{\text{src}.jf} \cup \{(a, e) | G'_{\text{tgt}.jf(a, e)}\} \\
G'_{\text{src}.mo} &= G_{\text{tgt}.mo} \\
G'_{\text{src}.ew} &= G_{\text{tgt}.ew}
\end{align*}
\]

Now we check the consistency of $G'_{\text{src}}$ event structure. We already know that $G_{\text{src}}$ and $G'_{\text{tgt}}$ are consistent.

Following the definition of $G'_{\text{src}}$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (COH'), (NCFU), (NCFSC) constraints immediately hold and hence $G'_{\text{src}}$ is also consistent.

Case $e \in D$: In this case we first append event $r$ with $r.\text{lab} = \text{Ld}_{\text{NA}}(x, v)$ which is immediate $G'_{\text{src}.po}$-successor of $c \in C$ where $\text{imm}(G'_{\text{tgt}.po})(c, e)$ holds. Moreover, $r$ is justified-from a write $w \in W$ and $G'_{\text{tgt}.po}(w, e)$ holds. Then we append event $e$ which is immediate $G'_{\text{src}.po}$-successor of $r$.

Thus $G'_{\text{src}}$ as follows:

\[
\begin{align*}
G'_{\text{src},E} &= G_{\text{src},E} \cup \{r, e\} \text{ where } r.\text{lab} = \text{Ld}_{\text{NA}}(x, v) \\
G'_{\text{src}.po} &= (G_{\text{src}.po} \cup \{(c, r), (r, e) | \text{imm}(G'_{\text{tgt}.po})(c, e)\})^+ \\
G'_{\text{src}.jf} &= G_{\text{src}.jf} \cup \{(w, r) | w \in W\} \\
G'_{\text{src}.mo} &= G_{\text{tgt}.mo} \\
G'_{\text{src}.ew} &= G_{\text{tgt}.ew}
\end{align*}
\]

Now we check the consistency of $G'_{\text{src}}$.
We already know that $G_{\text{src}}$ and $G'_{\text{tgt}}$ is consistent. Following the construction of $G'_{\text{src}}$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (NCFU), (NCFSC) constraints immediately hold.

Now we show that $G'_{\text{src}}$ satisfies (COH'). The outgoing edges from $r$ are $G'_{\text{src}.fr}$. Hence for an outgoing edge $G'_{\text{src}.fr}(r, a)$, there is $G_{\text{src}.mo}(w, a)$ edge. If $G'_{\text{src}.fr}(r, a)$ results in a $G'_{\text{src}.hb}; G'_{\text{src}.eco}$ cycle, then $G_{\text{src}.hb}; G_{\text{src}.eco}$ cycle is already there in $G_{\text{src}}$. But we know that $G_{\text{src}}$ is consistent and hence $G_{\text{src}.hb}; G_{\text{src}.eco}$ is not possible. Hence a contradiction and $G'_{\text{src}.hb}; G'_{\text{src}.eco}$ is also not possible. Thus $G'_{\text{src}}$ preserves (COH').

As a result, $G'_{\text{src}}$ is consistent.
**Source Execution Construction.** Next, we construct an execution \( X_t \in \text{ex}^{\text{WEAKEST-MO}}(G_{tgt}) \).

If \( W \subseteq (G_{tgt} \setminus X_t.E) \), then we find the corresponding execution \( X_s \in \text{ex}^{\text{WEAKEST-MO}}(G_{src}) \) such that \( X_s \) contains no event from \( W \). In that case \( X_s \) also does not contain any event created for \( \text{Ld}_{\text{NA}}(x, v) \) access.

Else if an event \( w \in W \) is in \( X_t \), then we know that we can find a source execution \( X_s \) which contains both \( w \) and \( r \). Thus \( X_s \) is as follows.

Thus \( X_s \) is as follows.

\[
X_s.E = X_t.E \cup \{ r \mid X_t.E \cap W \neq \emptyset \}
\]

\[
X_s.po = (X_t.po \cup \{ (c, r), (r, d) \mid (c, d) \in \text{imm}(X_t.po) \cap (C \times D) \land r \in (G_{src}.E \setminus G_{tgt}.E) \})^+\]

\[
X_s.rf = X_t.rf \cup \{ (w, r) \mid w \in X_t.E \cap W \}
\]

\[
X_s.mo = X_t.mo
\]

**Source Execution Consistency.** Now we check the consistency of \( X_s \).

We know that \( X_t \) is consistent. The (Well-formed), (total-MO), (Coherence), (Atomicity), and (SC) constraints hold as they hold for \( X_t \). As a result, \( X_s \) is consistent.

