A OTHER LOOPING CONSTRUCTS

Using the primitive looping construct `loop` e, the loop exiting construct $\texttt{break}_n$ e, and a simple register library, we can implement other types of loops as follows:

```
repeat e ≜
  loop
    let x = e in
    if x = 0 then 0 else break₁ e
```

```
for i = e₁ to e₂ do e₃ ≜
  let ivar = alloc(1) in
  store(ivar, e₁);
  loop
    let i = load(ivar) in
    store(ivar, i + 1);
    if i ≤ e₂ then e₃ else break₁ 0
```

B PROOFS OF §5 THEOREMS

B.1 Correctness of Theorem 2

Given an execution G, let us write $\mathsf{isMX}(G)$ for the following:

- $G.\mathsf{lhb}$ is a strict total order on $G.E$;
- $G.\mathsf{lhb}|_{\mathrm{imm}} \subseteq (C^x \times \mathcal{L}^x) \cup (\mathcal{L}^x \times \mathcal{U}^x) \cup (\mathcal{U}^x \times \mathcal{L}^x)$; and
- $[\mathcal{L}^x]; (G.\mathsf{lhb} \setminus G.\mathsf{po}); [\mathcal{L}^x \cup \mathcal{U}^x] \subseteq G.\mathsf{po}; [\mathcal{U}^x]; G.\mathsf{lhb}$.

We are then required to show that for all x and G, if $G \in \mathcal{G}_{\mathrm{c}}^{\mathsf{MX}} \cap \mathcal{G}_{\mathrm{wf}}^{\mathsf{MX}}$, then $\mathsf{isMX}(G_x)$ holds.

Proof. Pick an arbitrary x and $G \in \mathcal{G}_{\mathrm{c}}^{\mathsf{MX}} \cap \mathcal{G}_{\mathrm{wf}}^{\mathsf{MX}}$ with $G_x = \langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}, \mathsf{lhb}\rangle$. Pick an event e_0 from E such that it is minimal in lhb; that is, $\forall e \in E \setminus \{e_0\}.\ (e, e_0) \neq \mathsf{lhb}$. We first demonstrate that $e_0 \in C^x$. As e_0 is minimal with respect to lhb, $\mathsf{po} \subseteq \mathsf{lhb}$ and since from the well-formedness of G_x we know that the first event (in po order) in each thread is either a constructor or lock event, we know that $e_0 \in C^x \cup \mathcal{L}^x$. Let us assume $e_0 \in \mathcal{L}^x$. From the consistency of G_x we know that there exists u such that $(u, e_0) \in \mathsf{com} \cap \mathsf{so} \subseteq \mathsf{lhb}$. This however contradicts our assumption that e_0 is minimal with respect to lhb and we thus know $e_0 \in C^x$.

Let $E = E_o^0 \cup E_r^0$ with $E_o^0 \triangleq \{e_0\}$ and $E_r^0 \triangleq E \setminus \{e_0\}$. Let us write $\mathtt{numL}(S)$ to denote the number of lock events in the set of events S; and write $\mathtt{numU}(S)$ to denote the number of unlock events in the set of events S. Note that from the well-formedness of G_x we know that $\mathtt{numU}(E_r^0) \leq \mathtt{numL}(E_r^0)$. Moreover, from the well-formedness of G_x (namely the uniqueness of the constructor event) we know that $E_r^0 \cap C^x = \emptyset$. Without loss of generality, let m denote the number of lock events in E_r^0; i.e. $\mathtt{numL}(E_r^0) = m$. We next demonstrate that:

$$
\begin{array}{l}
\forall n \in \mathbb{N}.\ \forall E_r, E_o. \\
\quad E{=}E_o \uplus E_r \wedge \mathtt{numL}(E_r){=}n \wedge \mathtt{numU}(E_r) \leq \mathtt{numL}(E_r) \wedge E_r \cap C{=}\emptyset \\
\quad \wedge [E_r]; \mathsf{lhb}; [E_o]{=}\emptyset \wedge \mathsf{isMX}(G_x|_{E_o}) \wedge G_x|_{E_r} \text{is MX-well-formed on } x \\
\quad \wedge \exists! e_m.\ e_m{=}\max(E_o, \mathsf{lhb}) \wedge e_m \in C^x \cup \mathcal{U}^x \wedge \forall e \in E_o \setminus \{e_m\}.\ \mathsf{com}(e) \in E_o \\
\qquad \wedge \big(\forall l \in \mathcal{L}^x \cap E_o.\ (l, e_m) \in (\mathsf{lhb}|_{E_o})|_{\mathrm{imm}} \Rightarrow (l, e_m) \in \mathsf{po}\big)
\end{array}
\Rightarrow \mathsf{isMX}(G_x) \tag{1}
$$

where $\max(E_o, \mathsf{lhb})$ denotes the maximal elements in E_o with respect to lhb – note that lhb is total on E_o due to $\mathsf{isMX}(G_x|_{E_o})$.

The desired result then follows immediately from (1) and the definitions of E_o^0 and E_r^0. To show (1) we proceed by induction on n.

Base case $n = 0$

From the assumptions of the left-hand side we have $E_r = \emptyset$ and thus $E = E_o$. As such, from isMX($G_x|_{E_o}$) we have isMX(G_x), as required.

Inductive case $n{=}k{+}1$

$$\begin{array}{l}\forall j \in \mathbb{N}.\ \forall E_r, E_o. \\ \quad j \leq k \wedge E{=}E_o \uplus E_r \wedge \texttt{numL}(E_r){=}n \wedge \texttt{numU}(E_r) \leq \texttt{numL}(E_r) \wedge E_r \cap C{=}\emptyset \\ \quad \wedge [E_r]; \mathsf{lhb}; [E_o]{=}\emptyset \wedge \mathsf{isMX}(G_x|_{E_o}) \wedge G_x|_{E_r} \text{is MX-well-formed on } x \\ \quad \wedge \exists! e_m.\ e_m{=}\mathsf{max}(E_o, \mathsf{lhb}) \wedge e_m \in C^x \cup \mathcal{U}^x \wedge \forall e \in E_o \setminus \{e_m\}.\ \mathsf{com}(e) \in E_o \\ \qquad \wedge \big(\forall l \in \mathcal{L}^x \cap E_o.\ (l, e_m) \in (\mathsf{lhb}|_{E_o})|_{\mathrm{imm}} \Rightarrow (l, e_m) \in \mathsf{po}\big) \end{array} \Rightarrow \mathsf{isMX}(G_x) \tag{I.H.}$$

Pick an arbitrary E_r, E_o, e_m such that $E{=}E_o \uplus E_r$, $\texttt{numL}(E_r){=}n$, $\texttt{numU}(E_r) \leq \texttt{numL}(E_r)$, $E_r \cap C{=}\emptyset$, $[E_r]; \mathsf{lhb}; [E_o]{=}\emptyset$, isMX($G_x|_{E_o}$), $G_x|_{E_r}$ is MX-well-formed on x. e_m is unique and $e_m = \mathsf{max}(E_o, \mathsf{lhb})$, $e_m \in C^x \cup \mathcal{U}^x$, $\forall e \in E_o \setminus \{e_m\}.\ \mathsf{com}(e) \in E_o$, and $\forall l \in \mathcal{L}^x \cap E_o.\ (l, e_m) \in (\mathsf{lhb}|_{E_o})|_{\mathrm{imm}} \Rightarrow (l, e_m) \in \mathsf{po}$.

Pick an event $e_l \in E_r$ such that it is minimal in lhb; that is, $\forall e \in E_r \setminus \{e_l\}.\ (e, e_l) \neq \mathsf{lhb}$. As e_l is minimal with respect to lhb, po $\subseteq$ lhb, and since from the MX-well-formedness of $G_x|_{E_r}$ we know that the first event (in po order) in each thread is either a constructor or lock event, we know that $e_l \in C^x \cup \mathcal{L}^x$. Moreover, since $E_r \cap C^x = \emptyset$, we know that $e_l \in \mathcal{L}^x$. We next demonstrate that $(e_m, e_l) \in \mathsf{com} = \mathsf{so} \subseteq \mathsf{lhb}$.

From the consistency of G_x we know there exists $u \in C^x \cup \mathcal{U}^x$ such that $(u, e_l) \in \mathsf{com}$. There are then three cases to consider: a) $u = e_m$; or b) $u \in E_o \setminus \{e_m\}$; or c) $u \in E_r$. In case (a) we then have $(e_m, e_l) \in \mathsf{com}$ as required. In case (b) from $\forall e \in E_o \setminus \{e_m\}.\ \mathsf{com}(e) \in E_o$ we know there exists another $l \in E_o$ such that $(u, l) \in \mathsf{com}$. Since $l \in E_o$ and $e_l \in E_r$ we know $l \neq e_l$. Consequently, we have $(u, l), (u, e_l) \in \mathsf{com}$, contradicting the assumption that com is functional (since G_x is consistent). In case (c), from the well-formedness of $G_x|_{E_r}$ we know there exists l such that $(l, u) \in \mathsf{po}|_{\mathrm{imm}}$. We then have $l \xrightarrow{\mathsf{po}} u \xrightarrow{\mathsf{com}} e_l$; that is we have $l \xrightarrow{\mathsf{lhb}} e_l$, contradicting the assumption that e_l is a minimal element of E_r with respect to lhb.

From the MX-well-formedness of $G_x|_{E_r}$ we know there exists $e_u \in \mathcal{U}^x$ such that $(e_l, e_u) \in \mathsf{po}|_{\mathrm{imm}}$. As po $\subseteq$ lhb and $[E_r]; \mathsf{lhb}; [E_o]{=}\emptyset$, we know that $e_u \in E_r$. Let us then define $E'_o = E_o \uplus \{e_l, e_u\}$ and $E'_r = E_r \setminus \{e_l, e_u\}$; i.e. (1) $E = E'_o \uplus E'_r$, $\texttt{numL}(E'_r){=}n{-}1{=}k$, $\texttt{numU}(E'_r) \leq \texttt{numL}(E'_r)$, $E'_r \cap C{=}\emptyset$.

We next demonstrate that $[E'_r]; \mathsf{lhb}; [E'_o]{=}\emptyset$. As we already have $[E_r]; \mathsf{lhb}; [E_o]{=}\emptyset$ from the assumption, it suffices to show: $\forall e \in E'_r.\ (e, e_l) \notin \mathsf{lhb}$ and $\forall e \in E'_r.\ (e, e_u) \notin \mathsf{lhb}$. The former follows from the fact that e_l is a minimal element of E_r with respect to lhb. For the latter, let us proceed by contradiction and assume there exists $e \in E'_r$ such that $(e, e_u) \in \mathsf{lhb}$. As e_u is an unlock event without incoming so edges and $(e_l, e_u) \in \mathsf{po}|_{\mathrm{imm}}$, we know $(e, e_l) \in \mathsf{lhb}$. This however contradicts our assumption that e_l is a minimal element of E_r with respect to lhb. We thus have: (2) $[E'_r]; \mathsf{lhb}; [E'_o]{=}\emptyset$.
We next show that isMX($G_x|_{E'_o}$) holds. As isMX($G_x|_{E_o}$) holds, we know:
  lhb is a strict total order on E_o;
  $G_x|_{E_o}.\mathsf{lhb}|_{\mathrm{imm}} \subseteq (C^x \times \mathcal{L}^x) \cup (\mathcal{L}^x \times \mathcal{U}^x) \cup (\mathcal{U}^x \times \mathcal{L}^x)$; and
  $[\mathcal{L}^x]; (G_x|_{E_o}.\mathsf{lhb} \setminus G_x|_{E_o}.\mathsf{po}); [\mathcal{L}^x \cup \mathcal{U}^x] \subseteq G_x|_{E_o}.\mathsf{po}; [\mathcal{U}^x]; G_x|_{E_o}.\mathsf{lhb}$.
As $e_m = \mathsf{max}(E_o, \mathsf{lhb})$ and $e_m \xrightarrow{\mathsf{lhb}} e_l \xrightarrow{\mathsf{po}} e_u$, we know:
  lhb is a strict total order on E'_o and
  $G_x|_{E'_o}.\mathsf{lhb}|_{\mathrm{imm}} \subseteq (C^x \times \mathcal{L}^x) \cup (\mathcal{L}^x \times \mathcal{U}^x) \cup (\mathcal{U}^x \times \mathcal{L}^x)$.
Lastly, since $\forall l \in \mathcal{L}^x \cap E_o.\ (l, e_m) \in (\mathsf{lhb}|_{E_o})|_{\mathrm{imm}} \Rightarrow (l, e_m) \in \mathsf{po}$, we have
  $[\mathcal{L}^x]; (G_x|_{E'_o}.\mathsf{lhb} \setminus G_x|_{E'_o}.\mathsf{po}); [\mathcal{L}^x \cup \mathcal{U}^x] \subseteq G_x|_{E'_o}.\mathsf{po}; [\mathcal{U}^x]; G_x|_{E'_o}.\mathsf{lhb}$.
We thus know: (3) isMX($G_x|_{E'_o}$) holds.
Since $G_x|_{E_r}$ is MX-well-formed on x, from the definition of E'_r we have (4) $G_x|_{E'_r}$ is MX-well-formed on x. As $e_m = \mathsf{max}(E_o, \mathsf{lhb})$ and $e_m \xrightarrow{\mathsf{lhb}} e_l \xrightarrow{\mathsf{po}} e_u$, from teh definition of E'_o and since $e_u \in \mathcal{U}^x$ we

have: (5) $e_u = \max(E'_o, \mathsf{lhb})$ and $e_u \in C^x \cup \mathcal{U}^x$. Moreover, as $\forall e \in E_o \setminus \{e_m\}.\ \mathsf{com}(e) \in E_o$, and $(e_m, e_l) \in \mathsf{com}$, from the definition of E'_o we have (6) $\forall e \in E'_o \setminus \{e_u\}.\ \mathsf{com}(e) \in E'_o$. Lastly, since $e_m = \max(E_o, \mathsf{lhb})$ and $e_m \xrightarrow{\mathsf{lhb}} e_l \xrightarrow{\mathsf{po}} e_u$, and $(e_l, e_u) \in \mathsf{po}$, we know that:
(7) $\forall l \in \mathcal{L}^x \cap E'_o.\ (l, e_u) \in (\mathsf{lhb}|_{E'_o})|_{\mathrm{imm}} \Rightarrow (l, e_u) \in \mathsf{po}$.

Consequently, from (1)-(7) and (I.H.) we have $\mathsf{isMX}(G)$ as required.

□

B.2 Correctness of Theorem 3

Given a relation r, let us define the following relations for for all $n, m > 0$ and $k \geq 0$:

$$\begin{array}{lll} \mathsf{A}(\mathsf{r}) \triangleq \mathsf{com}^{-1}; \mathsf{r} & \mathsf{B}(\mathsf{r}) \triangleq \mathsf{com}; \mathsf{r} & \\ \mathsf{C}^{n,m}(\mathsf{r}) \triangleq (\mathsf{A}(\mathsf{r}); \mathsf{C}^{n-1,m}) \cup (\mathsf{B}(\mathsf{r}); \mathsf{C}^{n,m-1}) & \mathsf{C}^{k,0} \triangleq \mathsf{A}(\mathsf{r})^k & \mathsf{C}^{0,k} \triangleq \mathsf{B}(\mathsf{r})^k \end{array} \tag{2}$$

Pick an arbitrary execution $G = (E, \mathsf{po}, \mathsf{com}, \mathsf{so}, \mathsf{lhb})$ of the queue library such that $\mathsf{irreflexive}(\mathsf{C}^{n,n})$ holds for all $n > 0$. Let $\mathsf{to}_0 \triangleq \mathsf{lhb}$, $TO_0 = \{\mathsf{to}_0\}$ and for all $i \geq 0$ let us define:

$$\begin{aligned} TO_{i+1} \triangleq\ & \left\{ (\mathsf{to}_i \cup \{(d_1, d_2)\})^+ \;\middle|\; \mathsf{to}_i \in TO_i \wedge (P_1(\mathsf{to}_i, d_1, d_2) \vee P_2(\mathsf{to}_i, d_1, d_2)) \right\} \\ & \cup \left\{ (\mathsf{to}_i \cup \{(e_1, e_2)\})^+ \;\middle|\; \mathsf{to}_i \in TO_i \wedge (P_3(\mathsf{to}_i, e_1, e_2) \vee P_4(\mathsf{to}_i, e_1, e_2)) \right\} \\ & \cup \left\{ \mathsf{to}_i \;\middle|\; \begin{array}{l} \mathsf{to}_i \in TO_i \wedge \forall d_1, d_2 \in \mathcal{D}^q.\ \forall e_1, e_2 \in \mathcal{E}^q. \\ \quad \neg P_1(\mathsf{to}_i, d_1, d_2) \wedge \neg P_2(\mathsf{to}_i, d_1, d_2) \wedge \neg P_3(\mathsf{to}_i, e_1, e_2) \wedge \neg P_4(\mathsf{to}_i, e_1, e_2) \end{array} \right\} \end{aligned}$$

$$\begin{aligned} P_1(\mathsf{to}_i, d_1, d_2) \overset{\text{def}}{\Leftrightarrow}\ & d_1, d_2 \in \mathcal{D}^q \wedge (d_1, d_2) \notin \mathsf{to}_i \cup \mathsf{to}_i^{-1} \wedge \exists e_1, e_2. \\ & (e_1, d_1), (e_2, d_2) \in \mathsf{com} \wedge (e_1, e_2) \in \mathsf{to}_i \\ P_2(\mathsf{to}_i, d_1, d_2) \overset{\text{def}}{\Leftrightarrow}\ & d_1, d_2 \in \mathcal{D}^q \wedge (d_1, d_2) \notin \mathsf{to}_i \cup \mathsf{to}_i^{-1} \wedge \nexists.\ e_1, e_2 \\ & (e_1, d_1), (e_2, d_2) \in \mathsf{com} \wedge (e_1, e_2) \in \mathsf{to}_i \cup \mathsf{to}_i^{-1} \wedge \forall n \in \mathbb{N}^+.\ \mathsf{irreflexive}(\mathsf{C}^{n,n}(\mathsf{to}_{i+1})) \\ P_3(\mathsf{to}_i, e_1, e_2) \overset{\text{def}}{\Leftrightarrow}\ & e_1, e_2 \in \mathcal{E}^q \wedge (e_1, e_2) \notin \mathsf{to}_i \cup \mathsf{to}_i^{-1} \wedge \exists d_1, d_2. \\ & (e_1, d_1), (e_2, d_2) \in \mathsf{com} \wedge (d_1, d_2) \in \mathsf{to}_i \\ P_4(\mathsf{to}_i, e_1, e_2) \overset{\text{def}}{\Leftrightarrow}\ & e_1, e_2 \in \mathcal{E}^q \wedge (e_1, e_2) \notin \mathsf{to}_i \cup \mathsf{to}_i^{-1} \wedge \nexists.\ d_1, d_2 \\ & (e_1, d_1), (e_2, d_2) \in \mathsf{com} \wedge (d_1, d_2) \in \mathsf{to}_i \cup \mathsf{to}_i^{-1} \wedge \forall n \in \mathbb{N}^+.\ \mathsf{irreflexive}(\mathsf{C}^{n,n}(\mathsf{to}_{i+1})) \end{aligned}$$

In what follows, we write A_i for $\mathsf{A}(\mathsf{to}_i)$, B_i for $\mathsf{B}(\mathsf{to}_i)$ and $\mathsf{C}_i^{n,m} \triangleq \mathsf{C}^{n,m}(\mathsf{to}_i)$, when the choice of to_i is clear from the context.
We next demonstrate that:

$$\forall i \in \mathbb{N}.\ \forall \mathsf{to}_i \in TO_i.\ \forall n \in \mathbb{N}^+.\ \mathsf{irreflexive}(\mathsf{C}_i^{n,n}) \tag{3}$$

Proof. We proceed by induction on i.

Base case $i = 0$
Follows immediately from the definition of $TO_0 = \{\mathsf{to}_0\}$ (as $\mathsf{to}_0 \triangleq \mathsf{lhb}$) and the assumption of the lemma.

Inductive case $i = j+1$

$$\forall k \leq j.\ \forall \mathsf{to}_k \in TO_k.\ \forall n \in \mathbb{N}^+.\ \mathsf{irreflexive}(\mathsf{C}_k^{n,n}) \tag{I.H.}$$

Pick an arbitrary $\mathsf{to}_i \in TO_i$. Let us proceed by contradiction and assume that there exists a $\mathsf{C}_i^{k,k}$ cycle for some $k > 0$. Given the definition of to_i, there are now five cases to consider. The proof of the second and fourth cases follow immediately from the definition of to_i. The proof of the fifth case follows from (I.H.).

