Awamoche: Proof of Correctness

1 OPERATIONAL STEPS

Read

$$\frac{r \in \mathsf{avail}(G) \cap \mathsf{R}_l \setminus \mathsf{R}_l^{\mathsf{conf}} \qquad w \in G.\mathsf{W}_l \qquad \neg(\mathsf{spec}(r,w) \vee \mathsf{excl}(r,w)) \qquad \mathsf{block}(r,w) \Rightarrow w = \max_{G.\mathsf{co}}\{w' \in G.\mathsf{W}_l\}}{G \xrightarrow{r@w} \mathsf{AddRead}(G,r,w)}$$

Read-spec

$$\frac{r \in \mathsf{avail}(G) \cap \mathsf{R}_l \setminus \mathsf{R}^{\mathsf{conf}} \qquad w \in G.\mathsf{W}_l \qquad \mathsf{spec}(r,w) \vee \mathsf{excl}(r,w) \qquad \forall r' \in \mathsf{R}_l^{\mathsf{spec,excl}}.\, \mathit{completed}(G,r') \qquad \nexists r' \in G.\mathsf{R}_l^{\mathsf{spec,excl}}.\, G.\mathtt{rf}(r') = w}{G \xrightarrow{r@w} \mathsf{AddRead}(G,r,w)}$$

Read-bot

$$\frac{r \in \mathsf{avail}(G) \cap \mathsf{R}_l \setminus \mathsf{R}^{\mathsf{conf}} \qquad \forall r' \in \mathsf{R}_r.\, \neg \mathit{bottom}(G,r) \qquad \exists r' \in G.\mathsf{R}_l^{\mathsf{spec,excl}}.\, \neg \mathit{completed}(G,r') \wedge (\mathsf{spec}(r, G.\mathtt{rf}(r')) \vee \mathsf{excl}(r, G.\mathtt{rf}(r')))}{G \xrightarrow{r@\bot} \mathsf{AddRead}(G,r,\bot)}$$

Read-conf

$$\frac{r \in \mathsf{avail}(G) \cap \mathsf{R}_l^{\mathsf{conf}} \qquad w = G.\mathtt{rf}(\max_{G.\mathsf{po}} \{e \in G.\mathsf{R}_l^{\mathsf{spec}} \mid \mathtt{tid}(e) = \mathtt{tid}(w)\})}{G \xrightarrow{r@w} \mathsf{AddRead}(G,r,w)}$$

Write

$$\frac{w \in \mathsf{avail}(G) \cap \mathsf{W}_l \setminus \mathsf{W}^{\mathsf{excl,conf}} \qquad w_p \in G.\mathsf{W}_l}{G \xrightarrow{w@w_p} \mathsf{SetRFs}(\mathsf{AddWrite}(G,w,w_p), G.\mathsf{MBR}_l, w)}$$

Write-RMW

$$\frac{w \in \mathsf{avail}(G) \cap \mathsf{W}^{\mathsf{excl,conf}} \qquad w_p = G.\mathtt{rf}(\max_{G.\mathsf{po}} \{e \in G.\mathsf{E} \mid \mathtt{tid}(e) = \mathtt{tid}(w)\})}{G \xrightarrow{w@w_p} \mathsf{SetRFs}(\mathsf{AddWrite}(G,w,w_p), G.\mathsf{MBR}_l \cup G.\mathsf{R}_l^{\bot}, w)}$$

Plain

$$\frac{e \in \mathsf{avail}(G) \setminus \mathsf{R} \cup \mathsf{W}}{G \xrightarrow{e@-} \mathsf{Add}(G,e)}$$

2 MAXIMAL STEPS

Read

$$\frac{r \in \mathsf{R}_l \setminus \mathsf{R}^{\mathsf{conf}} \qquad G \xrightarrow{r@w} G' \qquad w = \max_{G.\mathsf{co}}\{e \in G.\mathsf{W}_l\} \vee w = \bot}{G \overset{r}{\rightsquigarrow} G'}$$

Write

$$\frac{w \in \mathsf{W}_l \setminus \mathsf{W}^{\mathsf{excl,conf}} \qquad G \xrightarrow{w@w_p} G' \qquad w_p = \max_{G.\mathsf{co}}\{e \in G.\mathsf{W}_l\}}{G \overset{w}{\rightsquigarrow} G'}$$

Rest

$$\frac{e \in \mathsf{R}^{\mathsf{conf}} \cup \mathsf{W}^{\mathsf{excl,conf}} \cup (\mathsf{E} \setminus \mathsf{R} \cup \mathsf{W}) \qquad G \xrightarrow{e\,@\_} G'}{G \overset{e}{\rightsquigarrow} G'}$$

3 ALGORITHMIC STEPS

Non-revisit Step

$$\frac{e \in \mathsf{nextCandidates}(G) \qquad \neg \mathsf{racy}(G, e) \qquad G \xrightarrow{e\,@p} G' \qquad p \neq \bot \qquad \mathsf{consistent}(G')}{G \underset{nr}{\overset{e\,@p}{\Rightarrow}} G'}$$