**Same Behavior.**

Now we check whether \( \text{Behavior}(X_s) = \text{Behavior}(X_t) \) holds.

For locations \( y \neq x \), \( \text{Behavior}|_y(X_s) = \text{Behavior}|_y(X_t) \) holds.

For \( x \) load \( r \) does not introduce any new \( \text{mo} \) edge and hence does not affect behavior of \( X_s \). Hence \( \text{Behavior}(X_s) = \text{Behavior}(X_t) \) holds.

**Race Preservation.**

Moreover, if \( X_t \) is racy, then the new read \( r \) does not introduce any new \( (X_s.po \setminus X_s.po) \) edge in \( X_s \). Hence \( X_s \) is also racy. As a result, if the target execution has undefined behavior due to data race then the source execution also has undefined behavior due to data race.
F. Proof of Correctness of Speculative Load

Theorem 8. The transformation $\epsilon \leadsto Ld_o(x, \_)$ is correct in WEAKESTMO-LLVM.

Proof. Let $R \subseteq G_{tgt}.E$ be the set of introduced events with label $Ld_o(x, v)$ in the target event structure $G_{tgt}$ such that $Ld_o(x, v) \in \mathbb{P}_{src}(i)$. Then, because of the relationship between the two programs, we know that for each such $r \in R$, $\tau \cdot \tau' \notin \mathbb{P}_{src}(i)$ holds. Let $C$ be the immediate $G_{tgt}.po$ successors of $R$ events.

Source Event Structure Construction.
To construct $G_{src}$, we follow the construction steps of $G_{tgt}$. For each target construction step that adds event $e$ to $G_{tgt}$ to get $G_{tgt}'$, we perform one or more corresponding steps going from $G_{src}$ to $G_{src}'$. We do a case analysis on the event $e$ of the target event structure.

Case $e \in R$:
In this case $G_{src}' = G_{src}$ and $G_{src}'$ is consistent as $G_{src}$ is consistent.

Case $e \in C$:
In this case we append $e$ to the event in $C$ as follows:

$$
G_{src}' = G_{src}.E \cup \{e\}
$$

$$
G_{src}'.po = (G_{src}.po \cup \{(c, e) \mid (c, e) \in C; \text{imm}(G_{tgt}.po); [R]; \text{imm}(G_{tgt}'.po))
$$

$$
G_{src}'.jf = G_{src}.jf \cup \{(a, e) \mid G_{tgt}.jf(a, e)
$$

$$
G_{src}'.mo = G_{tgt}.mo
$$

$$
G_{src}'.ew = G_{tgt}.ew
$$

Now we check the consistency of $G_{src}'$. We already know that $G_{src}$ and $G_{tgt}'$ is consistent. Following the construction of $G_{src}'$, the (CF), (CFJ), (VISJ), (ICF), (ICFJ), (COH'), (NCFU), (NCFSC) constraints immediately hold.

Case $e \in G_{tgt}'.E \setminus (C \cup R)$:

Source Execution Construction. Next, we construct an execution $X_t \in \text{ex}_{WEAKESTMO}(G_{tgt})$. If $R \subseteq (G_{tgt} \setminus X_t.E)$, then we find the corresponding execution $X_s \in \text{ex}_{WEAKESTMO}(G_{src})$ such that $X_s$ contains no event created for $Ld_o(x, v)$. Else if an event $r \in R$ is in $X_t$, then we know
that we can find an execution with $r \notin X_s.E$. Thus $X_s$ is as follows.

$$X_s.E = X_s.E \setminus R$$

$$X_s.po = X_s.po \setminus \{ (a, b) \mid a \in R \lor b \in R \}$$

$$X_s.rf = X_s.rf \setminus \{ (a, b) \mid a \in R \lor b \in R \}$$

$$X_s.mo = X_s.mo$$

**Source Execution Consistency.** Now we check the consistency of $X_s$.

Since $X_t$ is consistent, the (Well-formed), (total-MO), (Coherence), (Atomicity), (SC) constraints also hold for $X_s$.

**Same Behavior.** The $R$ events are loads and hence do not affect program behavior. Hence, $\text{Behavior}(X_s) = \text{Behavior}(X_t)$ holds.

**Race Preservation.** The $R$ events may introduce new read-write races in the target execution compared to the source execution. This is not correct in WEAKESTMO-C11 model, but it is fine in the WEAKESTMO-LLVM model.