Case 1

We then know there exist $\mathsf{to}_j \in TO_j$ and e_1, e_2, d_1, d_2 such that $\mathsf{to}_i = \mathsf{to}_j \cup \{(d_1, d_2)\}$, $d_1, d_2 \in \mathcal{D}^q$, $(d_1, d_2) \notin \mathsf{to}_j \cup \mathsf{to}_j^{-1}$, $(e_1, d_1), (e_2, d_2) \in \mathsf{com}$ and $(e_1, e_2) \in \mathsf{to}_j$.

As $\mathsf{C}_j^{n,n}$ is irreflexive for all $n > 0$, we know the $\mathsf{C}_i^{k,k}$ cycle involves the newly added edge $d_1 \xrightarrow{\mathsf{to}_i} d_2$. That is, either i) $e_1 \xrightarrow{\mathsf{com}} d_1 \xrightarrow{\mathsf{to}_i} d_2 \xrightarrow{\mathsf{C}_j^{k,k-1}} e_1$; or ii) there exist a, b such that $a \xrightarrow{\mathsf{to}_j^?} d_1 \xrightarrow{\mathsf{to}_i} d_2 \xrightarrow{\mathsf{to}_j} b \xrightarrow{\mathsf{C}_j^{k,k}} a$.

In case (i), as d_2 is a dequeue event (and cannot have an outgoing B_j edge), we know its outgoing edge is an A_j edge. That is, there exists a such that $d_2 \xrightarrow{\mathsf{com}^{-1}} e_2 \xrightarrow{\mathsf{to}_j} a \xrightarrow{\mathsf{C}_j^{k-1,k-1}} e_1$. We thus have $a \xrightarrow{\mathsf{C}_j^{k-1,k-1}} e_1 \xrightarrow{\mathsf{to}_j} e_2 \xrightarrow{\mathsf{to}_j} a$. As to_j is transitively closed, we have $a \xrightarrow{\mathsf{C}_j^{k-1,k-1}} e_1 \xrightarrow{\mathsf{to}_j} a$. From the definition of $\mathsf{C}_j^{k-1,k-1}$ we know that it ends with to_j. As such, we have $a \xrightarrow{\mathsf{C}_j^{k-1,k-1}} a$, contradicting (I.H.).

In case (ii), we also have $d_1 \xrightarrow{\mathsf{com}^{-1}} e_1 \xrightarrow{\mathsf{to}_j} e_2 \xrightarrow{\mathsf{com}} d_2$. We thus have $d_1 \xrightarrow{\mathsf{com}^{-1}} e_1 \xrightarrow{\mathsf{to}_j} e_2 \xrightarrow{\mathsf{com}} d_2 \xrightarrow{\mathsf{to}_j} b \xrightarrow{\mathsf{C}_j^{k,k}} a \xrightarrow{\mathsf{to}_j^?} d_1$. That is, we have $d_1 \xrightarrow{\mathsf{A}_j} e_2 \xrightarrow{\mathsf{B}_j} b \xrightarrow{\mathsf{C}_j^{k,k}} a \xrightarrow{\mathsf{to}_j^?} d_1$. From the definition of $\mathsf{C}_j^{k,k}$ we know that it ends with to_j. As to_j is transitively closed and $a \xrightarrow{\mathsf{to}_j^?} d_1$, we thus also have $b \xrightarrow{\mathsf{C}_j^{k,k}} d_1$. We then have $d_1 \xrightarrow{\mathsf{A}_j} e_2 \xrightarrow{\mathsf{B}_j} b \xrightarrow{\mathsf{C}_j^{k,k}} d_1$. That is, $d_1 \xrightarrow{\mathsf{C}_j^{k+1,k+1}} d_1$, contradicting (I.H.).

Case 3

We then know there exist $\mathsf{to}_j \in TO_j$ and e_1, e_2, d_1, d_2 such that $\mathsf{to}_i = \mathsf{to}_j \cup \{(e_1, e_2)\}$, $e_1, e_2 \in \mathcal{D}^q$, $(e_1, e_2) \notin \mathsf{to}_j \cup \mathsf{to}_j^{-1}$, $(e_1, d_1), (e_2, d_2) \in \mathsf{com}$ and $(d_1, d_2) \in \mathsf{to}_j$.

As $\mathsf{C}_j^{n,n}$ is irreflexive for all $n > 0$, we know the $\mathsf{C}_i^{k,k}$ cycle involves the newly added edge $e_1 \xrightarrow{\mathsf{to}_i} e_2$. That is, either i) $d_1 \xrightarrow{\mathsf{com}^{-1}} e_1 \xrightarrow{\mathsf{to}_i} e_2 \xrightarrow{\mathsf{C}_j^{k-1,k}} d_1$; or ii) there exist a, b such that $a \xrightarrow{\mathsf{to}_j^?} e_1 \xrightarrow{\mathsf{to}_i} e_2 \xrightarrow{\mathsf{to}_j} b \xrightarrow{\mathsf{C}_j^{k,k}} a$.

In case (i), as e_2 is an enqueue event (and cannot have an outgoing A_j edge), we know its outgoing edge is an B_j edge. That is, there exists a such that $e_2 \xrightarrow{\mathsf{com}} d_2 \xrightarrow{\mathsf{to}_j} a \xrightarrow{\mathsf{C}_j^{k-1,k-1}} d_1$. We thus have $a \xrightarrow{\mathsf{C}_j^{k-1,k-1}} d_1 \xrightarrow{\mathsf{to}_j} d_2 \xrightarrow{\mathsf{to}_j} a$. As to_j is transitively closed, we have $a \xrightarrow{\mathsf{C}_j^{k-1,k-1}} d_1 \xrightarrow{\mathsf{to}_j} a$. From the definition of $\mathsf{C}_j^{k-1,k-1}$ we know that it ends with to_j. As such, we have $a \xrightarrow{\mathsf{C}_j^{k-1,k-1}} a$, contradicting (I.H.).

In case (ii), we also have $e_1 \xrightarrow{\mathsf{com}} d_1 \xrightarrow{\mathsf{to}_j} d_2 \xrightarrow{\mathsf{com}^{-1}} e_2$. We thus have $e_1 \xrightarrow{\mathsf{com}} d_1 \xrightarrow{\mathsf{to}_j} d_2 \xrightarrow{\mathsf{com}^{-1}} e_2 \xrightarrow{\mathsf{to}_j} b \xrightarrow{\mathsf{C}_j^{k,k}} a \xrightarrow{\mathsf{to}_j^?} d_1$. That is, we have $e_1 \xrightarrow{\mathsf{B}_j} e_2 \xrightarrow{\mathsf{A}_j} b \xrightarrow{\mathsf{C}_j^{k,k}} a \xrightarrow{\mathsf{to}_j^?} e_1$. From the definition of $\mathsf{C}_j^{k,k}$ we know that it ends with to_j. As to_j is transitively closed and $a \xrightarrow{\mathsf{to}_j^?} d_1$, we thus also have $b \xrightarrow{\mathsf{C}_j^{k,k}} e_1$. We then have $e_1 \xrightarrow{\mathsf{B}_j} d_2 \xrightarrow{\mathsf{A}_j} b \xrightarrow{\mathsf{C}_j^{k,k}} d_1$. That is, $e_1 \xrightarrow{\mathsf{C}_j^{k+1,k+1}} e_1$, contradicting (I.H.).

□

We next demonstrate that:

$$\forall i \in \mathbb{N}.\ \forall \mathsf{to}_i \in TO_i.\ \forall \mathsf{to}_{i+1} \in TO_{i+1}.\ \mathsf{to}_i = \mathsf{to}_{i+1} \Rightarrow \\ \forall d_1, d_2 \in \mathcal{D}^q.\ (d_1, d_2) \in \mathsf{to}_i \cup \mathsf{to}_i^{-1} \tag{4}$$

Proof. Pick an arbitrary $i \in \mathbb{N}$, $\mathsf{to}_i \in TO_i$ and $\mathsf{to}_{i+1} \in TO_{i+1}$ such that $\mathsf{to}_i = \mathsf{to}_{i+1}$. We then proceed by contradiction. Let us assume there exist $d_1, d_2 \in \mathcal{D}^q$ such that $(d_1, d_2) \notin \mathsf{to}_i \cup \mathsf{to}_i^{-1}$. Let us write e_1 for $\mathsf{com}^{-1}(d_1)$ when it exists (i.e. when $(e_1, d_1) \in \mathsf{com}$) and write e_2 for $\mathsf{com}^{-1}(d_2)$ when it exists (i.e. when $(e_2, d_2) \in \mathsf{com}$). Let $S_1 = \mathsf{to}_i \uplus \{(d_1, d_2)\}$ and $S_2 = \mathsf{to}_i \uplus \{(d_2, d_1)\}$. From the definition of to_{i+1} we then know that $(e_1, e_2) \notin \mathsf{to}_i$ and there exist k, n such that $\neg\mathsf{irreflexive}(\mathsf{C}^{k,k}(S_1))$ and $\neg\mathsf{irreflexive}(\mathsf{C}^{n,n}(S_2))$.

As from (3) we know $\mathsf{irreflexive}(\mathsf{C}_i^{k,k})$ holds, we know the cycle in $\mathsf{C}^{k,k}(S_1)$ is due to the (d_1, d_2) edge. That is, either 1) $d_1 \xrightarrow{S_1 \setminus \mathsf{to}_i} d_2 \xrightarrow{\mathsf{C}_i^{k,k}} d_1$; or 2) there exist a, b such that $a \xrightarrow{\mathsf{to}_i^?} d_1 \xrightarrow{S_1 \setminus \mathsf{to}_i} d_2 \xrightarrow{\mathsf{to}_i} b \xrightarrow{\mathsf{C}_i^{k,k}} a$. Similarly, as from (3) we know $\mathsf{irreflexive}(\mathsf{C}_i^{n,n})$ holds, we know the cycle in $\mathsf{C}^{n,n}(S_2)$ is due to the (d_2, d_1) edge: either a) $d_2 \xrightarrow{S_2 \setminus \mathsf{to}_i} d_1 \xrightarrow{\mathsf{C}_i^{n,n}} d_2$; or b) there exist f, g such that $f \xrightarrow{\mathsf{to}_i^?} d_2 \xrightarrow{S_2 \setminus \mathsf{to}_i} d_1 \xrightarrow{\mathsf{to}_i} g \xrightarrow{\mathsf{C}_i^{n,n}} f$.

There are now four cases to consider. In case (1.a) we have $d_1 \xrightarrow{\mathsf{C}_i^{n,n}} d_2 \xrightarrow{\mathsf{C}_i^{k,k}} d_1$, i.e. $d_1 \xrightarrow{\mathsf{C}_i^{n+k,n+k}} d_1$, contradicting our result in (3).

In case (1.b) we then have $g \xrightarrow{\mathsf{C}_i^{n,n}} f \xrightarrow{\mathsf{to}_i^?} d_2 \xrightarrow{\mathsf{C}_i^{k,k}} d_1 \xrightarrow{\mathsf{to}_i} g$. As from the definitions of $\mathsf{C}_i^{n,n}$ and $\mathsf{C}_i^{k,k}$ we know they end with to_i, we then have $g \xrightarrow{\mathsf{C}_i^{n,n}} d_2 \xrightarrow{\mathsf{C}_i^{k,k}} g$. That is, we have $g \xrightarrow{\mathsf{C}_i^{n+k,n+k}} g$, contradicting our result in (3).

Similarly, in (2.a) we have $b \xrightarrow{\mathsf{C}_i^{k,k}} a \xrightarrow{\mathsf{to}_i^?} d_1 \xrightarrow{\mathsf{C}_i^{n,n}} d_2 \xrightarrow{\mathsf{to}_i} b$. As from the definitions of $\mathsf{C}_i^{n,n}$ and $\mathsf{C}_i^{k,k}$ we know they end with to_i, we then have $b \xrightarrow{\mathsf{C}_i^{k,k}} d_1 \xrightarrow{\mathsf{C}_i^{n,n}} b$. That is, we have $b \xrightarrow{\mathsf{C}_i^{n+k,n+k}} b$, contradicting our result in (3).

Lastly, in (2.b) we have $b \xrightarrow{\mathsf{C}_i^{k,k}} a \xrightarrow{\mathsf{to}_i^?} d_1 \xrightarrow{\mathsf{to}_i} g \xrightarrow{\mathsf{C}_i^{n,n}} f \xrightarrow{\mathsf{to}_i^?} d_2 \xrightarrow{\mathsf{to}_i} b$. As from the definitions of $\mathsf{C}_i^{n,n}$ and $\mathsf{C}_i^{k,k}$ we know they end with to_i and to_i is transitively closed, we then have $b \xrightarrow{\mathsf{C}_i^{k,k}} g \xrightarrow{\mathsf{C}_i^{n,n}} b$. That is, we have $b \xrightarrow{\mathsf{C}_i^{n+k,n+k}} b$, contradicting our result in (3). □

Similarly, we can demonstrate that:

$$\forall i \in \mathbb{N}.\ \forall \mathsf{to}_i \in TO_i.\ \forall \mathsf{to}_{i+1} \in TO_{i+1}.\ \mathsf{to}_i = \mathsf{to}_{i+1} \Rightarrow \\ \forall e_1, e_2 \in \mathcal{E}^q.\ (e_1, e_2) \in \mathsf{to}_i \cup \mathsf{to}_i^{-1} \tag{5}$$

Theorem 7. *For a given tuple $(E, \mathsf{po}, \mathsf{com}, \mathsf{so}, \mathsf{lhb})$, the $\mathsf{C}^{n,n}$ is irreflexive for all $n \in \mathbb{N}^+$ iff condition (7) on page 20 holds.*

Proof. For the $\Rightarrow$ direction, pick an arbitrary execution $G = (E, \mathsf{po}, \mathsf{com}, \mathsf{so}, \mathsf{lhb})$ of the queue library such that $\mathsf{irreflexive}(\mathsf{C}^{n,n})$ holds for all $n > 0$. Let $m \in \mathbb{N}$ be the least natural number for which there exist $\mathsf{to}_m \in TO_m$ and $\mathsf{to}_{m+1} \in TO_{m+1}$ such that $\mathsf{to}_m = \mathsf{to}_{m+1}$. Let us then define to as an arbitrary extension of to_m to a strict total order. From the definition of to_m we then know that to agrees with lhb. Let H denote the enumeration of events in E according to to. From (3), (4) and (5) it is then straightforward to demonstrate that $\mathsf{fifo}(\epsilon, H)$ holds.

For the $\Leftarrow$ direction pick an arbitrary execution $G = (E, \mathsf{po}, \mathsf{com}, \mathsf{so}, \mathsf{lhb})$ with a sequential enumeration H of $E \setminus C^q$ such that: (i) H agrees with lhb; (ii) $\mathrm{fifo}(\epsilon, H)$ holds. Let to denote the strict total order on E induced by H. We next demonstrate that:

$$\forall n \in \mathbb{N}^+.\ \mathsf{C}^{n,n}(\mathsf{to}) \subseteq \mathsf{to}$$

We then have $\mathsf{C}^{n,n} \triangleq \mathsf{C}^{n,n}(\mathsf{lhb}) \subseteq \mathsf{C}^{n,n}(\mathsf{to})$. As such, the irreflexivity of $\mathsf{C}^{n,n}$ for an arbitrary $n \in \mathbb{N}^+$ simply follows from above and the irreflexivity of the strict total order to.

To show that $\forall n \in \mathbb{N}^+.\ \mathsf{C}^{n,n}(\mathsf{to}) \subseteq \mathsf{to}$, we proceed by induction on n.

Base case $n = 1$

Pick an arbitrary $(a, b) \in \mathsf{C}^{1,1}(\mathsf{to})$. From the definition of $\mathsf{C}^{1,1}(\mathsf{to})$ we then know that there exists c such that either 1) $(a, c) \in \mathsf{com}^{-1}; \mathsf{to}; \mathsf{com}$ and $(c, b) \in \mathsf{to}$; or 2) $(a, c) \in \mathsf{com}; \mathsf{to}; \mathsf{com}^{-1}$ and $(c, b) \in \mathsf{to}$.

In both cases from $\mathrm{fifo}(\epsilon, H)$ and the definition of to we know that $(a, c) \in \mathsf{to}$. As in both cases we have $(c, b) \in \mathsf{to}$ and to is transitively closed, we have $(a, b) \in \mathsf{to}$ as required.

Base case $n = k{+}1$

$$\forall m \in \mathbb{N}^+.\ m \leq k \Rightarrow \mathsf{C}^{m,m}(\mathsf{to}) \subseteq \mathsf{to} \qquad \text{(I.H.)}$$

Pick an arbitrary $(a, b) \in \mathsf{C}^{n,n}(\mathsf{to})$. From the definition of $\mathsf{C}^{n,n}$ we know there exists at least one adjacent pair of $\mathsf{A}(\mathsf{to})$ and $\mathsf{B}(\mathsf{to})$ edges. That is, there exists i, j, c, d such that: $a \xrightarrow{\mathsf{C}^{i,j}(\mathsf{to})} c \xrightarrow{\mathsf{C}^{1,1}(\mathsf{to})} d \xrightarrow{\mathsf{C}^{k-i,k-j}(\mathsf{to})} b$. As such, from the proof of the base case we know $a \xrightarrow{\mathsf{C}^{i,j}(\mathsf{to})} c \xrightarrow{\mathsf{to}} d \xrightarrow{\mathsf{C}^{k-i,k-j}(\mathsf{to})} b$. As the $\mathsf{C}^{i,j}(\mathsf{to})$ path ends with a to edge for all i, j and to is transitive, we have $a \xrightarrow{\mathsf{C}^{i,j}(\mathsf{to})} d \xrightarrow{\mathsf{C}^{k-i,k-j}(\mathsf{to})} b$. That is, $a \xrightarrow{\mathsf{C}^{k,k}(\mathsf{to})} b$. Consequently, from (I.H.) we have $(a, b) \in \mathsf{to}$, as required.

□

C ADDITIONAL SPECIFICATIONS

C.1 Multiple-Readers-Single-Writer Lock Library Specification

We consider a multiple-readers-single-writer (MRSW) lock library with six methods: 1) *new-MSRW*( ), for constructing a lock; 2) *wlock*(x), for acquiring x in writer mode; 3) *wunlock*(x), for releasing the writer lock on x; 4) *rlock*(x), for acquiring x in reader mode; 5) *runlock*(x), for releasing a reader lock on x; and 6) *plock*(x), for promoting a reader lock on x to a writer one. A reader lock on x is promoted once all reader locks on x (except that of the promoter) are released.

The *MRSW interface* is $\langle \mathcal{M}^{\mathrm{RW}}, \mathcal{M}_c^{\mathrm{RW}}, \mathtt{loc}^{\mathrm{RW}} \rangle$, where $\mathcal{M}_c^{\mathrm{RW}} \triangleq \bigcup_{x \in \mathsf{Loc}} \mathcal{M}_c^x$ with $\mathcal{M}_c^x \triangleq \{\mathtt{new\text{-}MSRW}(x)\}$; $\mathcal{M}^{\mathrm{RW}} \triangleq \bigcup_{x \in \mathsf{Loc}} \mathcal{M}^x$ with $\mathcal{M}^x \triangleq \mathcal{M}_c^x \cup \{\mathtt{wlock}(x), \mathtt{wunlock}(x), \mathtt{rlock}(x), \mathtt{runlock}(x), \mathtt{plock}(x)\}$; and $\forall l \in \mathcal{M}^x.\ \mathtt{loc}^{\mathrm{MX}}(l) = \{x\}$. For an MRSW lock at location x, we define the following event sets:

$$\mathcal{WL}^x \triangleq \{e \mid \mathtt{lab}(e){=}\mathtt{wlock}(x)\} \qquad \mathcal{WU}^x \triangleq \{e \mid \mathtt{lab}(e){=}\mathtt{wunlock}(x)\}$$
$$\mathcal{RL}^x \triangleq \{e \mid \mathtt{lab}(e){=}\mathtt{rlock}(x)\} \qquad \mathcal{RU}^x \triangleq \{e \mid \mathtt{lab}(e){=}\mathtt{runlock}(x)\}$$
$$\mathcal{C}^x \triangleq \{e \mid \mathtt{lab}(e){=}\mathtt{new\text{-}MSRW}(x)\} \qquad \mathcal{PL}^x \triangleq \{e \mid \mathtt{lab}(e){=}\mathtt{plock}(x)\}$$

Let $\mathcal{L}^x \triangleq \mathcal{WL}^x \cup \mathcal{RL}^x \cup \mathcal{PL}^x$ and $\mathcal{U}^x \triangleq \mathcal{WU}^x \cup \mathcal{RU}^x$.

A tuple $\langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}, \mathsf{lhb} \rangle$ is *RW-consistent* on x iff:

(1) there is at most one constructor event: $E^{\mathsf{c}} = \emptyset \vee \exists c \in \mathcal{C}^x.\ E^{\mathsf{c}} = \{c\}$;
(2) com relates matching lock events: $\mathsf{com} = \mathsf{com}_w \cup \mathsf{com}_r \cup \mathsf{com}_p$ with:
$\mathsf{com}_w, \mathsf{com}_p \subseteq (\mathcal{C}^x \cup \mathcal{U}^x) \times \mathcal{WL}^x \qquad \mathsf{com}_r \subseteq (\mathcal{C}^x \cup \mathcal{WU}^x) \times \mathcal{RL}^x$

(3) each event is matched by at most one lock except for reader locks, i.e. for all e, e_1, e_2:
$e_1 \neq e_2 \wedge (e, e_1), (e, e_2) \in \mathsf{com} \Rightarrow (e_1, e_2 \in \mathcal{RL}^x) \vee (e_1 \in \mathcal{RL}^x \wedge e_2 \in \mathcal{PL}^x) \vee (e_1 \in \mathcal{PL}^x \wedge e_2 \in \mathcal{RL}^x)$
(4) each lock is matched by at most one event: com^{-1} is functional;
(5) all lock events are matched: $E \cap \mathcal{L}^x = rng(\mathsf{com})$; and
(6) every matching edges is synchronising: $\mathsf{so} = \mathsf{com}$.

Intuitively, com describes the order of lock acquisition. For each $l \in \mathcal{WL}^x$ with $(e, l) \in \mathsf{com}_w$, when $e \in \mathcal{U}^x$ then e denotes the unlock event releasing the lock *immediately* before it is acquired by l; when $e \in C^x$ then e denotes the constructor initialising the lock, and thus l corresponds to the very first $wlock(x)$ call. As l acquires the lock in the (exclusive) writer mode, no other lock may be matched with its predecessor e (see (3)). For each $l \in \mathcal{RL}^x$ with $(e, l) \in \mathsf{com}_r$, the case where $e \in C^x$ can be described analogously; when $e \in \mathcal{WU}^x$, then e denotes the last time the lock in writer mode was released. As MRSW locks allow for multiple reader locks simultaneously, multiple events in $\mathcal{RL}^x$ may be matched with the same event in $\mathcal{WL}^x$ (see (3)). Lastly, for each $l \in \mathcal{PL}^x$ with $(e, l) \in \mathsf{com}_p$, when $e \in \mathcal{RU}^x$ then e denotes the event releasing the last reader lock on x; when $e \in \mathcal{WU}^x$, then e denotes the last time the lock in writer mode was released; when $e \in C^x$, then l denotes the first $plock(x)$ call, prior to any writer lock acquisition. In the latter two cases, at the time of promoting the lock via l, no other reader locks (other than that being promoted) are held on x and thus l follows the last writer lock release or the constructor. As such, the event acquiring a reader lock as well as its subsequent promotion may both be matched by e (see (3)).

A tuple $\langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}, \mathsf{lhb} \rangle$ is *RW-well-formed* on x iff:

- $\min(\mathsf{po}) \subseteq C^x \cup \mathcal{L}^x$ and $\mathsf{po}|_{\text{imm}}(E^c) \subseteq \mathcal{L}^x$;
- $[\mathcal{RL}^x]; \mathsf{po}|_{\text{imm}} = \mathsf{po}|_{\text{imm}}; [\mathcal{PL}^x \cup \mathcal{RU}^x]$; and
- $[\mathcal{WL}^x \cup \mathcal{PL}^x]; \mathsf{po}|_{\text{imm}} = \mathsf{po}|_{\text{imm}}; [\mathcal{WU}^x]$.

Analogously to mutex well-formedness, MRSW well-formedness requires that the first call in each thread be either to the constructor or for lock acquisition, a constructor call be immediately followed (in po) by a lock acquisition; each reader unlock or lock promotion call be immediately preceded (in po) by a reader lock acquisition call and vice versa; and each writer unlock call be immediately preceded (in po) by a writer lock acquisition or lock promotion call and vice versa.

Strong MRSW-Consistency. Note that there are no com edges between events of reader locks. Consequently, as $\mathsf{so} = \mathsf{com}$, reader lock events do not synchronise with one another. This is to keep the specification as general as possible and admit certain MRSW implementations with weaker guarantees such as those discussed in §E. Nevertheless, we can strengthen this specification by extending the domain of com_r as: $\mathsf{com}_r \subseteq (C^x \cup \mathcal{WU}^x \cup \mathcal{RL}^x \cup \mathcal{RU}^x) \times \mathcal{RL}^x$, and the domain of com_p as: $\mathsf{com}_p \subseteq (\mathcal{RU}^x \cup \mathcal{RL}^x) \times \mathcal{PL}^x$. That is, for each $l \in \mathcal{RL}^x$ with $(e, l) \in \mathsf{com}_r$, the case where $e \in C^x$ can be described as before; when $e \in \mathcal{WU}^x \cup \mathcal{RU}^x$ then e denotes the (reader or writer) unlock event releasing the lock *immediately* before it is acquired by l; when $e \in \mathcal{RL}^x$, then e denotes the event acquiring a reader lock on x *immediately* before it is also acquired by l. This is because MRSW locks allow for multiple reader locks simultaneously. Note that this is in contrast to the com_r edges above. In particular, in the above (weaker) description, when $e \in \mathcal{WU}^x$, then e denotes the last release of the writer lock on x before its acquisition by l; i.e. e may not be *immediately* preceding l and may be interleaved by several reader (lock or unlock) events. Similarly, for each $l \in \mathcal{PL}^x$ with $(e, l) \in \mathsf{com}_p$, when $e \in \mathcal{RU}^x$, then e denotes the reader unlock event releasing the lock *immediately* before it is acquired by l; when $e \in \mathcal{RL}^x$, then e denotes the *very reader lock* being promoted: no other reader locks on x have been acquired between its acquisition (e) and its promotion (l). As such, in contrast to (3), we require that com be functional.

We thus denote $\langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}, \mathsf{lhb} \rangle$ as *strongly RW-consistent* on x iff: (1) as above; (2) com =

$\mathsf{com}_w \cup \mathsf{com}_r \cup \mathsf{com}_p$ with $\mathsf{com}_r \subseteq (\mathcal{C}^x \cup \mathcal{WU}^x \cup \mathcal{RL}^x \cup \mathcal{RU}^x) \times \mathcal{RL}^x$, com_w as above, and $\mathsf{com}_p \subseteq (\mathcal{RL}^x \cup \mathcal{RU}^x) \times \mathcal{PL}^x$; (3) com is functional; and (4)-(6) as above.

Definition C.1 (MRSW library)**.** The *MRSW library* is $L^{\mathsf{RW}} \triangleq \langle \mathcal{M}^{\mathsf{RW}}, \mathcal{M}^{\mathsf{RW}}_{\mathsf{c}}, \mathtt{loc}^{\mathsf{RW}}, \mathcal{G}^{\mathsf{RW}}_{\mathsf{c}}, \mathcal{G}^{\mathsf{RW}}_{\mathsf{wf}} \rangle$, where $\mathcal{G}^{\mathsf{RW}}_{\mathsf{c}} \triangleq \left\{ G \in \mathcal{G}_{L^{\mathsf{RW}}} \,\middle|\, \forall x.\, G_x \text{ RW-consistent on } x \right\}$ and $\mathcal{G}^{\mathsf{RW}}_{\mathsf{wf}} \triangleq \left\{ G \in \mathcal{G}_{L^{\mathsf{RW}}} \,\middle|\, \forall x.\, G_x \text{ RW-well-formed on } x \right\}$. Let $\mathcal{G}^{\mathsf{SRW}}_{\mathsf{c}} \triangleq \left\{ G \in \mathcal{G}_{L^{\mathsf{RW}}} \,\middle|\, \forall x.\ G_x \text{ strongly RW-consistent on } x \right\}$; the *strong MRSW library* is $L^{\mathsf{SRW}} \triangleq \langle \mathcal{M}^{\mathsf{RW}}, \mathcal{M}^{\mathsf{RW}}_{\mathsf{c}}, \mathtt{loc}^{\mathsf{RW}}, \mathcal{G}^{\mathsf{SRW}}_{\mathsf{c}}, \mathcal{G}^{\mathsf{RW}}_{\mathsf{wf}} \rangle$.

C.2 Set Library Specification

We consider a set library with four methods: *new-set*( ), for constructing a new set; *add*(s, v) for adding v to the set at s; *rem*(s, v) for removing v from the set at s; and *is-in*(s, v) for checking the membership of v in the set at s. An *add*(s, v) call successfully adds v to s only if it does not already contain v; analogously, a *rem*(s, v) successfully removes v if s contains v. Similarly, the return value of *is-in*(s, v) indicates whether s contains v. As such, all three operations return a boolean reflecting their outcomes. We present a set specification in our framework below. Once again, we forgo a strong specification with a total execution order in the linearisability style, and opt instead for a weaker specification more suitable for the WMC setting.

We define the *set interface* as $\langle \mathcal{M}^{\mathsf{Set}}, \mathcal{M}^{\mathsf{Set}}_{\mathsf{c}}, \mathtt{loc}^{\mathsf{Set}} \rangle$, where $\forall l \in \mathcal{M}^s.\ \mathtt{loc}^{\mathsf{MX}}(l) = \{s\}$ and

$$\mathcal{M}^{\mathsf{Set}}_{\mathsf{c}} \triangleq \bigcup_{s \in \mathsf{Loc}} \mathcal{M}^s_{\mathsf{c}} \qquad \mathcal{M}^s_{\mathsf{c}} \triangleq \{\mathtt{new\text{-}set}(s)\}$$

$$\mathcal{M}^{\mathsf{Set}} \triangleq \bigcup_{s \in \mathsf{Loc}} \mathcal{M}^s \qquad \mathcal{M}^s \triangleq \mathcal{M}^s_{\mathsf{c}} \cup \left\{ \mathtt{add}(s, v, o), \mathtt{rem}(s, v, o), \mathtt{is\text{-}in}(s, v, o) \,\middle|\, v \in \mathsf{Val} \wedge o \in \{\top, \bot\} \right\}$$

For a set at location s, we define the following sets of events:

$$\mathcal{C}^s \triangleq \{e \mid \mathtt{lab}(e) = \mathtt{new\text{-}set}(s)\} \qquad \mathcal{I}^{s,v,o} \triangleq \{e \mid \mathtt{lab}(e) = \mathtt{is\text{-}in}(s, v, o)\}$$

$$\mathcal{A}^{s,v,o} \triangleq \left\{ e \,\middle|\, \mathtt{lab}(e) = \mathtt{add}(s, v, o) \right\} \qquad \mathcal{R}^{s,v,o} \triangleq \{e \mid \mathtt{lab}(e) = \mathtt{rem}(s, v, o)\}$$

Let $\mathcal{A}^{s,o} \triangleq \bigcup_{v \in \mathsf{Val}} \mathcal{A}^{s,v,o}$ and $\mathcal{A}^s \triangleq \mathcal{A}^{s,\top} \cup \mathcal{A}^{s,\bot}$; let us similarly define $\mathcal{R}^{s,o}$, $\mathcal{R}^s$, $\mathcal{I}^{s,o}$ and $\mathcal{I}^s$. A tuple $\langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}, \mathsf{lhb} \rangle$ is *set-consistent on s* iff:

(1) there is at most one constructor event: $E^{\mathsf{c}} = \emptyset \vee \exists c \in \mathcal{C}^s.\ E^{\mathsf{c}} = \{c\}$;
(2) com relates matching events: $\mathsf{com} \triangleq \mathsf{com}_r \cup \mathsf{com}_i \cup \mathsf{com}_f$, where
$$\mathsf{com}_r \subseteq \bigcup_{v \in \mathsf{Val}} \mathcal{A}^{s,v,\top} \times \mathcal{R}^{s,v,\top} \qquad \mathsf{com}_i \subseteq \bigcup_{v \in \mathsf{Val}} \mathcal{A}^{s,v,\top} \times \mathcal{I}^{s,v,\top} \qquad \mathsf{com}_f \subseteq \bigcup_{v \in \mathsf{Val}} \mathcal{A}^{s,v,\top} \times \mathcal{A}^{s,v,\bot}$$
(3) every remove, membership and failed add is matched by at most one add: com^{-1} is functional;
(4) every add is matched by at most one remove: com_r is functional;
(5) every unmatched remove or membership returns $\bot$: $\big(E \cap (\mathcal{R}^s \cup \mathcal{I}^s)\big) \setminus \mathit{rng}(\mathsf{com}) \subseteq \mathcal{R}^{s,\bot} \cup \mathcal{I}^{s,\bot}$;
(6) every failed add event is matched: $(E \cap \mathcal{A}^{s,\bot}) \setminus \mathit{rng}(\mathsf{com}) = \emptyset$;
(7) every matching edge is synchronising: $\mathsf{so} = \mathsf{com}$; and
(8) value v cannot be added twice before being removed first; that is, for all v:
$[\mathcal{A}^{s,v,\top}]; \mathsf{lhb}; [\mathcal{A}^{s,v,\top}]; \mathsf{lhb}; \mathsf{com}_r^{-1}$ is irreflexive and $[\mathcal{A}^{s,v,\top} \setminus \mathit{rng}(\mathsf{com}_r)]; \mathsf{lhb}; [\mathcal{A}^{s,v,\top}] = \emptyset$
(9) adding a value must not fail when it is already removed: $\mathsf{com}_r; \mathsf{lhb}; \mathsf{com}_f^{-1}$ is irreflexive;
(10) removing a value must not fail when the value is yet to be removed:
$\forall v.\ [\mathcal{R}^{s,v,\top} \cup \mathcal{I}^{s,v,\top}]; \mathsf{com}_r^{-1} \cup \mathsf{com}_f^{-1}; \mathsf{lhb}; [\mathcal{R}^{s,v,\bot}]; \mathsf{lhb}$ is irreflexive;
(11) membership check for a value must not fail when the value is yet to be removed:
$\forall v.\ [\mathcal{R}^{s,v,\top} \cup \mathcal{I}^{s,v,\top}]; \mathsf{com}_r^{-1} \cup \mathsf{com}_f^{-1}; \mathsf{lhb}; [\mathcal{I}^{s,v,\bot}]; \mathsf{lhb}$ is irreflexive;
(12) a remove or membership with a previous unmatched add cannot return $\bot$:
$\forall v.\ [\mathcal{A}^{s,v,\top} \setminus \mathit{dom}(\mathsf{com}_r)]; \mathsf{lhb}; [\mathcal{R}^{s,v,\bot} \cup \mathcal{I}^{s,v,\bot}] = \emptyset$;

$$
\begin{array}{lll}
 & \mathit{lock}(x) \triangleq & \\
\mathit{new\text{-}mutex}(\,) \triangleq & \quad \texttt{loop} & \mathit{unlock}(x) \triangleq \\
\quad \texttt{let } x = \mathit{alloc}() \texttt{ in } x & \quad\quad \texttt{if } \mathit{compare\text{-}set}(x, 0, 1, \texttt{acq}) \texttt{ then} & \quad \mathit{store}(x, 0, \texttt{rel}); \\
 & \quad\quad\quad \texttt{break}_1\,() &
\end{array}
$$

Fig. 7. A simple mutex implementation using a release acquire register

(13) successful remove events cannot match with adds which are already removed: $[\mathcal{A}^{s,v,\top}];\mathsf{lhb};[\mathcal{R}^{s,v,\top}];\mathsf{lhb};\mathsf{com}_r^{-1}$ is irreflexive;
(14) successful membership events cannot match with adds which are already removed: $\mathsf{com}_i^{-1};\mathsf{com}_r;\mathsf{lhb}$ is irreflexive.

Intuitively, $(a, r) \in \mathsf{com}_r$ denotes that r removes a value added by a. As such, each successful remove is matched by exactly one add, and each successful add is matched by at most one remove. Similarly, $(a, i) \in \mathsf{com}_i$ denotes that i observes the value added by a. Each successful membership is thus matched by exactly one add. However, as membership calls leave the set unchanged, multiple membership events may be matched by the same add. Lastly, $(a, f) \in \mathsf{com}_f$ denotes that f fails to add its value to the set as it has been previously added by a. Each failed add is hence matched by exactly one successful add, whilst each successful add may be matched by several failed adds.

Definition C.2 (Set library). The *set library* is $L^{\mathsf{Set}} \triangleq \langle \mathcal{M}^{\mathsf{Set}}, \mathcal{M}_{\mathsf{c}}^{\mathsf{Set}}, \mathtt{loc}^{\mathsf{Set}}, \mathcal{G}_{\mathsf{c}}^{\mathsf{Set}}, \mathcal{G}_{L^{\mathsf{Set}}} \rangle$, where $\mathcal{G}_{\mathsf{c}}^{\mathsf{Set}} \triangleq \left\{ G \in \mathcal{G}_{L^{\mathsf{Set}}} \;\middle|\; \forall s.\ G_s \text{ is set-consistent on } s \right\}$.

D A SOUND MUTEX IMPLEMENTATION

In Fig. 7 we present a simple implementation of mutex locks using release-acquire registers. As we formalise in Thm. 8, this implementation is sound with respect to the mutex library L^{MX}.

Theorem 8. *The mutex implementation in Fig. 7 is a sound implementation of L^{MX}.*

Proof. The full proof is mechanised in the Coq proof assistant and is available as auxiliary material. □

```
new-MSRW() ≜
  let x = alloc() in x

rlock(x) ≜
  loop
    let c = load(x, rlx) in
    if is-even(c) then
      if compare-set(x, c, c+2, acqrel) then
        break_1 ()

runlock(x) ≜
  fetch-add(x, −2, rel)

wunlock(x) ≜
  store(x, 0, rel);
```

```
wlock(x) ≜
  loop
    if compare-set(x, 0, 1, acq) then
      break_1 ()

plock(x) ≜
  loop
    let b = 0 in
    if b then
      let c = load(x, acq) in
      if c == 3 then break_1 ()
    else let c = load(x, rlx) in
      if is-even(c) then
        if compare-set(x, c, c+1, acq) then
          b = 1
```

Fig. 8. An implementation of strong MRSW locks using a C11 register

E SOUND MRSW LOCK IMPLEMENTATIONS

E.1 A Sound Strong MRSW Lock Implementation

We present a strong MRSW lock implementation using the C11 registers: a lock at location x is represented as a C11 register at x. The state of a lock x is represented by an integer value. A lock x may hold either:

(1) value 0, denoting that the lock is free (not held in read or write mode); or
(2) value 1, denoting that the lock is held (exclusively) in write mode; or
(3) an even value $2n$ with $n > 0$, denoting that x is held in (shared) read mode by n readers; or
(4) an odd value $2n+3$ with $n > 0$, denoting that the lock is currently being promoted, awaiting the release of n readers; or
(5) value 3, denoting that the lock is successfully promoted.

As such, the implementation of *wlock*(x) simply spins until it can atomically update (via atomic *compare-set*) the value of x from zero (free) to one (acquired in write mode). Dually, the implementation of *wunlock*(x) simply releases the write lock by atomically assigning x to zero.

The implementation of *plock*(x) is more involved. As multiple readers may attempt to promote their reader locks simultaneously, promotion is granted on a 'first-come-first-served' bases. As such, the implementation of *plock*(x) first reads the value of x (the else branch). If x holds an odd value, then another reader is currently promoting x and thus promotion must be retried. On the other hand, if x holds an even value, then its value is atomically incremented (to an odd value) to signal the intention to promote. Moreover, b is set to 1 to indicate that the intention to promote has been successfully registered and promotion can enter the next waiting phase. The implementation then proceeds by spinning until all other readers have released their locks on x (i.e. $x == 3$), at which point x is successfully promoted and the implementation terminates. Note that once a reader has signalled its intention to promote x (by incrementing x to an odd value), any other such attempt to promote the lock on x, as well as calls to acquire it in read mode will fail thereafter until such time that x is released by its current promoter.

The implementation of *rlock*(x) is similar. It first checks whether x is odd (held in write mode or being promoted). If so then the implementation spins until x is even (free or held in read mode), at which point its value is incremented by two (to increase the number of readers by one) using the atomic *fetch-add* operation, and x is successfully acquired in read mode. Dually, the implementation of $runlock(x)$ atomically decrements the value of x by two to decrease the reader count by one.

Implementation Correctness. Let I denote the strong MRSW lock implementation in Fig. 8. To show the soundness of I, we appeal to Thm. 1 and show that I is locally sound on L^{SRW}.

Pick an arbitrary Λ, f, $G{=}\langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}\rangle$, E'', po'' such that G is Λ-consistent and Λ-well-formed and $\mathsf{abs}_{L^{\mathsf{SRW}},I}(f, \langle E, \mathsf{po}\rangle, \langle E'', \mathsf{po}''\rangle)$.
We must next find com'', so'' such that $\langle E'', \mathsf{po}'', \mathsf{com}'', \mathsf{so}'', \mathsf{lhb}''\rangle \in L^{\mathsf{SRW}}.\mathcal{G}_{\mathsf{c}}$, where lhb'' is the same as lhb' in Def. 11.

For each location x, without loss of generality let us assume G contains n_x read lock calls on x, n'_x read unlock calls on x, m_x write lock calls on x, m'_x write unlock calls on x and p_x lock promotion calls on x. Let us enumerate each of read lock calls, read unlock calls, write lock calls, write unlock calls and lock promotion calls arbitrarily. Note that:

- the MRSW constructor at location x contains a single event c_x where $\mathtt{lab}(c_x) = \mathtt{alloc}(x, 0)$.
- For each ith read lock operation on x, the G contains the trace $\theta_i^{rl(x)} = rf^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} rr \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} rl$, where rf^* denotes the events of those iterations that failed to acquire the reader lock, $\mathtt{lab}(rr) = \mathtt{load}(\mathtt{rlx}, x, rv_i)$, $\mathtt{lab}(rl) = \mathtt{compare\text{-}set}(\mathtt{acqrel}, x, rv_i, rv_i{+}2)$, and rv_i is an even value.
- for each ith read unlock operation on x, the G contains the trace $\theta_i^{ru(x)}$ with a single event ru, where $\mathtt{lab}(ru) = \mathtt{fetch\text{-}add}(\mathtt{rel}, x, v_i, v_i{-}2)$ for some v_i.
- for each ith write lock operation on x, the G contains the trace $\theta_i^{wl(x)} = wf^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} wl$, where wf^* denotes the events of those iterations that failed to acquire the writer lock and $\mathtt{lab}(wl) = \mathtt{compare\text{-}set}(\mathtt{acq}, x, 0, 1)$.
- for each ith write unlock operation on x, the G contains the trace $\theta_i^{wu(x)}$ with a single event wu, where $\mathtt{lab}(wu) = \mathtt{store}(\mathtt{rel}, x, 0)$.
- for each ith read lock promotion operation on x, the G contains the trace $\theta_i^{pl(x)} = pf1^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pr \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pi \xrightarrow{\mathsf{po}} pf2^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pl$, where $pf1^*$ denotes the events of those iterations that failed to indicate lock promotion, $pf2^*$ denotes the events of those iterations that failed to promote the lock, $\mathtt{lab}(pr) = \mathtt{load}(\mathtt{rlx}, x, pv_i)$, $\mathtt{lab}(pi) = \mathtt{compare\text{-}set}(\mathtt{acqrel}, x, pv_i, pv_i{+}1)$, pv_i is an even value, and $\mathtt{lab}(pl) = \mathtt{load}(\mathtt{acq}, x, 3)$.

Let $\mathtt{imp}(.) : E'' \longrightarrow E$ be defined as:

$$\mathtt{imp}(e) \triangleq \begin{cases} c_x & \exists x.\ e = f(c_x) \\ \theta_i^{rl(x)}.rl & \exists i, x.\ e = f(\theta_i^{rl(x)}.rl) \\ \theta_i^{ru(x)}.ru & \exists i, x.\ e = f(\theta_i^{ru(x)}.ru) \\ \theta_i^{wl(x)}.wl & \exists i, x.\ e = f(\theta_i^{wl(x)}.wl) \\ \theta_i^{wu(x)}.wu & \exists i, x.\ e = f(\theta_i^{wu(x)}.wu) \\ \theta_i^{pl(x)}.pl & \exists i, x.\ e = f(\theta_i^{pl(x)}.pl) \end{cases}$$

Let us define: $\mathsf{so}'' = \mathsf{com}''$ with com'' defined as follows:

$$\mathsf{com}'' \triangleq \left\{ \begin{array}{l|l} (e_1, e_1'), & x \in \mathsf{Loc} \wedge (\mathtt{imp}(e_1), \mathtt{imp}(e_1')) \in \mathsf{com} \wedge \exists i.\ \mathtt{imp}(e_1') \in \theta_i^{rl(x)} \\ (e_2, e_2'), & \wedge (\mathtt{imp}(e_2), \mathtt{imp}(wl_j^x)) \in \mathsf{com} \wedge \exists j.\ \mathtt{imp}(e_2') \in \theta_j^{wl(x)} \\ (f_a(e_3), f(\theta_k^{pl(x)}.pl)), & \wedge \exists k, e_3.\ (\theta_k^{pl(x)}.pi, \theta_k^{pl(x)}.pl) \in \mathsf{com} \wedge (e_3, \theta_k^{pl(x)}.pi) \in \mathsf{com} \\ (f_a(e_4), f(\theta_h^{pl(x)}.pl)) & \wedge \exists h, e_4.\ (e_4, \theta_h^{pl(x)}.pl) \in \mathsf{com} \wedge e_4 \neq \theta_h^{pl(x)}.pi \end{array} \right\}$$

To show that $G' = \langle E'', \text{po}'', \text{com}'', \text{so}'', \text{lhb}'' \rangle \in L^{\text{SRW}}.\mathcal{G}_c$, we are then required to show for all $x \in \text{Loc}$, G'_x is RW-consistent on x. Pick an arbitrary $x \in \text{Loc}$ and let $G'_x = \langle E', \text{po}', \text{com}', \text{so}', \text{lhb}' \rangle$. We then need to show:

(1) G'_x contains at most one constructor event;
(2) $\text{com}' = \text{com}_w \cup \text{com}_r \cup \text{com}_p$ with:
$\text{com}_w \subseteq (C^x \cup \mathcal{U}^x) \times \mathcal{WL}^x$; $\text{com}_r \subseteq (C^x \cup \mathcal{WU}^x \cup \mathcal{RL}^x \cup \mathcal{RU}^x) \times \mathcal{RL}^x$; and $\text{com}_p \subseteq (\mathcal{RL}^x \cup \mathcal{RU}^x) \times \mathcal{PL}^x$
(3) com' is functional;
(4) com'^{-1} is functional;
(5) $E \cap \mathcal{L}^x = rng(\text{com}')$; and
(6) $\text{so}' = \text{com}'$.

Parts (1), (5) and (6) follow immediately from the construction of G'_x. For part (2), pick an arbitrary $(a, b) \in \text{com}'$. From the definition of com' we then know that there exists i such that either a) $b = rl^x_i$ and $(\texttt{imp}(a), \texttt{imp}(rl^x_i)) \in \text{com}$; or b) $b = wl^x_i$ and $(\texttt{imp}(a), \texttt{imp}(wl^x_i)) \in \text{com}$; or c) $b = pl^x_i$ and there exists e such that $a = f_a(e)$, $(\theta^{pl(x)}_i.pi, \texttt{imp}(pl^x_i)) \in \text{com}$ and $(e, \theta^{pl(x)}_i.pi) \in \text{com}$; or d) $b = pl^x_i$ and there exists e such that $a = f_a(e)$, $(e, \texttt{imp}(pl^x_i)) \in \text{com}$ and $e \neq \theta^{pl(x)}_i.pi$.

In case (a), since the value read by $\texttt{imp}(rl^x_i)$ is even, from the implementation encapsulation (Thm. 1) we know there exists j such that $\texttt{imp}(a) = c_x$ or $\texttt{imp}(a) = \theta^{rl(x)}_j.rl$ or $\texttt{imp}(a) = \theta^{ru(x)}_j.ru$ or $\texttt{imp}(a) = \theta^{wu(x)}_j.wu$. As such, we know that either $a \in C^x \cup \mathcal{RL}^x \cup \mathcal{RU}^x \cup \mathcal{WU}^x$, as required.

Similarly in case (b), since the value read by $\texttt{imp}(wl^x_i)$ is zero, from the implementation encapsulation (Thm. 1) we know there exists j such that $\texttt{imp}(a) = c_x$ or $\texttt{imp}(a) = \theta^{ru(x)}_j.ru$ or $\texttt{imp}(a) = \theta^{wu(x)}_j.wu$. As such, we know that either $a \in C^x \cup \mathcal{RU}^x \cup \mathcal{WU}^x$, as required.

In case (c) we then know that the value read by $\theta^{pl(x)}_i.pi$ is 2; and thus from the implementation encapsulation (Thm. 1) we know e is either a read unlock event or a read lock event. That is, there exists j such that $e = \theta^{ru(x)}_j.ru$ or $e = \theta^{rl(x)}_j.rl$, as required.

In case (d), since the value read by $\texttt{imp}(pl^x_i)$ is 3, from the implementation encapsulation (Thm. 1) we know there exists j such that $e = \theta^{ru(x)}_j.ru$, as required.

For part (3) we proceed by contradiction. Let us assume there exists e, e_1, e_2 such that $(e, e_1), (e, e_2) \in \text{com}$. We then know there exists i such that either a) $e_1 = rl^x_i$; or b) $e_1 = wl^x_i$; or c) $e_1 = pl^x_i$. Similarly, we know there exists j such that either i) $e_1 = rl^x_j$; or ii) $e_2 = wl^x_j$; or iii) $e_2 = pl^x_j$.

In case (a-i), from the definition of com' we know $(\texttt{imp}(e), \texttt{imp}(rl^x_i)), (\texttt{imp}(e), rl^x_j) \in \text{com}$. However, since both rl^x_i and rl^x_j are atomic update operations, from the C11 consistency we know that there exists mo such that $(\texttt{imp}(e), \texttt{imp}(rl^x_i)), (\texttt{imp}(e), \texttt{imp}(rl^x_j)) \in \text{mo}|_{\text{imm}}$, and that either $\texttt{imp}(rl^x_i) \xrightarrow{\text{mo}} \texttt{imp}(rl^x_j)$ or $\texttt{imp}(rl^x_j) \xrightarrow{\text{mo}} \texttt{imp}(rl^x_i)$. In the former case we have $\texttt{imp}(e) \xrightarrow{\text{mo}} \texttt{imp}(rl^x_i) \xrightarrow{\text{mo}} \texttt{imp}(rl^x_j)$ and thus $(\texttt{imp}(e), \texttt{imp}(rl^x_j)) \notin \text{mo}|_{\text{imm}}$, leading to contradiction. Similarly, in the latter case we have $\texttt{imp}(e) \xrightarrow{\text{mo}} \texttt{imp}(rl^x_j) \xrightarrow{\text{mo}} \texttt{imp}(rl^x_i)$ and thus $(\texttt{imp}(e), \texttt{imp}(rl^x_i)) \notin \text{mo}|_{\text{imm}}$, leading to contradiction.

The proof of the remaining cases (a-ii)-(a-iii), (b-i)-(b-iii) and (c-i)-(c-iii) are analogous and are omitted here.

Part (4) follows from the definition of com' and the functionality of com^{-1} for C11 registers. □

```
new-MSRW() ≜
  let x = alloc(K) in x

rlock(x) ≜
  let t = get-tid() in
  loop
    if compare-set(x[t], 0, 2, acq) then
      break₁ ()

runlock(x) ≜
  let t = get-tid() in
  strore(x[t], 0, rel)

wunlock(x) ≜
  for i = 0 to K do
    store(x[i], 0, rel);
```

```
wlock(x) ≜
  for i = 0 to K do
    loop
      if compare-set(x[i], 0, 1, acq) then
        break₁ ()

plock(x) ≜
  let t = get-tid() in
  let c = load(x[t], rlx) in
  if c == 2 then
    if t == 0 then store(x[t], 1, rlx)
    else
      loop
        if compare-set(x[0], 0, 1, acq) then
          store(x[t], 1, rel); break₁ ()
    for i = 1 to K do
      if (i ≠ t) then
        loop
          if compare-set(x[i], 0, 1, acq) then
            break₁ ()
```

Fig. 9. An implementation of weak MRSW locks (for K threads) using a K-array of C11 registers

E.2 A Sound Weak MRSW Lock Implementation

Our second MRSW lock implementation is similarly implemented using C11 registers and is given in Fig. 9. In this implementation, a lock at location x is represented as an *ordered* map of size K at location x. The map at x contains one entry per thread (when there are K threads present) as follows. For each thread with identifier τ, the $x[\tau]$ map entry records the current locking privileges of τ on x. More concretely, when $x[\tau] = 0$, then τ does not hold the x lock; when $x[\tau] = 2$, then τ holds x in read mode; and when $x[\tau] = 1$; then *some* thread (either τ or another thread) either holds x in write mode, or it is in the process of acquiring x in write mode. The x lock is held in write mode only when all entries in x are mapped to one. As we describe shortly, for thread τ to acquire x in write mode, it must inspect each entry in x (in order), wait for it be free (zero) and then set it to one. In our implementation, we assume that the thread identifier can be obtained by calling *get-tid*( ). We identify the top-most thread by $\tau = 0$; as such, the entry of top-most thread in each map is ordered before all other threads.

We proceed with a more detailed explanation of our implementation after introducing our map notation.

Map notation. We write $\mathbf{1}$ to denote a map where all entries have value 1; similarly, we write $\mathbf{0}$ to denote a map where all entries have value 0. Lastly, we write $S \subseteq x$, to denote that the values held in map x are a superset of S. The lock map x associated with location x can be in one of the following states:

- $x = \mathbf{0}$ when x is free;
- $x = \mathbf{1}$ when x is held in write mode;
- $\{2\} \subseteq x$ when x is held in read mode (by those threads τ where $x[\tau] = 2$).

When thread τ calls $rlock(x)$, it simply spins until the lock is free ($x = \mathbf{0}$ and thus $x[\tau] = 0$), at which point it acquires it in read mode by setting $x[\tau]$ to two. Dually, when τ calls $runlock(x)$ it simply sets $x[\tau]$ to zero.

Analogously, when τ calls $wlock(x)$, it traverses the x map in order, spinning on each entry until it is free (0) and subsequently acquiring it (by setting it to 1). Conversely, when τ calls $wunlock(x)$, it releases x by traversing x in order and setting each entry to one.

Recall that in order to promote a reader lock, the calling thread must already hold a reader lock on x. As such, the implementation of $plock(x)$ first check whether the calling thread τ currently holds a reader lock on x, i.e. $x[\tau] = 2$. If this is not the case then the implementation simply returns. To understand the remainder of the implementation, first consider the case where $plock(x)$ is called by $\tau \neq 0$, i.e. a thread other than the top-most thread. The implementation of $plock(x)$ then inspects the *first* entry in the map ($x[0]$), i.e. that of the top-most thread. If $x[0] = 1$, then x is currently being acquired by another thread; the promotion must thus be retried. If on the other hand $x[0] \neq 1$ (i.e. $x[0] = 0$ or $x[0] = 2$), the implementation spins until it is zero and atomically updates it to one, signalling its intention to promote x. This pre-empts the promotion of x by other threads: any such attempt would fail as now $x[0] = 1$. The implementation then sets its own entry ($x[\tau]$) to one, traverses the map in order, and spins on each entry until they too can be set to one. At this point the lock is successfully promoted and the implementation returns. Note that it is safe for τ to update its own entry $x[\tau]$ to one: at this point in execution no thread holds the writer lock on x, no thread can promote its lock on x, and those threads with a reader lock on x never access the $x[\tau]$ entry – the read lock calls of another thread τ' solely accesses $x[\tau']$.

Let us now consider the case when the top-most thread with $\tau = 0$ calls `can-promote x`. Since prior to a $plock(x)$ call τ owns a reader lock on x, i.e. $x[\tau] = 2$, no other thread can promote its x lock. As such, τ successfully sets $x[\tau]$ to one, signalling its intention to promote x. In other words, the promotion is skewed in favour of the top-most thread: if a thread races against the top-most thread to promote x, the top-most thread always wins. With the exception of the top-most thread, promotion is done on a 'first-come-first-served' basis. The rest of the implementation is then carried out as before: the map x is traversed in turn and each entry is set to one.

Implementation Correctness. Let I denote the weak MRSW lock implementation in Fig. 9. To show the soundness of I, we appeal to Thm. 1 and show that I is locally sound on L^{SRW}.

Pick an arbitrary $\Lambda, f, G{=}\langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}\rangle, E'', \mathsf{po}''$ such that G is Λ-consistent and Λ-well-formed and $\mathsf{abs}_{L^{\mathsf{RW}}, I}(f, \langle E, \mathsf{po}\rangle, \langle E'', \mathsf{po}''\rangle)$.
We must next find $\mathsf{com}'', \mathsf{so}''$ such that $\langle E'', \mathsf{po}'', \mathsf{com}'', \mathsf{so}'', \mathsf{lhb}''\rangle \in L^{\mathsf{RW}}.\mathcal{G}_{\mathsf{c}}$, where lhb'' is the same as lhb' in Def. 11.

For each location x, without loss of generality let us assume G contains n_x read lock calls on x, n'_x read unlock calls on x, m_x write lock calls on x, m'_x write unlock calls on x and p_x lock promotion calls on x. Let us enumerate each of read lock calls, read unlock calls, write lock calls, write unlock calls and lock promotion calls arbitrarily. Note that:

- the MRSW constructor at location x contains a single event c_x where $\mathtt{lab}(c_x) = \mathtt{alloc}(x, 0)$.
- For each ith read lock operation on x, the G contains the trace $\theta_i^{rl(x)} = t \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} rf^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} rl$, where rf^* denotes the events of those iterations that failed to acquire the reader lock (failed CAS), $\mathtt{lab}(t) = \mathtt{get\text{-}tid}(\tau)$ for some τ, $\mathtt{lab}(rl) = \mathtt{compare\text{-}set}(\mathtt{acqrel}, x[\tau], 0, 2)$.
- for each ith read unlock operation on x, the G contains the trace $\theta_i^{ru(x)} = t \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} ru$, where $\mathtt{lab}(t) = \mathtt{get\text{-}tid}(\tau)$ for some τ and $\mathtt{lab}(ru) = \mathtt{store}(\mathtt{rel}, x[\tau], 0)$.

- for each ith write lock operation on x, the G contains the trace $\theta_i^{wl(x)} = wf_0^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} wl_0 \xrightarrow{\mathsf{po}} \cdots \xrightarrow{\mathsf{po}} wf_K^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} wl_K$, where for each $j \in \{0 \cdots K\}$, the wf_j^* denotes the events for which the CAS on $x[j]$ failed, and $\texttt{lab}(wl_j) = \texttt{compare-set}(\texttt{acq}, x[j], 0, 1)$.
- for each ith write unlock operation on x, the G contains the trace $\theta_i^{wu(x)} = wu_0 \xrightarrow{\mathsf{po}} \cdots \xrightarrow{\mathsf{po}} wu_K$, where for each $j \in \{0 \cdots K\}$, $\texttt{lab}(wu_j) = \texttt{store}(\texttt{rel}, x[j], 0)$.
- for each ith read lock promotion operation on x the G contains either the trace $\theta_i^{bpl(x)}$; or the trace $\theta_i^{hpl(x)}$; or the trace $\theta_i^{pl(x)}$.

  The $\theta_i^{bpl(x)}$ trace is of the form $t \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} cp$ with $\texttt{lab}(t) = \texttt{get-tid}(\tau)$ for some τ, and $\texttt{lab}()cp = \texttt{load}(\texttt{rlx}, x[\tau], v)$ for some $v \neq 2$.

  The $\theta_i^{hpl(x)}$ is of the form: $t \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} cp \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pi \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pf_1^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pl_1 \xrightarrow{\mathsf{po}} \cdots \xrightarrow{\mathsf{po}} pf_K^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pl_K$, where $\texttt{lab}(t) = \texttt{get-tid}(0)$ and $\texttt{lab}(pi) = \texttt{store}(\texttt{rlx}, x[0], 1)$.

  .

  The $\theta_i^{pl(x)}$ is of the form: $t \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} cp \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pif^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pi \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pf_1^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pl_1 \xrightarrow{\mathsf{po}} \cdots \xrightarrow{\mathsf{po}} pf_{\tau-1}^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pl_{\tau-1} \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pf_{\tau+1}^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pl_{\tau+1} \xrightarrow{\mathsf{po}} \cdots \xrightarrow{\mathsf{po}} pf_K^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} pl_K$, where $\texttt{lab}(t) = \texttt{get-tid}(\tau)$ and $\tau \neq 0$. The pif^* denotes the sequence of events failing to indicate lock promotion by setting $x[0]$ to 1; and $\texttt{lab}(pi) = \texttt{compare-set}(\texttt{acq}, x[0], 0, 1)$.
  In both cases, $\texttt{lab}()cp = \texttt{load}(\texttt{rlx}, x[\tau], 2)$ and for $j \in \{1, \cdots, K\}$, the pf_j^* denotes the events for which the CAS on $x[j]$ failed, and $\texttt{lab}(pl_j) = \texttt{compare-set}(\texttt{acq}, x[j], 0, 1)$.

Let us define: $\mathsf{so}'' = \mathsf{com}'$, with com'' defined as follows:

$$
\begin{aligned}
\mathsf{com}'' \triangleq\ & \left\{ (f_a(a), f_a(b)) \;\middle|\; (a,b) \in \mathsf{com} \wedge \exists i,j,x,\tau.\ b{=}\theta_i^{wl(x)}.wl_\tau \wedge a{=}\theta_j^{ru(x)}.ru \right\} \\
\cup & \left\{ (f_a(a), f_a(b)) \;\middle|\; \begin{array}{l} (a,b) \in \mathsf{com} \wedge \exists i,j,x. \\ \quad b{=}\theta_i^{wl(x)}.wl_0 \wedge a{=}\theta_j^{wu(x)}.wu_0 \\ \quad \wedge \forall k,c.\ 0 \leq k \leq K \wedge (c, \theta_i^{wl(x)}.wl_k) \in \mathsf{com} \Rightarrow \neg\exists h.\ c = \theta_h^{ru(x)}.ru \end{array} \right\} \\
\cup & \left\{ (f_a(a), f_a(b)) \;\middle|\; \begin{array}{l} (a,b) \in \mathsf{com} \wedge \exists i,j,x,\tau. \quad b{=}\theta_i^{wl(x)}.wl_0 \wedge a{=}c_x \\ \quad \wedge \forall k,c.\ 0 \leq k \leq K \wedge (c, \theta_i^{wl(x)}.wl_k) \in \mathsf{com} \Rightarrow \neg\exists h.\ c = \theta_h^{ru(x)}.ru \end{array} \right\} \\
\cup & \left\{ (f_a(a), f_a(b)) \;\middle|\; \begin{array}{l} (a,b) \in \mathsf{com} \wedge \exists i,j,x,\tau. \\ \quad b{=}\theta_i^{rl(x)}.rl \wedge (a = c_x \vee a{=}\theta_j^{wu(x)}.wu_\tau) \end{array} \right\} \\
\cup & \left\{ (f_a(a), f_a(b)) \;\middle|\; \begin{array}{l} \exists c,i,j,k,x,\tau. \\ \quad (c,b) \in \mathsf{com} \wedge b{=}\theta_i^{rl(x)}.rl \wedge c{=}\theta_j^{ru(x)}.ru \\ \quad \wedge (a = c_x \vee a{=}\theta_k^{wu(x)}.wu_\tau) \wedge (a,c) \in \mathsf{mo} \\ \quad \wedge \forall d.\ a \xrightarrow{\mathsf{mo}} d \xrightarrow{\mathsf{mo}} c \Rightarrow \\ \quad \neg\exists h.\ (d{=}\theta_h^{wl(x)}.wl_\tau \vee d{=}\theta_h^{wu(x)}.wu_\tau \vee d{=}\theta_h^{pl(x)}.pl_\tau \vee d{=}\theta_h^{hpl(x)}.pl_\tau) \end{array} \right\} \\
\cup & \left\{ (f_a(a), f_a(b)) \;\middle|\; (a,b) \in \mathsf{com} \wedge \exists i,j,x,\tau.\ (b{=}\theta_i^{pl(x)}.pl_\tau \vee b{=}\theta_i^{hpl(x)}.pl_\tau) \wedge a{=}\theta_j^{ru(x)}.ru \right\} \\
\cup & \left\{ (f_a(a), f_a(b)) \;\middle|\; \begin{array}{l} (a,b) \in \mathsf{com} \wedge \exists i,j,x. \\ \quad b{=}\theta_i^{pl(x)}.pl_0 \wedge a{=}\theta_j^{wu(x)}.wu_0 \\ \quad \wedge \forall k,c.\ 0 \leq k \leq K \wedge (c, \theta_i^{pl(x)}.pl_k) \in \mathsf{com} \Rightarrow \neg\exists h.\ c = \theta_h^{ru(x)}.ru \end{array} \right\}
\end{aligned}
$$

$$\cup \left\{ (f_a(a), f_a(b)) \;\middle|\; \begin{array}{l} (a,b) \in \textsf{com} \land \exists i, j, x. \\ \quad b{=}\theta_i^{hpl(x)}.pl_0 \land a{=}\theta_j^{wu(x)}.wu_0 \\ \quad \land \forall k, c.\ 0 \le k \le K \land (c, \theta_i^{hpl(x)}.pl_k) \in \textsf{com} \Rightarrow \neg\exists h.\ c = \theta_h^{ru(x)}.ru \end{array} \right\}$$

$$\cup \left\{ (f_a(a), f_a(b)) \;\middle|\; \begin{array}{l} (a,b) \in \textsf{com} \land \exists i, j, x, \tau.\ b{=}\theta_i^{pl(x)}.pl_0 \land a{=}c_x \\ \quad \land \forall k, c.\ 0 \le k \le K \land (c, \theta_i^{pl(x)}.pl_k) \in \textsf{com} \Rightarrow \neg\exists h.\ c = \theta_h^{ru(x)}.ru \end{array} \right\}$$

$$\cup \left\{ (f_a(a), f_a(b)) \;\middle|\; \begin{array}{l} (a,b) \in \textsf{com} \land \exists i, j, x, \tau.\ b{=}\theta_i^{hpl(x)}.pl_0 \land a{=}c_x \\ \quad \land \forall k, c.\ 0 \le k \le K \land (c, \theta_i^{hpl(x)}.pl_k) \in \textsf{com} \Rightarrow \neg\exists h.\ c = \theta_h^{ru(x)}.ru \end{array} \right\}$$

To show that $G' = \langle E'', \textsf{po}'', \textsf{com}'', \textsf{so}'', \textsf{lhb}'' \rangle \in L^{\text{RW}}.\mathcal{G}_c$, we are then required to show for all $x \in \text{Loc}$, G'_x is RW-consistent on x. Pick an arbitrary $x \in \text{Loc}$ and let $G' = \langle E', \textsf{po}', \textsf{com}', \textsf{so}', \textsf{lhb}' \rangle$. We then need to show:

(1) G'_x contains at most one constructor event;
(2) $\textsf{com}' = \textsf{com}_w \cup \textsf{com}_r \cup \textsf{com}_p$ with:
$\textsf{com}_w \subseteq (C^x \cup \mathcal{U}^x) \times \mathcal{WL}^x \quad \textsf{com}_r \subseteq (C^x \cup \mathcal{WU}^x) \times \mathcal{RL}^x \quad \textsf{com}_p \subseteq (C^x \cup \mathcal{U}^x) \times \mathcal{PL}^x$
(3) for all e, e_1, e_2:
$e_1 {\neq} e_2 \land (e, e_1), (e, e_2) {\in} \textsf{com}' \Rightarrow (e_1, e_2 {\in} \mathcal{RL}^x) \lor (e_1 {\in} \mathcal{RL}^x \land e_2 {\in} \mathcal{PL}^x) \lor (e_1 {\in} \mathcal{PL}^x \land e_2 {\in} \mathcal{RL}^x)$
(4) $\textsf{com}'^{-1}$ is functional;
(5) $E \cap \mathcal{L}^x = rng(\textsf{com}')$; and
(6) $\textsf{so}' = \textsf{com}'$.

Parts (1), (5) and (6) follow immediately from the construction of G'_x. For part (2), pick an arbitrary $(a, b) \in \textsf{com}'$. From the definition of $\textsf{com}'$ we then know that there exists i, j such that either:
i) $b = wl_i^x$, $a = wu_j^x \lor a = ru_j^x \lor a{=}con_x$; or
ii) $b = rl_i^x$, $a = wu_j^x \lor a = con_x$; or
iii) $b = pl_i^x$, $a = wu_j^x \lor a = ru_j^x \lor a{=}con_x$
In case (i), we thus have $(a, b) \in (C^x \cup \mathcal{U}^x) \times \mathcal{WL}^x$, as required. In case (ii) we have $(a, b) \in (C^x \cup \mathcal{WU}^x) \times \mathcal{RL}^x$, as required. In case (iii), we have $(a, b) \in (C^x \cup \mathcal{U}^x) \times \mathcal{WL}^x$, as required.

For part (3) we proceed by contradiction. Let us assume there exists e, e_1, e_2 such that $e_1 \neq e_2$, $(e, e_1), (e, e_2) \in \textsf{com}$ and either: i) $e_1, e_2 \in \mathcal{WL}^x$; or ii) $e_1 \in \mathcal{WL}^x$ and $e_2 \in \mathcal{RL}^x$; or iii) $e_1 \in \mathcal{WL}^x$ and $e_2 \in \mathcal{PL}^x$; or iv) $e_1, e_2 \in \mathcal{PL}^x$.

In case (i), we know there exits a, b_1, b_2 such that $e = f_a(a), e_1 = f_a(b_1), e_2 = f_a(b_2), (a, b_1), (a, b_2) \in \textsf{com}$ and b_1 and b_2 are both atomic CAS operations. As such, from the C11 consistency we know $(a, b_1), (a, b_2) \in \textsf{mo}|_{\text{imm}}$. Moreover, from C11 consistency we have either $(b_1, b_2) \in \textsf{mo}$ or $(b_2, b_1) \in \textsf{mo}$. In the former case we then have $a \xrightarrow{\textsf{mo}} b_1 \xrightarrow{\textsf{mo}} b_2$ and thus $(a, b_2) \notin \textsf{mo}|_{\text{imm}}$, leading to a contradiction. Similarly, in the latter case we then have $a \xrightarrow{\textsf{mo}} b_2 \xrightarrow{\textsf{mo}} b_1$ and thus $(a, b_1) \notin \textsf{mo}|_{\text{imm}}$, leading to a contradiction.

The proof of cases (ii) and (iii) are analogous to that of (i) and are omitted here.

In case (ii) we then know there exists a, b_1, b_2, c such that $e = f_a(a)$, $e_1 = f_a(b_1)$, $e_2 = f_a(b_2)$, b_1 and b_2 are both atomic CAS operations, $(a, b_1) \in \textsf{com}$, and either a) $(a, b_2) \in \textsf{com}$; b) $(a, c) \in \textsf{mo}$, $(c, b_2) \in \textsf{com}$ and $\neg(a \xrightarrow{\textsf{mo}} b_1 \xrightarrow{\textsf{mo}} c)$. The proof of case (a) is analogous to the proof of case (1) above. For part (b), as b_2 is an atomic CAS from the C11 consistency we have $(c, b) \in \textsf{mo}|_{\text{imm}}$. Moreover, from C11 consistency we have either $(b_1, c) \in \textsf{mo}$ or $(c, b_1) \in \textsf{mo}$. However, as we have $\neg(a \xrightarrow{\textsf{mo}} b_1 \xrightarrow{\textsf{mo}} c)$ and $(a, b_1) \in \textsf{mo}$, we then have $(c, b_1) \in \textsf{mo}$. As such, we have $a \xrightarrow{\textsf{mo}} c \xrightarrow{\textsf{mo}} b_1$ and thus $(a, b_1) \notin \textsf{mo}|_{\text{imm}}$, leading to a contradiction.

Part (4) follows from the definition of $\textsf{com}'$ and the functionality of $\textsf{com}^{-1}$ for C11 registers. □

$$
\begin{array}{ll}
\begin{array}{l}
\textit{new-queue}() \triangleq \\
\quad \texttt{let } l = \textit{new-mutex}() \texttt{ in} \\
\quad \texttt{let } q = \textit{alloc}(+\infty) \texttt{ in} \\
\quad \textit{store}(q, l, \texttt{rlx}); \textit{store}(q{+}1, 2, \texttt{rlx}); q \\
\textit{enq}(q, v) \triangleq \\
\quad \texttt{let } l = \textit{load}(q, \texttt{rlx}) \texttt{ in} \\
\quad \textit{lock}(l); \texttt{let } i = \textit{load}(q{+}1, \texttt{rlx}) \texttt{ in} \\
\quad \textit{store}(q{+}i, v, \texttt{rlx}); \textit{store}(q{+}1, i{+}1, \texttt{rlx}); \\
\quad \textit{unlock}(l)
\end{array}
&
\begin{array}{l}
\textit{deq}(q) \triangleq \\
\quad \texttt{let } l = \textit{load}(q, \texttt{rlx}) \texttt{ in} \\
\quad \textit{lock}(l); \texttt{let } \textit{range} = \textit{load}(q{+}1, \texttt{rlx}) \texttt{ in} \\
\quad \texttt{for } i = 1 \texttt{ to } \textit{range} \texttt{ do} \\
\quad\quad \texttt{let } x = \textit{load}(q + i, \texttt{rlx}) \texttt{ in} \\
\quad\quad \texttt{if } x \neq 0 \texttt{ then} \\
\quad\quad\quad \textit{unlock}(l); \texttt{break}_1\ x \\
\quad \textit{unlock}(l); \bot
\end{array}
\end{array}
$$

Fig. 10. The locking queue implementation

F A SOUND STRONG QUEUE IMPLEMENTATION

In Fig. 10 we present a simple implementation of a strong queue using release-acquire registers and the mutex library. As we formalise in Thm. 9, this implementation is sound with respect to the strong queue library L^{SQ}.

Theorem 9. *The queue implementation in Fig. 7 is a sound implementation of the strong queue library L^{SQ}.*

Proof. The full proof is mechanised in the Coq proof assistant and is available as auxiliary material. □

G THE SOUNDNESS OF EXCHANGER IMPLEMENTATION

Let I denote the exchanger implementation in Fig. 5. To show the soundness of I, we appeal to Thm. 1 and show that I is locally sound on L^{X}.

Pick an arbitrary Λ, f, $G{=}\langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}\rangle$, E', po' such that G is Λ-consistent and Λ-well-formed and $\mathsf{abs}_{L^{\mathsf{X}},I}(f, \langle E, \mathsf{po}\rangle, \langle E', \mathsf{po}'\rangle)$. We must next find com', so' such that $\langle E', \mathsf{po}', \mathsf{com}', \mathsf{so}', \mathsf{lhb}'\rangle \in L^{\mathsf{X}}.\mathcal{G}_{\mathsf{c}}$, where lhb' is as defined in Def. 11.

Note that each exchange operation $\textit{exchange}(g, v)$ either:

1. offers a value at index $g{+}k$ (for some $k \in \mathbb{N}^{+}$ where k is an odd number) and returns unmatched due to a timeout; or
2. offers a value at index $g{+}k$ (for some $k \in \mathbb{N}^{+}$ where k is an odd number) and matches with value v' at index $g{+}k{+}1$; or
3. tries to offer a value at index $g{+}k$ (for some $k \in \mathbb{N}^{+}$ where i is an even number) and returns unmatched as the slot at index $g{+}k$ is already taken; or
4. offers a value at index $g{+}k$ (for some $k \in \mathbb{N}^{+}$ where k is an even number) and matches with value v' at index $g{+}k{-}1$.

Without loss of generality, let us assume e contains n exchange calls. For each i^{th} exchange operation of the form $\textit{exchange}(g, v_i)$, the G_i contains a trace of one of the following forms depending which of the four categories above it falls into:

- $\theta_i^{t(g)}$ is of the form $s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} o \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} t$, with $\mathsf{lab}(s) = \mathtt{load}(\mathtt{rlx}, g, j_i)$, $\mathsf{lab}(o) = \mathtt{CAS}(\mathtt{rel}, g{+}j_i, 0, v_i)$, and $\mathsf{lab}(t) = \mathtt{CAS}(\mathtt{rlx}, g{+}j_i{+}1, 0, \bot)$, for some j_i and v_i where j_i is odd;
- $\theta_i^{o(g)}$ is of the form $s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} o \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} a \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r$, with $\mathsf{lab}(s) = \mathtt{load}(\mathtt{rlx}, g, j_i)$, $\mathsf{lab}(o) = \mathtt{CAS}(\mathtt{rel}, g{+}j_i, 0, v_i)$, $\mathsf{lab}(a) = \mathtt{load}(\mathtt{rlx}, g{+}j_i{+}1, v_i')$ with $v_i' \neq 0$, $\mathsf{lab}(r) = \mathtt{load}(\mathtt{acq}, g{+}j_i{+}1, v_i')$, for some j_i, v_i and v_i' where j_i is odd;

- $\theta_i^{f(g)}$ is of the form $s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} f \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} o \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} c$, with $\mathtt{lab}(s) = \mathtt{load}(\mathtt{rlx}, g, j_i)$, $\mathtt{lab}(f) = \mathtt{load}(\mathtt{rlx}, g{+}j_i, w_i)$ with $w_i \neq 0$, $\mathtt{lab}(o) = \mathtt{load}(\mathtt{rlx}, g{+}j_i{+}1, w'_i)$ with $w'_i \neq 0$, and $\mathtt{lab}(c) = \mathtt{CAS}(\mathtt{rlx}, g, j_i, j_i{+}2)$ or $\mathtt{lab}(c_i) = \mathtt{load}(\mathtt{rlx}, g, u_i)$ with $u_i \neq j_i$, for some j_i and v_i where j_i is odd;
- $\theta_i^{e(g)}$ is of the form $s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} f \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} o \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} c \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r$, with $\mathtt{lab}(s) = \mathtt{load}(\mathtt{rlx}, g, j_i)$, $\mathtt{lab}(f) = \mathtt{load}(\mathtt{rlx}, g{+}j_i, w_i)$ with $w_i \neq 0$, $\mathtt{lab}(o) = \mathtt{CAS}(\mathtt{rel}, g{+}j_i{+}1, 0, v_i)$, $\mathtt{lab}(c) = \mathtt{CAS}(\mathtt{rlx}, g, j_i, j_i{+}2)$ or $\mathtt{lab}(c) = \mathtt{load}(\mathtt{rlx}, g, u_i)$ with $u_i \neq j_i$, $\mathtt{lab}(r) = \mathtt{load}(\mathtt{acq}, g{+}j_i, v'_i)$, for some j_i, v_i and v'_i where j_i is odd.

Let $\mathsf{so}' = \mathsf{com}'$ with

$$\mathsf{com}' \triangleq \left\{ (a,b),(b,a) \;\middle|\; \begin{array}{l} \exists i,j,g.\ f(e_i^{o(g)}.o){=}a \wedge f(e_j^{e(g)}.o){=}b \\ \wedge (\theta_i^{o(g)}.o, \theta_j^{e(g)}.r) \in \mathsf{com} \wedge (\theta_j^{e(g)}.o, \theta_i^{o(g)}.r) \in \mathsf{com} \end{array} \right\}$$

To show that $G' = \langle E', \mathsf{po}', \mathsf{com}', \mathsf{so}', \mathsf{lhb}' \rangle \in L^{\mathsf{X}}.\mathcal{G}_{\mathsf{c}}$, we are then required to show for all $g \in \mathsf{Loc}$, G'_g is exchanger-consistent on g. Pick an arbitrary $g \in \mathsf{Loc}$, we then need to show:

1) $E'^c = \emptyset \vee \exists c \in C^g.\ E'^c = \{c\}$;
2) com' is *symmetric*, *irreflexive* and $\mathsf{com}' \subseteq \bigcup_{v_1,v_2 \in \mathsf{Val}} \mathcal{X}^{g,v_1,v_2} \times \mathcal{X}^{g,v_2,v_1} \setminus \mathrm{id}$;
3) com' is functional;
4) $E' \cap \mathcal{X}^g \setminus \mathit{dom}(\mathsf{com}') \subseteq \bigcup_{v \in \mathsf{Val}} \mathcal{X}^{g,v,\perp}$; and
5) $\mathsf{so}' = \mathsf{com}'$.

Parts (1), (2), and (5) follow immediately from the construction of G_s and the consistency of the C library. The proof of parts (3) and (4) follows from the definition of com', the consistency of G_i and the definition of the C library.

H CORRECTNESS OF THE HERLIHY-WING QUEUE IMPLEMENTATIONS

H.1 Soundness of the Strong Herlihy-Wing Queue Implementation

Let I denote the strong Herlihy-Wing implementation in Fig. 2. To show the soundness of I, we appeal to Thm. 1 and show that I is locally sound on L^{SQ}.

Pick an arbitrary $\Lambda, f, G{=}\langle E, \text{po}, \text{com}, \text{so}\rangle, E'', \text{po}''$ such that G is Λ-consistent and Λ-well-formed and $\text{abs}_{L^{\text{SQ}}, I}(f, \langle E, \text{po}\rangle, \langle E'', \text{po}''\rangle)$. We must next find $\text{com}'', \text{so}''$ such that $\langle E'', \text{po}'', \text{com}'', \text{so}'', \text{lhb}''\rangle \in L^{\text{SQ}}.\mathcal{G}_c$, where lhb'' is the same as lhb' in Def. 11.

For each queue at q, let us enumerate the enqueue operations on q by their insertion index. For instance, the very first enqueue operation is that which inserts the new value at index $q{+}1$. That is, the ith enqueue operation is that which inserts its value at index $q{+}i$. Similarly, let us enumerate the dequeue operations by their removal index. That is, the jth dequeue operation is that which removes the element at index $q{+}j$. When program $\mathtt{e}$ contains n_q enqueue operations on q and m_q dequeue operations on q then:

- the constructor of the queue at location q contains a trace with a single event c_q where $\mathtt{lab}(c_q) = \mathtt{alloc}(q, 0)$;
- for each ith enqueue operation $enq(q, v_i)$ with $v_i \neq \bot$, G contains a trace of the form $\theta_i^{e(q)} = e_i^1 \xrightarrow{\text{po}|_{\text{imm}}} e_i^2$, where $\mathtt{lab}(e_i^1){=}\mathtt{FAA}(\mathtt{rel}, q, i{-}1, 1)$ and $\mathtt{lab}(e_i^2){=}\mathtt{store}(\mathtt{rel}, q{+}i, v_i)$;
- for each jth dequeue operation, G contains a trace of the form $\theta_j^d = \theta_j^f \xrightarrow{\text{po}} d_i^1 \xrightarrow{\text{po}} f_i^1 \xrightarrow{\text{po}} \cdots \xrightarrow{\text{po}} f_i^{j-1} \xrightarrow{\text{po}|_{\text{imm}}} d_i^2$, such that all events of θ_j^f are read events: $\forall a \in \theta_j^f.\ \mathtt{lab}(a) = \mathtt{load}(-,-,-)-$; $\mathtt{lab}(d_1){=}\mathtt{load}(\mathtt{acq}, q, len_j)$; $\mathtt{lab}(f_k) = \mathtt{load}(\mathtt{acqrel}, q{+}k, 0)$ for all $k \in \{1 \cdots j{-}1\}$; and $\mathtt{lab}(d_2){=}\mathtt{AX}(\mathtt{acqrel}, q{+}j, w_j, 0)$, for some $w_j \neq \bot$, range $len_j \in \mathbb{N}^+$ and $1 \leq j \leq len_j$.

Let us define: $\mathtt{imp}(.) : E'' \longrightarrow E$ as follows:

$$\mathtt{imp}(e) \triangleq \begin{cases} c_q & \exists q.\ f(c_q) = e \\ \theta_i^{e(q)}.e_i^2 & \exists i, q.\ f(\theta_i^{e(q)}.e_i^2) = e \\ \theta_i^{d(q)}.d_i^2 & \exists i, q.\ f(\theta_i^{d(q)}.d_i^2) = e \end{cases}$$

Let $\text{so}'' = \text{com}''$ with

$$\text{com}'' \triangleq \left\{(e, d) \,\middle|\, \exists i, q.\ f(\theta_i^{e(q)}.e_i^2) = e \wedge f(\theta_i^{d(q)}.d_i^2) = d \wedge (\theta_i^{e(q)}.e_i^2, \theta_i^{d(q)}.d_i^2) \in \text{com}\right\}$$

Let $\text{hb} = G.\text{hb} = (\text{po} \cup \text{so})^+$. Note that from the definition of lhb'', hb and the construction of so' above we know the (6) below holds. As such, from the irreflexivity of hb, we also know (7) holds.

$$\forall a, b.\ (a, b) \in \text{lhb}'' \Leftrightarrow a \neq b \wedge (\mathtt{imp}(a), \mathtt{imp}(b)) \in \text{hb} \tag{6}$$

$$\forall a.\ (a, a) \notin \text{lhb}'' \tag{7}$$

Moreover, it is straightforward to demonstrate that given the C11 memory model and the values written, the (8) property below holds. Consequently, since the "range" value read by each dequeue operation is greater than or equal the slot acquired by its matching enqueuing thread, the $\mathtt{rel}$ mode of fetch-and-add operations in enqueue and the $\mathtt{acq}$ mode of "range" reads in dequeue operations, thanks to the release sequences of C11 the (9) property below holds.

$$\forall a, b \in \{1 \cdots n\}.\ a < b \Rightarrow (e_a^1, e_b^1) \in \text{mo} \tag{8}$$

$$\forall \pi.\ (e_\pi^2, d_\pi^2) \in \text{com} \Rightarrow (e_\pi^1, d_\pi^1) \in \text{so} \tag{9}$$

Given the encapsulation of G and the definition of I, we know that for all $\pi \in \mathbb{N}$ and all queue locations q, $G.E \cap \left\{e \,\middle|\, \mathtt{loc}(e) = q{+}\pi\right\} = W_\pi$, where $W_\pi \triangleq E_e \cap (\{e_\pi^2\} \cup W_\pi^f \cup W_\pi^d)$, with $W_\pi^f \triangleq \left\{f_k^\pi \,\middle|\, 1 \leq k \leq m\right\}$ and $W_\pi^d \triangleq \{d_\pi^2\}$.

Given the definition of the C11 library, we then know that for each $\pi \in \mathbb{N}^+$, the W_π is totally ordered by a strict total order mo. Consequently, given the release-acquire (acqrel) mode of update events in $W^f_\pi \cup W^d_\pi$, it is straightforward to demonstrate that:

$$\forall \pi \in \mathbb{N}.\ \forall w_1, w_2 \in W^f_\pi \cup W^d_\pi. \\ ((w_1, w_2) \in (\text{so} \cap \text{mo}) \cup (\text{so}^{-1} \cap \text{mo}^{-1})) \lor (w_1, e^2_\pi), (e^2_\pi, w_2) \in \text{mo} \lor (w_2, e^2_\pi), (e^2_\pi, w_1) \in \text{mo} \tag{10}$$

We next demonstrate that:

$$\forall a, b.\ a < b \Rightarrow (e^2_b, e^2_a) \notin \text{hb} \tag{11}$$

$$\forall \pi, \pi' \in \mathbb{N}^+.\ d^2_\pi \xrightarrow{\text{com}^{-1}} e^2_\pi \xrightarrow{\text{hb}} e^2_{\pi'} \xrightarrow{\text{com}} d^2_{\pi'} \Rightarrow d^2_\pi \xrightarrow{\text{hb}} d^2_{\pi'} \tag{12}$$

For part (11) we proceed by contradiction. Let us assume there exists a, b such that $a < b$ and $(e^2_b, e^2_a) \in \text{hb}$. Since G is consistent and e^2_b, e^2_a are write events, we know that they do not have any incoming so edges. As such, as $\text{hb} = (\text{po} \cup \text{so})^+$, we know that there exists e such that $e^2_b \xrightarrow{\text{hb}} e \xrightarrow{\text{po}} e^2_a$. Moreover, since $e^1_a \xrightarrow{\text{po}|_{\text{imm}}} e^2_a$, we also know that $e \xrightarrow{\text{po}} e^1_a$. That is, we have $(e^2_b, e^1_a) \in \text{hb}$. On the other hand, from (8) we know that $(e^1_a, e^1_b) \in \text{mo}$. We then have: $e^1_a \xrightarrow{\text{mo}} e^1_b \xrightarrow{\text{po}} e^2_b \xrightarrow{\text{hb}} e^1_a$. That is, we have $e^1_a \xrightarrow{\text{mo}} e^1_b \xrightarrow{\text{hb}} e^1_a$, contradicting the assumption that G is consistent.

For part (12), as $e^2_\pi \xrightarrow{\text{hb}} e^2_{\pi'}$, from (11) we know that $\pi < \pi'$. Moreover, as $(e^2_\pi, d^2_\pi) \in \text{com}$, from the consistency of the C library we know that $(e^2_\pi, d^2_\pi) \in \text{mo}|_{\text{imm}}$. As such, from (10) we know that either i) $(d^2_\pi, f^\pi_{\pi'}) \in \text{so}$; or ii) $(f^\pi_{\pi'}, e^2_\pi) \in \text{mo}$.

In case (i) we have $d^2_\pi \xrightarrow{\text{so}} f^\pi_{\pi'} \xrightarrow{\text{po}} d^2_{\pi'}$; i.e. $d^2_\pi \xrightarrow{\text{hb}} d^2_{\pi'}$, as required.

In case (ii),Since G is consistent and $e^2_\pi, e^2_{\pi'}$ are write events, we know that they do not have any incoming so edges. As such, as $\text{hb} = (\text{po} \cup \text{so})^+$, we know that there exists e such that $e^2_\pi \xrightarrow{\text{hb}} e \xrightarrow{\text{po}} e^2_{\pi'}$. Moreover, since $e^1_{\pi'} \xrightarrow{\text{po}|_{\text{imm}}} e^2_{\pi'}$, we also know that $e \xrightarrow{\text{po}} e^1_{\pi'}$. That is, we have $(e^2_\pi, e^1_{\pi'}) \in \text{hb}$. Moreover, from (9) we have $(e^1_{\pi'}, d^1_{\pi'}) \in \text{so} \subseteq \text{hb}$. As such, from the assumption of the case we have $e^2_\pi \xrightarrow{\text{hb}} e^1_{\pi'} \xrightarrow{\text{hb}} d^1_{\pi'} \xrightarrow{\text{po}} f^\pi_{\pi'} \xrightarrow{\text{mo}} e^2_\pi$. That is, we have $e^2_\pi \xrightarrow{\text{hb}} f^\pi_{\pi'} \xrightarrow{\text{mo}} e^2_\pi$, contradicting the assumption that G is consistent.

□

To show that $G' = (E'', \text{po}'', \text{com}'', \text{so}'', \text{lhb}'') \in L^{\text{SQ}}.\mathcal{G}_c$, we are then required to show that for all locations q, G'_q is queue consistent on q. Pick an arbitrary location q and let $G'_q = \langle E', \text{po}', \text{com}', \text{so}', \text{lhb}' \rangle$. We then need to show:

1) $E'^c = \emptyset \lor \exists c \in C^q.\ E'^c = \{c\}$;
2) $\text{com}' \subseteq \bigcup_{v \in \text{Val} \setminus \{\bot\}} \mathcal{E}^{q,v} \times \mathcal{D}^{q,v}$
3) $\text{com}', (\text{com}')^{-1}$ are functional
4) $E' \cap \mathcal{D}^q \setminus rng(\text{com}') \subseteq \mathcal{D}^{q,\bot}$
5) $[\mathcal{E}^q \setminus dom(\text{com}')]; \text{lhb}'; [\mathcal{D}^{q,\bot}]{=}\emptyset$
6) $\text{so}' = \text{com}'$
7) there exists a sequential enumeration S of the events in E' such that: (i) S respects lhb'; and (ii) $\text{fifo}(\epsilon, S)$ holds.

Parts (1), (2) and (6) follow simply from the construction of G'. Part (5) holds trivially as $\mathcal{D}^{q,\bot} = \emptyset$.

For part (4), we demonstrate that $E' \cap \mathcal{D}^q \setminus rng(\text{com}') = \emptyset$. We proceed by contradiction. Let us assume there exists $d \in E' \cap \mathcal{D}^q$ such that $d \notin rng(\text{com}')$. From the construction of G' we then know there exists $\theta_j^{d(q)}$ and d_j^2 such that $d_j^2 \in \theta_j^{d(q)}$ and $\mathtt{lab}(d_j^2) = \mathtt{AX}(\mathtt{acq}, q{+}s_j, w_j, 0)$ for some s_j, w_j such that $s_j > 0$ and $w_j \neq \bot$. From the definition of consistency for the C library we know there exists a write event a such that $(a, d_j^2) \in \mathsf{com}$. Given the assumption of the case (A), and the shape of the θ^e and θ^d traces, we then know that there exists $\theta_i^{e(q)}$ and e_i^2 such that $e_i^2 \in \theta_i^{e(q)}$; $a = e_i^2$; and $\mathtt{lab}(e_i^2) = \mathtt{store}(\mathtt{rel}, q{+}s_j, w_j)$. As such, from the construction of G' we know there exists $e \in \mathcal{E}^{q,w_j}$ such that $(e, d) \in \mathsf{com}'$. This however contradicts our assumption that $d \notin \mathit{rng}(\mathsf{com}')$.

For part (3) we proceed by contradiction. Let us assume that $(\mathsf{com}')^{-1}$ is not functional and there exist $e'_1, e'_2 \in \mathcal{E}^q$ and $d' \in \mathcal{D}^q$ such that $e'_1 \neq e'_2$ and $(e'_1, d'), (e'_2, d') \in \mathsf{com}'$. From the construction of G' we then know there exist $\theta_i^{e(q)}, \theta_k^{e(q)}, \theta_j^{d(q)}$ such that $\theta_i^{e(q)} \neq \theta_k^{e(q)}$ $(e_i^2, d_j^2), (e_k^2, d_j^2) \in \mathsf{com}$. That is, $(e_i^2, d_j^2), (e_k^2, d_j^2) \in \mathsf{com}_{L^{\mathrm{C}}}$. This however contradicts the assumption that G is consistent with respect to the C library, i.e. the assumption $\mathsf{com}_{L^{\mathrm{C}}}^{-1}$ is functional.
Let us next assume that com' is not functional and there exist $e' \in \mathcal{E}^q$ and $d'_1, d'_2 \in \mathcal{D}^q$ such that $d'_1 \neq d'_2$ and $(e', d'_1), (e', d'_2) \in \mathsf{com}'_{L^{\mathrm{SQ}}}$. From the construction of G' we then know there exist $\theta_j^{d(q)}, \theta_k^{d(q)}, \theta_i^{e(q)}$ such that $\theta_j^{d(q)} \neq \theta_k^{d(q)}$ $(e_i^2, d_j^2), (e_i^2, d_k^2) \in \mathsf{com}$. That is $(e_i^2, d_j^2), (e_i^2, d_k^2) \in \mathsf{com}_{L^{\mathrm{C}}}$. As d_j^2, d_k^2 are atomic update events, from the definition of the C library and the consistency of G we know they are ordered by a total modification order mo. Without loss of generality, let us assume that $(d_j^2, d_k^2) \in \mathsf{mo}$. From the definition of the C library and the consistency of G, we then also have $(e_i^2, d_j^2) \in \mathsf{mo}$. As such, we have $(d_k^2, d_j^2) \in \mathsf{com}^{-1}; \mathsf{mo}$. Consequently, we have $d_k^2 \xrightarrow{\mathsf{com}^{-1};\mathsf{mo}} d_j^2 \xrightarrow{\mathsf{mo}} d_k^2$, contradicting the assumption that G is consistent.

For part (7), in what follows we demonstrate that for all $n \in \mathbb{N}^+$, the irreflexive$(\mathsf{C}^{n,n})$ holds for G'. The desired property of (7) then follows immediately from Thm. 7.

To demonstrate that $\forall n \in \mathbb{N}^+.\ \mathsf{irreflexive}(\mathsf{C}^{n,n})$ holds, we proceed by induction on n.

Base case $n = 1$

We proceed by contradiction. Let us assume there exist d_1, d_2, e_1, e_2 such that $d_1 \xrightarrow{\mathsf{com}'^{-1}} e_1 \xrightarrow{\mathsf{hb}'} e_2 \xrightarrow{\mathsf{com}'} d_2 \xrightarrow{\mathsf{hb}'} d_1$. From the definition of com', the definition of $\mathtt{imp}(.)$ and (6) we then have $\mathtt{imp}(d_1) \xrightarrow{\mathsf{com}^{-1}} \mathtt{imp}(e_1) \xrightarrow{\mathsf{hb}} \mathtt{imp}(e_2) \xrightarrow{\mathsf{com}} \mathtt{imp}(d_2) \xrightarrow{\mathsf{hb}} \mathtt{imp}(d_1)$. From (12) we then have $\mathtt{imp}(d_1) \xrightarrow{\mathsf{hb}} \mathtt{imp}(d_2) \xrightarrow{\mathsf{hb}} \mathtt{imp}(d_1)$. As such, from the transitivity of hb we have $\mathtt{imp}(d_1) \xrightarrow{\mathsf{hb}} \mathtt{imp}(d_1)$, contradicting the assumption that G is consistent.

Inductive case $n = m{+}1$

$$\forall k \in \mathbb{N}^+.\ k \leq m \Rightarrow \mathsf{irreflexive}(\mathsf{C}^{k,k}) \tag{I.H.}$$

We proceed by contradiction. Let us assume that there exist a $\mathsf{C}^{n,n}$ cycle. As $n > 0$, we know there is at least one adjacent pair of $\mathsf{com}'^{-1}; \mathsf{hb}'$ and $\mathsf{com}'; \mathsf{hb}'$ edges. That is, there exist a, b such that and $a \xrightarrow{\mathsf{com}'^{-1};\mathsf{hb}';\mathsf{com}';\mathsf{hb}'} b \xrightarrow{\mathsf{C}^{m,m}} a$. As such, from (6), (12), the transitivity of hb and the functionality of $\mathtt{imp}(.)$ we have $\mathtt{imp}(a) \xrightarrow{\mathsf{hb}} \mathtt{imp}(b)$ and $a \neq b$. Consequently, from (6) we have $(a, b) \in \mathsf{hb}'$. Moreover, since by definition the $\mathsf{C}^{m,m}$ edge ends with hb' and $a \xrightarrow{\mathsf{hb}'} b$, from the transitivity of hb' and since $b \xrightarrow{\mathsf{C}^{m,m}} a$ we have $b \xrightarrow{\mathsf{C}^{m,m}} b$. This however contradicts our assumption in (I.H.).

H.2 Soundness of the Weak Herlihy-Wing Queue Implementation

Let I denote the weak Herlihy-Wing implementation in Fig. 2. To show the soundness of I, we appeal to Thm. 1 and show that I is locally sound on L^{Q}.

Pick an arbitrary $\Lambda, f, G{=}\langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}\rangle, E'', \mathsf{po}''$ such that G is Λ-consistent and Λ-well-formed and $\mathsf{abs}_{L^{\mathrm{Q}}, I}(f, \langle E, \mathsf{po}\rangle, \langle E'', \mathsf{po}''\rangle)$.
We must next find com'', so'' such that $\langle E'', \mathsf{po}'', \mathsf{com}'', \mathsf{so}'', \mathsf{lhb}''\rangle \in L^{\mathrm{SQ}}.\mathcal{G}_{\mathrm{c}}$, where lhb'' is the same as lhb' in Def. 11.

In the remainder of this proof, we assume all identically-named definitions (e.g. G', com', so' and $\mathtt{imp}(.)$ and so forth) are as defined for the strong Herlihy-Wing queue implementation unless otherwise stated.

Note that due to the encapsulation of G and the definition of I, we know that for all $\pi \in \mathbb{N}$:

$$G.E \cap \left\{ e \mid \mathtt{loc}(e) = q{+}\pi \right\} = W_\pi$$

As such, given the consistency of the C library L^{C}, we then know that for each $\pi \in \mathbb{N}^+$, the W_π is totally ordered by a strict total order mo.

Observe that the (6), (7) (8), (9) and (11) properties also hold for the weak implementation.

We next demonstrate that:

$$\forall \pi, \pi' \in \mathbb{N}^+.\ d^2_\pi \xrightarrow{\mathsf{com}^{-1}} e^2_\pi \xrightarrow{\mathsf{hb}} e^2_{\pi'} \xrightarrow{\mathsf{com}} d^2_{\pi'} \Rightarrow (d^2_{\pi'}, d^2_\pi) \notin \mathsf{hb} \tag{13}$$

We proceed by contradiction. Let us assume there exist π, π' such that $(e^2_\pi, d^2_\pi) \in \mathsf{com}$, $(e^2_\pi, e^2_{\pi'}) \in \mathsf{hb}$, $(e^2_{\pi'}, d^2_{\pi'}) \in \mathsf{com}$ and $(d^2_{\pi'}, d^2_\pi) \in \mathsf{hb}$. As $e^2_\pi \xrightarrow{\mathsf{hb}} e^2_{\pi'}$, from (11) we know that $\pi < \pi'$. Moreover, as $(e^2_\pi, d^2_\pi) \in \mathsf{com}$, from the consistency of the C library we know that $(e^2_\pi, d^2_\pi) \in \mathsf{mo}|_{\mathrm{imm}}$. As such, since the writes in W_π are totally ordered by mo (see above), we know that either i) $(d^2_\pi, f^\pi_{\pi'}) \in \mathsf{mo}$; or ii) $(f^\pi_{\pi'}, e^2_\pi) \in \mathsf{mo}$.

In case (i) we then have $d^2_\pi \xrightarrow{\mathsf{mo}} f^\pi_{\pi'} \xrightarrow{\mathsf{po}} d^2_{\pi'} \xrightarrow{\mathsf{hb}} d^2_\pi$. That is, we have $d^2_\pi \xrightarrow{\mathsf{mo}} f^\pi_{\pi'} \xrightarrow{\mathsf{hb}} d^2_\pi$, contradicting the assumption that G is consistent.

In case (ii), since G is consistent and $e^2_\pi, e^2_{\pi'}$ are write events, we know that they do not have any incoming so edges. As such, as $\mathsf{hb} = (\mathsf{po} \cup \mathsf{so})^+$, we know that there exists e such that $e^2_\pi \xrightarrow{\mathsf{hb}} e \xrightarrow{\mathsf{po}} e^2_{\pi'}$. Moreover, since $e^1_{\pi'} \xrightarrow{\mathsf{po}|_{\mathrm{imm}}} e^2_{\pi'}$, we also know that $e \xrightarrow{\mathsf{po}} e^1_{\pi'}$. That is, we have $(e^2_\pi, e^1_{\pi'}) \in \mathsf{hb}$. Moreover, from (9) we have $(e^1_{\pi'}, d^1_{\pi'}) \in \mathsf{so} \subseteq \mathsf{hb}$. As such, from the assumption of the case we have $e^2_\pi \xrightarrow{\mathsf{hb}} e^1_{\pi'} \xrightarrow{\mathsf{hb}} d^1_{\pi'} \xrightarrow{\mathsf{po}} f^\pi_{\pi'} \xrightarrow{\mathsf{mo}} e^2_\pi$. That is, we have $e^2_\pi \xrightarrow{\mathsf{hb}} f^\pi_{\pi'} \xrightarrow{\mathsf{mo}} e^2_\pi$, contradicting the assumption that G is consistent.

□

To show that $G' = (E'', \mathsf{po}'', \mathsf{com}'', \mathsf{so}'', \mathsf{lhb}'') \in L^{\mathrm{SQ}}.\mathcal{G}_{\mathrm{c}}$, we are then required to show that for all locations q, G'_q is queue consistent on q. Pick an arbitrary location q and let $G'_q = \langle E', \mathsf{po}', \mathsf{com}', \mathsf{so}', \mathsf{lhb}'\rangle$.
We then need to show:
1) $\mathsf{com}' \subseteq \bigcup_{v \in \mathsf{Val} \setminus \{\bot\}} \mathcal{E}^{q,v} \times \mathcal{D}^{q,v}$
2) com', $(\mathsf{com}')^{-1}$ are functional
3) $E' \cap \mathcal{D}^q \setminus rng(\mathsf{com}') \subseteq \mathcal{D}^{q,\bot}$
4) $[\mathcal{E}^q \setminus dom(\mathsf{com}')]; \mathsf{lhb}'; [\mathcal{D}^{q,\bot}]{=}\emptyset$
5) $\mathsf{so}' = \mathsf{com}'$
6) $\mathsf{com}^{-1}; \mathsf{lhb}; \mathsf{com}; \mathsf{lhb}$ is irreflexive.

Proof of parts (1-5) are as in the case of the strong implementation. For part (6) we proceed by contradiction. Let us assume there exist d_1, d_2, e_1, e_2 such that $d_1 \xrightarrow{\mathsf{com}'^{-1}} e_1 \xrightarrow{\mathsf{hb}'} e_2 \xrightarrow{\mathsf{com}'} d_2 \xrightarrow{\mathsf{hb}'} d_1$. From the definition of com′, the definition of $\mathtt{imp}(.)$ and (6) we then have $\mathtt{imp}(d_1) \xrightarrow{\mathsf{com}^{-1}} \mathtt{imp}(e_1) \xrightarrow{\mathsf{hb}} \mathtt{imp}(e_2) \xrightarrow{\mathsf{com}} \mathtt{imp}(d_2) \xrightarrow{\mathsf{hb}} \mathtt{imp}(d_1)$. From (13) we then have $\mathtt{imp}(d_1) \xrightarrow{\mathsf{hb}} \mathtt{imp}(d_2) \xrightarrow{\mathsf{hb}} \mathtt{imp}(d_1)$. As such, from the transitivity of hb we have $\mathtt{imp}(d_1) \xrightarrow{\mathsf{hb}} \mathtt{imp}(d_1)$, contradicting the assumption that G is consistent.

I THE SOUNDNESS OF THE WEAK STACK IMPLEMENTATION

Let I denote the weak stack implementation in Fig. 6. To show the soundness of I, we appeal to Thm. 1 and show that I is locally sound on L^{WS}.

Pick an arbitrary $\Lambda, f, G{=}\langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}\rangle, E'', \mathsf{po}''$ such that G is Λ-consistent and Λ-well-formed and $\mathsf{abs}_{L^{\mathsf{WS}},I}(f, \langle E, \mathsf{po}\rangle, \langle E'', \mathsf{po}''\rangle)$.
We must next find $\mathsf{com}'', \mathsf{so}''$ such that $\langle E'', \mathsf{po}'', \mathsf{com}'', \mathsf{so}'', \mathsf{lhb}''\rangle \in L^{\mathsf{WS}}.\mathcal{G}_c$, where lhb'' is the same as lhb' in Def. 11.

Let us assume without loss of generality that for each location s, the G contains n_s push operations and m_s pop operations. We will shortly enumerate these operations in order of their lock acquisition. Note that for each $i \in \{1 \cdots n_s\}$, the i^{th} push operation $\mathit{try\text{-}push}(s, v_i)$ either:

- pushes v_i on the stack at s and thus G contains the events in the trace: $\theta_i^{as(s)} = l \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r_t \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} a \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} w_t \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} u$, where $\mathtt{lab}(l) = \mathtt{CAS}(\mathtt{acqrel}, s, 0, 1)$, $\mathtt{lab}(r_t) = \mathtt{load}(\mathtt{rlx}, s{+}1, t)$ for some top value t, $\mathtt{lab}(a) = \mathtt{store}(\mathtt{rlx}, s{+}t{+}1, v_i)$, $\mathtt{lab}(w_t) = \mathtt{store}(\mathtt{rlx}, s{+}1, t{+}1)$, and $\mathtt{lab}(u) = \mathtt{store}(\mathtt{rel}, s, 0)$; or
- fails to push v_i on the stack as it fails to acquire the lock at s, and thus G contains the single-event trace: $\theta_i^{af(s)} = f$, where $\mathtt{lab}(f) = \mathtt{load}(\mathtt{acq}, s, 1)$.

Similarly, for each $i \in \{1 \cdots m_s\}$, the i^{th} pop operation $\mathit{try\text{-}pop}(s)$ either:

- pops w_i from the stack at s and thus G contains the events in the trace: $\theta_i^{as(s)} = l \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r_t \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} p \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} w_t \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} u$, where $\mathtt{lab}(l) = \mathtt{CAS}(\mathtt{acqrel}, s, 0, 1)$, $\mathtt{lab}(r_t) = \mathtt{load}(\mathtt{rlx}, s{+}1, t)$ for some top value t, $\mathtt{lab}(r) = \mathtt{load}(\mathtt{rlx}, s{+}t, v_i)$, $\mathtt{lab}(p) = \mathtt{store}(\mathtt{rlx}, s{+}t, 0)$, $\mathtt{lab}(w_t) = \mathtt{store}(\mathtt{rlx}, s{+}1, t{-}1)$, and $\mathtt{lab}(u) = \mathtt{store}(\mathtt{rel}, s, 0)$; or
- fails to pop from the stack as it fails to acquire the lock at s and thus G contains the single-event trace: $\theta_i^{rf1(s)} = f$, where $\mathtt{lab}(f) = \mathtt{load}(\mathtt{acq}, s, 1)$.
- fails to pop from the stack array as it is empty, and thus G contains the events in the trace: $\theta_i^{rf2(s)} = l \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} f \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} u$, where $\mathtt{lab}(l) = \mathtt{CAS}(\mathtt{acqrel}, s, 0, 1)$, $\mathtt{lab}(f){=}\mathtt{load}(\mathtt{rlx}, s{+}1, 1)$, and $\mathtt{lab}(u) = \mathtt{store}(\mathtt{rel}, s, 0)$.

Moreover, the constructor of the stack at location s contains a trace of the form $\theta^{c(s)} = c_s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} c_l \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} c_t$ where $\mathtt{lab}(c_s) = \mathtt{alloc}(s, 0)$, $\mathtt{lab}(c_l){=}\mathtt{store}(\mathtt{rel}, s, 0)$, and $\mathtt{lab}(c_t){=}\mathtt{store}(\mathtt{rel}, s{+}1, 1)$.

Let us define $\mathtt{imp}(.) : E'' \longrightarrow E$ as:

$$\mathtt{imp}(e) \triangleq \begin{cases} \theta^{c(s)}.c_t & \exists s.\ \theta^{c(s)}.c_t = e \\ \theta_i^{as(s)}.a & \exists i, s.\ f(\theta_i^{as(s)}.a) = e \\ \theta_i^{af(s)}.f & \exists i, s.\ f(\theta_i^{af(s)}.f) = e \\ \theta_i^{rs(s)}.r & \exists i, s.\ f(\theta_i^{rs(s)}.r) = e \\ \theta_i^{rf1(s)}.f & \exists i, s.\ f(\theta_i^{rf1(s)}.f) = e \\ \theta_i^{rf2(s)}.f & \exists i, s.\ f(\theta_i^{rf2(s)}.f) = e \end{cases}$$

Let $\mathsf{so}'' = \mathsf{com}''$ with

$$\mathsf{com}'' \triangleq \left\{(a, r) \,\middle|\, \exists i, j, s.\ f(\theta_i^{as(s)}.a) = af(\theta_j^{rs(s)}.r) = r \wedge (\theta_i^{as(s)}.a, \theta_j^{rs(s)}.r) \in \mathsf{com}\right\}$$

It is straightforward to demonstrate that given the `acqrel` mode of the lock acquisitions, the `rel` mode of lock releases, and the specification of the C library we have:

$$
\begin{array}{l}
\forall s.\ \forall i, j \in \{1 \cdots n{+}m\}.\ \big((\theta_i^{as(s)}.u, \theta_j^{rs(s)}.l) \in \mathsf{hb} \lor (\theta_j^{rs(s)}.u, \theta_i^{as(s)}.l) \in \mathsf{hb}\big) \\
\land \forall s.\ \forall i, j \in \{1 \cdots n{+}m\}.\ (\theta_i^{as(s)}.u, \theta_j^{as(s)}.l) \in \mathsf{hb} \lor (\theta_j^{as(s)}.u, \theta_i^{as(s)}.l) \in \mathsf{hb} \\
\land \forall i, j \in \{1 \cdots n{+}m\}.\ \big((\theta_i^{rs(s)}.u, \theta_j^{rs(s)}.l) \in \mathsf{hb} \lor (\theta_j^{rs(s)}.u, \theta_i^{rs(s)}.l) \in \mathsf{hb}\big)
\end{array}
$$

For each location s, let us then enumerate the *successful* push and pop operations (i.e. those with a $\theta^{as(s)}$ or $\theta^{rs(s)}$ trace) in order of their lock acquisition. That is, the *first* operation is either i) a successful push operation associated with trace $\theta_1^{as(s)}$ such that for all $i \neq 1$, $\theta_1^{as(s)}.u \xrightarrow{\mathsf{hb}} \theta_i^{as(s)}.l$ and $\theta_1^{as(s)}.u \xrightarrow{\mathsf{hb}} \theta_i^{rs(s)}.l$; or ii) a successful pop operation associated with trace $\theta_1^{rs(s)}$ such that for all $i \neq 1$, $\theta_1^{rs(s)}.u \xrightarrow{\mathsf{hb}} \theta_i^{as(s)}.l$ and $\theta_1^{rs(s)}.u \xrightarrow{\mathsf{hb}} \theta_i^{as(s)}.l$.

Let us write $\mathsf{isPush}(s, i)$ when the i^{th} operation on s (as ordered above) is a successful push operation with trace $\theta_i^{as(s)}$. Similarly, let us write $\mathsf{isPop}(s, i)$ when the i^{th} operation on s (as ordered above) is a successful pop operation with trace $\theta_i^{es(s)}$. We can then demonstrate that:

$$
\begin{array}{l}
\forall s.\ \forall i, j \in \{1 \cdots n_s{+}m_s\}.\ i < j \Rightarrow \\
\quad \mathsf{isPush}(s, i) \land \mathsf{isPop}(s, j) \land (\theta_i^{as(s)}.u, \theta_j^{rs(s)}.l) \in \mathsf{hb} \\
\quad \lor \mathsf{isPush}(s, i) \land \mathsf{isPush}(s, j) \land (\theta_i^{as(s)}.u, \theta_j^{as(s)}.l) \in \mathsf{hb} \\
\quad \lor \mathsf{isPop}(s, i) \land \mathsf{isPop}(s, j) \land (\theta_i^{rs(s)}.u, \theta_j^{rs(s)}.l) \in \mathsf{hb}
\end{array}
\tag{14}
$$

Let us write $\mathsf{match}(s, i, j)$ when $\mathsf{isPush}(s, i) \land \mathsf{isPop}(s, j) \land (\theta_i^{as(s)}.a, \theta_j^{rs(s)}.r) \in \mathsf{com}$. Let us write $\mathtt{top}(s, i)$ for t when either: 1) $\mathsf{isPush}(s, i)$ and $\mathtt{lab}(\theta_i^{as(s)}.w_t) = \mathtt{store}(\mathtt{rlx}, s{+}1, t)$; or 2) $\mathsf{isPop}(s, i)$ and $\mathtt{lab}(\theta_i^{rs(s)}.r_t) = \mathtt{load}(\mathtt{rlx}, s{+}1, t)$. It is then straightforward to demonstrate that for all i, j:

$$
\begin{array}{l}
\mathsf{isPush}(s, i) \land \mathsf{isPush}(s, i{+}1) \Rightarrow \mathtt{top}(s, i{+}1) = \mathtt{top}(s, i){+}1 \\
\mathsf{isPush}(s, i) \land \mathsf{isPop}(s, i{+}1) \Rightarrow \mathsf{match}(s, i, i{+}1) \\
\mathsf{match}(s, i, j) \Rightarrow \mathtt{top}(s, j) = \mathtt{top}(s, i)
\end{array}
\tag{15}
$$

It is then straightforward to demonstrate by induction that for all i, j:

$$
\begin{array}{l}
i < j \land \mathsf{isPush}(s, i) \land \mathsf{isPush}(s, j) \Rightarrow \mathtt{top}(s, i) < \mathtt{top}(s, j) \lor \exists k.\ i < k < j \land \mathsf{isPop}(s, k) \land \mathsf{match}(s, i, k) \\
i < j \land \mathsf{isPush}(s, i) \land \mathsf{isPop}(s, j) \Rightarrow \mathtt{top}(s, i) < \mathtt{top}(s, j) \lor \exists k.\ i < k \leq j \land \mathsf{isPop}(s, k) \land \mathsf{match}(s, i, k)
\end{array}
\tag{16}
$$

Let $\mathsf{hb} = G.\mathsf{hb} = (\mathsf{po} \cup \mathsf{so})^+$. Note that from the definition of lhb'', hb and the construction of so' above we know the (17) below holds. As such, from the irreflexivity of hb, we also know (18) holds.

$$
\forall a, b.\ (a, b) \in \mathsf{hb}' \Rightarrow (\mathtt{im}(a), \mathtt{im}(b)) \in \mathsf{hb} \tag{17}
$$

$$
\forall a.\ (a, a) \notin \mathsf{hb}' \tag{18}
$$

To show that $G' = (E'', \mathsf{po}'', \mathsf{com}'', \mathsf{so}'', \mathsf{lhb}'') \in L^{\mathsf{WS}}.\mathcal{G}_c$, we are then required to show that for all locations s, G'_s is weak-stack consistent on s. Pick an arbitrary location s and let $G'_s = \langle E', \mathsf{po}', \mathsf{com}', \mathsf{so}', \mathsf{lhb}' \rangle$. We then need to show:

1) $E'^{c} = \emptyset \lor \exists c \in \mathcal{C}^q.\ E'^{c} = \{c\}$;

2) $\mathsf{com}' \subseteq \bigcup_{v \in \mathsf{Val} \setminus \{\bot\}} \mathcal{A}^{s,v,\top} \times \mathcal{R}^{s,v,\top}$;
3) $\mathsf{com}', \mathsf{com}'^{-1}$ are functional;
4) $E' \cap \mathcal{R}^s \setminus rng(\mathsf{com}') \subseteq \mathcal{R}^{s,\bot,\bot}$

5) $\mathsf{so}' = \mathsf{com}'$; and
6) $\forall a_1, a_2, r_1, r_2.\ (a_1, r_1), (a_2, r_2) \in \mathsf{com}' \wedge (a_1, a_2), (r_1, r_2) \in \mathsf{lhb}' \Rightarrow (a_2, r_1) \notin \mathsf{lhb}'$.

Parts (1), (2) and (5) follow immediately from the construction of G'. Part (3) follows from the definition of com', the consistency of G and the definition of the C library and the fact that com^{-1} is functional for L^{C}.

For part (4), we proceed by contradiction. Let us assume there exists $r \in E' \cap \mathcal{R}^s$ such that $r \notin rng(\mathsf{com}')$ and $r \notin \mathcal{R}^{s,\perp,\perp}$. Let $\mathtt{lab}(r) = \mathtt{try\text{-}pop}(s, v, -)$. From the construction of G' we then know that there exists i and r' such that $r' = \mathtt{imp}(r)$ and either: i) $r' = \theta_i^{rs(s)}.r$ and $r = r_i^{rs(s)}$; or ii) $r' = \theta_i^{rf1(s)}.f$ and $r = r_i^{rf1(s)}$; or iii) $r' = \theta_i^{rf2(s)}.f$ and $r = r_i^{rf2(s)}$. In cases (ii) and (iii) from the construction of G' we then know $\mathtt{lab}(r) = \mathtt{try\text{-}pop}(s, \perp, \perp)$ and thus $r \in \mathcal{R}^{s,\perp,\perp}$, contradicting the assumption that $r \notin \mathcal{R}^{s,\perp,\perp}$. In case (i), from the shape of the $\theta_i^{rs(s)}$ we know there exists t such that $\mathtt{lab}(\theta_i^{rs(s)}.r_t) = \mathtt{load}(\mathtt{rlx}, s{+}1, t)$, $\mathtt{lab}(r') = \mathtt{load}(\mathtt{rlx}, s{+}t, v)$. Moreover, from the consistency of the C library we know there exists w such that $(w, r') \in \mathsf{com}$ and $\mathtt{lab}(w) = \mathtt{store}(-, s{+}t, v)$. As such, from the well-formedness of G and the shape of the implementation traces we know that there exists j such that $w = \theta_j^{as(s)}.a$. Consequently, since $(w, r')'in\mathsf{com}$, from the construction of G' we know that $(a_j^{as(s)}, r_i^{rs(s)}) \in \mathsf{com}'$. Since $r = r_i^{rs(s)}$, this contradicts our assumption that $r \notin rng(\mathsf{com}')$.

For part (6), we proceed by contradiction. Let us assume there exist a_1, a_2, r_1, r_2 such that $(a_1, r_1), (a_2, r_2) \in \mathsf{com}'$, $(a_1, a_2), (r_1, r_2) \in \mathsf{lhb}'$ and $(a_2, r_1) \in \mathsf{lhb}'$.

From the construction of G' we then know there exist i, j, k, l such that $\mathtt{imp}(a_1) = \theta_i^{as(s)}.a$, $\mathtt{imp}(r_1) = \theta_j^{rs(s)}.r$, $\mathtt{imp}(a_2) = \theta_k^{as(s)}.a$, $\mathtt{imp}(r_2) = \theta_l^{rs(s)}.r$, and $(\mathtt{imp}(a_1), \mathtt{imp}(r_1)), (\mathtt{imp}(a_2), \mathtt{imp}(r_2)) \in \mathsf{com}$. That is we have $\mathrm{match}(s, i, j)$ and $\mathrm{match}(s, j, k)$.

From (19) we then know $(\mathtt{imp}(a_1), \mathtt{imp}(a_2)), (\mathtt{imp}(r_1), \mathtt{imp}(r_2)), (\mathtt{imp}(a_2), \mathtt{imp}(r_1)) \in \mathsf{hb}$. From (14) we then have $i < k$, $k < j$ and $j < l$, since otherwise we would get an hb cycle contradicting the assumption that G is consistent. As such, since we also have $\mathrm{match}(s, i, j)$ and $\mathrm{match}(s, k, l)$, and $\mathrm{match}(s, ., .)$ is uniquely determined (due to the functionality of com and com^{-1}), from (16) we have $\mathtt{top}(s, i) < \mathtt{top}(s, k)$ and $\mathtt{top}(s, k) < \mathtt{top}(s, j)$. That is, we have $\mathtt{top}(s, i) < \mathtt{top}(s, j)$. On the other hand, since $\mathrm{match}(s, i, j)$ holds, from (15) we have $\mathtt{top}(s, i) = \mathtt{top}(s, j)$. This however leads to a contradiction as we both have $\mathtt{top}(s, i) < \mathtt{top}(s, j)$ and $\mathtt{top}(s, i) = \mathtt{top}(s, j)$.

J THE SOUNDNESS OF THE ELIMINATION STACK IMPLEMENTATION

Let I denote the elimination stack implementation in Fig. 6. To show the soundness of I, we appeal to Thm. 1 and show that I is locally sound on L^{S}.

Pick an arbitrary $\Lambda, f, G{=}\langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}\rangle, E'', \mathsf{po}''$ such that G is Λ-consistent and Λ-well-formed and $\mathsf{abs}_{L^{\mathsf{S}},I}(f, \langle E, \mathsf{po}\rangle, \langle E'', \mathsf{po}''\rangle)$.
We must next find $\mathsf{com}'', \mathsf{so}''$ such that $\langle E'', \mathsf{po}'', \mathsf{com}'', \mathsf{so}'', \mathsf{lhb}''\rangle \in L^{\mathsf{S}}.\mathcal{G}_{\mathsf{c}}$, where lhb'' is the same as lhb' in Def. 11.

Note that the constructor of the stack at s contains a trace of the following form in G: $\theta^{c(s)} = c_s \xrightarrow{\mathsf{po}} c_{ws} \xrightarrow{\mathsf{po}} c_{ea} \xrightarrow{\mathsf{po}} c^0_{ea} \xrightarrow{\mathsf{po}} \cdots \xrightarrow{\mathsf{po}} c^{k-1}_{ea} \xrightarrow{\mathsf{po}} c^w_{ws} \xrightarrow{\mathsf{po}} c^w_{ea}$ where $\mathtt{lab}(c_s) = \mathtt{alloc}(s, 0)$, $\mathtt{lab}(c_{ws}){=}\mathtt{alloc}(ws, 0)$, $\mathtt{lab}(c_{ea}){=}\mathtt{alloc}(ea, 0)$, $\mathtt{lab}(c^w_{ws}){=}\mathtt{store}(\mathtt{rlx}, s, ws)$, $\mathtt{lab}(c^w_{ea}) = \mathtt{store}(\mathtt{rlx}, s{+}1, ea)$, and for each $j \in \{0 \cdots k{-}1\}$, c^j_{ea} is of the form $n_j \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} w_j$, where $\mathtt{lab}(n_j) = \mathtt{new\text{-}exchanger}(x_j)$ for some x_j, and $\mathtt{lab}(w_j) = \mathtt{store}(\mathtt{rlx}, ea{+}j, x_j)$.

When G contains n_s push operations on the stack at s and m_s pop operations, let us enumerate them arbitrarily. Note that for each $i \in \{1 \cdots n_s\}$, the i^{th} push operation $push(s, v_i)$ either:

- pushes v_i on the weak stack at s and thus G contains the events in the trace: $\theta^{as(s)}_i = r_s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r_e \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} f^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} a$, where $\mathtt{lab}(r_s) = \mathtt{load}(\mathtt{rlx}, s, ws)$, $\mathtt{lab}(r_e) = \mathtt{load}(\mathtt{rlx}, s{+}1, ea)$, f^* denotes the loop iterations that fail to push v_i, $\mathtt{lab}(a) = \mathtt{try\text{-}push}(ws, v_i, 1)$; or
- fails to push v_i on the weak stack and thus pushes it on the elimination array at $s{+}1$; as such G contains the events in the trace: $\theta^{ae(s)}_i = r_s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r_e \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} f^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} a_f \xrightarrow{\mathsf{po}} a$, where $\mathtt{lab}(r_s) = \mathtt{load}(\mathtt{rlx}, s, ws)$, $\mathtt{lab}(r_e) = \mathtt{load}(\mathtt{rlx}, s{+}1, ea)$, f^* denotes the loop iterations that fail to push v, $\mathtt{lab}(a_f) = \mathtt{try\text{-}push}(ws, v_i, 0)$, $\mathtt{lab}(a) = \mathtt{exchange}(ea[j], v_i, \mathtt{POP})$ for some $j \in \{0 \cdots k{-}1\}$.

Similarly, for each $i \in \{1 \cdots m_s\}$, the i^{th} pop operation $pop(s)$ either:

- pops w_i from the weak stack and thus G contains the events in the trace: $\theta^{rs}_i = r_s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r_e \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} f^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r$, where $\mathtt{lab}(r_s){=}\mathtt{load}(\mathtt{rlx}, s, ws)$, $\mathtt{lab}(r_e) = \mathtt{load}(\mathtt{rlx}, s{+}1, ea)$, f^* denotes loop iterations that fail to pop, and $\mathtt{lab}(r) = \mathtt{try\text{-}pop}(ws, w_i, 1)$; or
- fails to pop from the weak stack and thus pops w_i from the elimination array at $s{+}1$; as such, G contains the events in the trace: $\theta^{re}_i = r_s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r_e \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} f^* \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r_f \xrightarrow{\mathsf{po}} r$, where $\mathtt{lab}(r_s){=}\mathtt{load}(\mathtt{rlx}, s, ws)$, $\mathtt{lab}(r_e){=}\mathtt{load}(\mathtt{rlx}, s{+}1, ea)$, f^* denotes the iterations that fail to pop, $\mathtt{lab}(r_f){=}\mathtt{try\text{-}pop}(ws, \bot, 0)$, and $\mathtt{lab}(r){=}\mathtt{exchange}(ea[j], \mathtt{POP}, w_i)$, for some $j \in \{0 \cdots k{-}1\}$.

Let us define $\mathtt{imp}(.) : E'' \to E$ as:

$$\mathtt{imp}(e) \triangleq \begin{cases} \theta^{c(s)}.c^w_{ea} & \exists s.\ \theta^{c(s)}.c^w_{ea} = e \\ \theta^{as(s)}_i.a & \exists i, s.\ f(\theta^{as(s)}_i.a) = e \\ \theta^{ae(s)}_i.a & \exists i, s.\ f(\theta^{ae(s)}_i.a) = e \\ \theta^{rs(s)}_i.r & \exists i, s.\ f(\theta^{rs(s)}_i.r) = e \\ \theta^{re(s)}_i.r & \exists i, s.\ f(\theta^{re(s)}_i.r) = e \end{cases}$$

Let $\mathsf{so}'' = \mathsf{com}''$ with

$$\begin{aligned} \mathsf{com}'' \triangleq\ & \left\{(a, r) \,\middle|\, \exists i, j, s.\ f(\theta^{as(s)}_i.a) = af(\theta^{rs(s)}_j.r) = r \land (\theta^{as(s)}_i.a, \theta^{rs(s)}_j.r) \in \mathsf{com}\right\} \\ \cup & \left\{(a, r) \,\middle|\, \exists i, j, s.\ f(\theta^{ae(s)}_i.a) = af(\theta^{re(s)}_j.r) = r \land (\theta^{ae(s)}_i.a, \theta^{re(s)}_j.r) \in \mathsf{com}\right\} \end{aligned}$$

Let $\text{hb} = G.\text{hb} = (\text{po} \cup \text{so})^+$. Note that from the definition of lhb'', hb and the construction of so' above we know the (19) below holds. As such, from the irreflexivity of hb, we also know (20) holds. Let $\text{hb}' = (\text{po}' \cup \text{so}')^+$. We next demonstrate that:

$$\forall a, b.\ (a, b) \in \text{hb}' \Rightarrow (\texttt{im}(a), \texttt{im}(b)) \in \text{hb} \tag{19}$$

$$\forall a.\ (a, a) \notin \text{hb}' \tag{20}$$

To show that $G' = (E'', \text{po}'', \text{com}'', \text{so}'', \text{lhb}'') \in L^{\mathsf{S}}.\mathcal{G}_{\mathsf{c}}$, we are then required to show that for all locations s, G'_s is stack consistent on s. Pick an arbitrary location s and let $G'_s = \langle E', \text{po}', \text{com}', \text{so}', \text{lhb}' \rangle$. We then need to show:

1) $E'^c = \emptyset \vee \exists c \in C^q.\ E'^c = \{c\}$;
2) $\text{com}' \subseteq \bigcup_{v \in \mathsf{Val} \setminus \{\perp\}} \mathcal{A}^{s,v} \times \mathcal{R}^{s,v}$;
3) $\text{com}', \text{com}'^{-1}$ are functional;
4) $E' \cap \mathcal{R}^s \setminus rng(\text{com}') \subseteq \mathcal{R}^{s,\perp}$
5) $[\mathcal{A}^s \setminus dom(\text{com}')]; \text{hb}'; [\mathcal{R}^{s,\perp}] = \emptyset$;
6) $\text{so}' = \text{com}'$; and
7) $\forall a_1, a_2, r_1, r_2.\ (a_1, r_1), (a_2, r_2) \in \text{com}' \wedge (a_1, a_2), (r_1, r_2) \in \text{hb}' \Rightarrow (a_2, r_1) \notin \text{hb}'$.

Parts (1), (2) and (6) follow immediately from the construction of G'. Part (3) follows from the construction of G' and the fact that com and com^{-1} are functional for L^{WS} and L^{X}. Part (5) follows trivially as $\mathcal{R}^{s,\perp} = \emptyset$.

For part (4), we demonstrate that $E' \cap \mathcal{R}^s \setminus rng(\text{com}') = \emptyset$. We proceed by contradiction. Let us assume there exists $r \in E' \cap \mathcal{R}^s$ such that $r \notin rng(\text{com}')$. Let $\texttt{lab}(r) = \mathsf{pop}(s, v)$ for some value v. From the construction of G' we then know that there exists i and r' such that $r' = \texttt{imp}(r)$ and either: i) $r' = \theta_i^{rs(s)}.r$; or ii) $r' = \theta_i^{re(s)}.r$.

In case (i) we then know that $\texttt{lab}(r') = \texttt{try-pop}(ws, v, 1)$. As such, from the consistency of G and the specification of the weak stack library we know there exist w' such that $\texttt{lab}(w)' = \texttt{try-push}(ws, v, 1)$ and $(w', r') \in \text{com}$. Moreover, given the well-formedness of G and the shape of G traces we know that there exist j such that $w' = \theta_j^{as(s)}.a$. Consequently, from the construction of G' we know there exist w such that $\texttt{imp}(w) = w'$ and $(w, r) \in \text{com}'$, contradicting our assumption that $r \notin rng(\text{com}')$.

Similarly, in case (ii) we then know that $\texttt{lab}(r') = \texttt{exchange}(ea[j], \texttt{POP}, v)$ for some j. As such, from the consistency of G and the specification of the exchanger library we know there exist w' such that $\texttt{lab}(w)' = \texttt{exchange}(ea[j], v, \texttt{POP})$ and $(w', r'), (r', w') \in \text{com}$. Moreover, given the well-formedness of G and the shape of G traces we know that there exist j such that $w' = \theta_j^{ae(s)}.a$. Consequently, from the construction of G' we know there exist w such that $\texttt{imp}(w) = w'$ and $(w, r) \in \text{com}'$, contradicting our assumption that $r \notin rng(\text{com}')$.

For part (7) we proceed by contradiction. Let us assume there exist a_1, a_2, r_1, r_2 such that $(a_1, r_1), (a_2, r_2) \in \text{com}'$, $(a_1, a_2), (r_1, r_2) \in \text{lhb}'$, and $(a_2, r_1) \in \text{lhb}'$. From (19) and the construction of G' we know $(\texttt{imp}(a_1), \texttt{imp}(r_1)), (\texttt{imp}(a_2), \texttt{imp}(r_2)) \in \text{com}$, $(\texttt{imp}(a_1), \texttt{imp}(a_2)), (\texttt{imp}(r_1), \texttt{imp}(r_2)) \in \text{hb}$, and $(\texttt{imp}(a_2), \texttt{imp}(r_1)) \in \text{hb}$. There are now three cases to consider: i) $\texttt{imp}(a_2) = \theta_i^{as(s)}.a$, $\texttt{imp}(r_2) = \theta_j^{rs(s)}$, $\texttt{imp}(a_1) = \theta_k^{as(s)}.a$, $\texttt{imp}(r_1) = \theta_l^{rs(s)}$ for some i, j, k, l; or ii)$\texttt{imp}(a_2) = \theta_i^{ae(s)}.a$ and $\texttt{imp}(r_2) = \theta_j^{re(s)}$ for some i, j; or iii) $\texttt{imp}(a_1) = \theta_i^{ae(s)}.a$ and $\texttt{imp}(r_1) = \theta_j^{re(s)}$ for some i, j.s

Case (i) however leads to contradiction as it contradicts the L^{WS}-consistency of $(G)_{L^{\mathsf{WS}}}$. In case (ii) from the definition of the L^{X} library and the symmetry of its com relation we know that

$(\texttt{imp}(r_2), \texttt{imp}(a_2)) \in \textcolor{red}{\mathsf{com}}$. As such, we have $\texttt{imp}(a_2) \xrightarrow{\mathsf{hb}} \texttt{imp}(r_1) \xrightarrow{\mathsf{hb}} \texttt{imp}(r_2) \xrightarrow{\mathsf{com}} \texttt{imp}(a_2)$. That is, $\texttt{imp}(a_2) \xrightarrow{\mathsf{hb}} \texttt{imp}(r_2) \xrightarrow{\mathsf{com}} \texttt{imp}(a_2)$, contradicting the assumption that G is consistent.

Similarly, in case (iii) from the definition of the L^{X} library and the symmetry of its com relation we know that $(\texttt{imp}(r_1), \texttt{imp}(a_1)) \in \mathsf{com}$. As such, we have $\texttt{imp}(a_1) \xrightarrow{\mathsf{hb}} \texttt{imp}(a_2) \xrightarrow{\mathsf{hb}} \texttt{imp}(r_1) \xrightarrow{\mathsf{com}} \texttt{imp}(a_1)$. That is, $\texttt{imp}(a_1) \xrightarrow{\mathsf{hb}} \texttt{imp}(r_1) \xrightarrow{\mathsf{com}} \texttt{imp}(a_1)$, contradicting the assumption that G is consistent.

K THE SOUNDNESS OF EXCHANGER IMPLEMENTATION

Let I denote the exchanger implementation in Fig. 5. To show the soundness of I, we appeal to Thm. 1 and show that I is locally sound on L^{X}.

Pick an arbitrary $\Lambda, f, G{=}\langle E, \mathsf{po}, \mathsf{com}, \mathsf{so}\rangle, E', \mathsf{po}'$ such that G is Λ-consistent and Λ-well-formed and $\mathsf{abs}_{L^{\mathsf{X}},I}(f, \langle E, \mathsf{po}\rangle, \langle E', \mathsf{po}'\rangle)$. We must next find $\mathsf{com}', \mathsf{so}'$ such that $\langle E', \mathsf{po}', \mathsf{com}', \mathsf{so}', \mathsf{lhb}'\rangle \in L^{\mathsf{X}}.\mathcal{G}_{\mathsf{c}}$, where lhb' is as defined in Def. 11.

Note that each exchange operation $exchange(g, v)$ either:

(1) offers a value at index $g{+}k$ (for some $k \in \mathbb{N}^+$ where k is an odd number) and returns unmatched due to a timeout; or
(2) offers a value at index $g{+}k$ (for some $k \in \mathbb{N}^+$ where k is an odd number) and matches with value v' at index $g{+}k{+}1$; or
(3) tries to offer a value at index $g{+}k$ (for some $k \in \mathbb{N}^+$ where i is an even number) and returns unmatched as the slot at index $g{+}k$ is already taken; or
(4) offers a value at index $g{+}k$ (for some $k \in \mathbb{N}^+$ where k is an even number) and matches with value v' at index $g{+}k{-}1$.

Without loss of generality, let us assume e contains n exchange calls. For each i^{th} exchange operation of the form $exchange(g, v_i)$, the G_i contains a trace of one of the following forms depending which of the four categories above it falls into:

- $\theta_i^{t(g)}$ is of the form $s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} o \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} t$, with $\texttt{lab}(s) = \texttt{load}(\texttt{rlx}, g, j_i)$, $\texttt{lab}(o) = \texttt{CAS}(\texttt{rel}, g{+}j_i, 0, v_i)$, and $\texttt{lab}(t) = \texttt{CAS}(\texttt{rlx}, g{+}j_i{+}1, 0, \bot)$, for some j_i and v_i where j_i is odd;
- $\theta_i^{o(g)}$ is of the form $s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} o \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} a \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r$, with $\texttt{lab}(s) = \texttt{load}(\texttt{rlx}, g, j_i)$, $\texttt{lab}(o) = \texttt{CAS}(\texttt{rel}, g{+}j_i, 0, v_i)$, $\texttt{lab}(a) = \texttt{load}(\texttt{rlx}, g{+}j_i{+}1, v_i')$ with $v_i' \neq 0$, $\texttt{lab}(r) = \texttt{load}(\texttt{acq}, g{+}j_i{+}1, v_i')$, for some j_i, v_i and v_i' where j_i is odd;
- $\theta_i^{f(g)}$ is of the form $s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} f \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} o \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} c$, with $\texttt{lab}(s) = \texttt{load}(\texttt{rlx}, g, j_i)$, $\texttt{lab}(f) = \texttt{load}(\texttt{rlx}, g{+}j_i, w_i)$ with $w_i \neq 0$, $\texttt{lab}(o) = \texttt{load}(\texttt{rlx}, g{+}j_i{+}1, w_i')$ with $w_i' \neq 0$, and $\texttt{lab}(c) = \texttt{CAS}(\texttt{rlx}, g, j_i, j_i{+}2)$ or $\texttt{lab}(c_i) = \texttt{load}(\texttt{rlx}, g, u_i)$ with $u_i \neq j_i$, for some j_i and v_i where j_i is odd;
- $\theta_i^{e(g)}$ is of the form $s \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} f \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} o \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} c \xrightarrow{\mathsf{po}|_{\mathsf{imm}}} r$, with $\texttt{lab}(s) = \texttt{load}(\texttt{rlx}, g, j_i)$, $\texttt{lab}(f) = \texttt{load}(\texttt{rlx}, g{+}j_i, w_i)$ with $w_i \neq 0$, $\texttt{lab}(o) = \texttt{CAS}(\texttt{rel}, g{+}j_i{+}1, 0, v_i)$, $\texttt{lab}(c) = \texttt{CAS}(\texttt{rlx}, g, j_i, j_i{+}2)$ or $\texttt{lab}(c) = \texttt{load}(\texttt{rlx}, g, u_i)$ with $u_i \neq j_i$, $\texttt{lab}(r) = \texttt{load}(\texttt{acq}, g{+}j_i, v_i')$, for some j_i, v_i and v_i' where j_i is odd.

Let $\mathsf{so}' = \mathsf{com}'$ with

$$\mathsf{com}' \triangleq \left\{ (a, b), (b, a) \;\middle|\; \begin{array}{l} \exists i, j, g.\ f(e_i^{o(g)}.o){=}a \wedge f(e_j^{e(g)}.o){=}b \\ \wedge (\theta_i^{o(g)}.o, \theta_j^{e(g)}.r) \in \mathsf{com} \wedge (\theta_j^{e(g)}.o, \theta_i^{o(g)}.r) \in \mathsf{com} \end{array} \right\}$$

To show that $G' = \langle E', \mathsf{po}', \mathsf{com}', \mathsf{so}', \mathsf{lhb}'\rangle \in L^{\mathsf{X}}.\mathcal{G}_{\mathsf{c}}$, we are then required to show for all $g \in \mathrm{Loc}$, G_g' is exchanger-consistent on g. Pick an arbitrary $g \in \mathrm{Loc}$, we then need to show:

1) $E'^{c} = \emptyset \vee \exists c \in C^{g}.\ E'^{c} = \{c\}$;
2) com′ is *symmetric*, *irreflexive* and $\mathsf{com}' \subseteq \bigcup_{v_1, v_2 \in \mathsf{Val}} \mathcal{X}^{g, v_1, v_2} \times \mathcal{X}^{g, v_2, v_1} \setminus \mathsf{id}$;
3) com′ is functional;
4) $E' \cap \mathcal{X}^{g} \setminus \mathit{dom}(\mathsf{com}') \subseteq \bigcup_{v \in \mathsf{Val}} \mathcal{X}^{g, v, \perp}$; and
5) $\mathsf{so}' = \mathsf{com}'$.

Parts (1), (2), and (5) follow immediately from the construction of G_s and the consistency of the C library. The proof of parts (3) and (4) follows from the definition of com′, the consistency of G_i and the definition of the C library.