Write Revisit Step

$$\frac{\begin{array}{c} w \in \mathsf{nextCandidates}(G) \cap \mathsf{W}_l \qquad \neg \mathsf{racy}(G, w) \qquad r \in \mathsf{R}_l \setminus \mathsf{R}^{\mathsf{block},\bot} \\ [d_1, \cdots, d_n] = \mathsf{sort}_<(\{e \in G.\mathsf{E} \mid r < e \wedge \langle e, w\rangle \notin \mathsf{Add}(G, w, -).\mathsf{po}\mathtt{rf}\}) \\ G_\emptyset \sqsubseteq G'' \overset{r}{\rightsquigarrow} \overset{d_1}{\rightsquigarrow} \cdots \overset{d_n}{\rightsquigarrow} G \qquad G'' \xrightarrow{w\,@w_p} \xrightarrow{r\,@w} G' \qquad \mathsf{consistent}(G') \end{array}}{\langle G, <\rangle \underset{rv\ r}{\overset{w\,@w_p}{\Rightarrow}} \langle G', \mathsf{Restrict}(<, G''.\mathsf{E} \cup \{r\}) +\!\!+ [w]\rangle}$$

Spec Revisit Step

$$\frac{\begin{array}{c} r \in \mathsf{nextCandidates}(G) \cap \mathsf{R}_l^{\mathsf{spec,excl}} \qquad w \in \mathsf{W}_l \qquad \mathsf{spec}(r, w) \vee \mathsf{excl}(r, w) \\ r' \in G.\mathsf{R}_l^{\mathsf{spec,excl}} \qquad G.\mathtt{rf}(r') = w \\ [d_1, \cdots, d_n] = \mathsf{sort}_<(\{e \in G.\mathsf{E} \mid r' < e \wedge \langle e, r\rangle \notin \mathsf{Add}(G, r, w).\mathsf{po}\mathtt{rf}\}) \\ G_\emptyset \sqsubseteq G'' \overset{r'}{\rightsquigarrow} \overset{d_1}{\rightsquigarrow} \cdots \overset{d_n}{\rightsquigarrow} G \qquad G'' \xrightarrow{r\,@w} \xrightarrow{r'\,@\bot} G' \qquad \mathsf{consistent}(G') \end{array}}{\langle G, <\rangle \underset{rv\ r'}{\overset{r\,@w}{\Rightarrow}} \langle G', \mathsf{Restrict}(<, G''.\mathsf{E} \cup \{r'\} +\!\!+ [r])\rangle}$$

4 DEFINITIONS

Definition 4.1 (Prefix). Given consistent executions G and G', we say that G is a *prefix* of G', and write $G \sqsubseteq G'$, if G can be extended to G' with a series of operational steps.

$$G \sqsubseteq G' \triangleq G \rightarrow^* G'$$

Definition 4.2 (Available prefix). Given a consistent execution G and an event $e \in \mathsf{avail}(G)$, we define the *available prefix* of G w.r.t. e, as the restriction of G to the events in the $G.\mathsf{po}\mathtt{rf}$-prefix of the po-last event e' of the thread of e in G, i.e.,

$\mathsf{availPrefix}(G, e) \triangleq G|_{dom(G.\mathsf{po}\mathtt{rf}?;[e'])}$, where $e' \triangleq \max_{G.\mathsf{po}} \{e'' \in G.\mathsf{E} \mid \mathtt{tid}(e') = \mathtt{tid}(e)\}$.

Definition 4.3 (Full Execution). Given an execution G, we say that G is full, and write *full*(G), if $\mathsf{avail}(G) = \emptyset$.

Definition 4.4 (Pending Read). Given an execution G and a read $r \in G.\mathsf{R}$, we say than r is pending in G, and write *pending*(G, r), if either $r \in \mathsf{R}^{\mathsf{conf}}$, $G.\mathtt{rf}(r) \neq \bot$, and $\neg$*completed*(G, r), or $r \in \mathsf{R}^{\mathsf{excl}} \setminus \mathsf{R}^{\mathsf{conf}}$, $rng([r]; G.\mathsf{po}) = \emptyset$, and $\mathtt{val}(G.\mathtt{rf}(r))$ makes the RMW operation succeed.

Definition 4.5 (Well-picked). Given an execution G and an event $e \in \mathsf{avail}(G)$, we say that e is well-picked for G, and write *well−picked*(G, e), if

$$(e \in \mathsf{W}^l \vee (e \in \mathsf{R}^l \wedge \mathsf{spec}(e) \cup \mathsf{excl}(e) \neq \emptyset)) \Rightarrow \nexists e' \in G'.\mathsf{R}_l^{\mathsf{spec,excl}}.\, \mathit{pending}(G, e')$$

Definition 4.6 (Next Candidates). Given an execution G, we define the set of next candidates

$$\mathsf{nextCandidates}(G) \triangleq \{e \in \mathsf{avail}(G) \mid \mathit{well{-}picked}(G, e)\}$$

4.1 Blocking

Definition 4.7. Given an execution G, we write $G.\mathsf{BR}$ for the set of blocked read events.

$$G.\mathsf{BR} \triangleq \{r \in G.\mathsf{R} \mid \mathtt{valw}(G.\mathtt{rf}(r)) \in \mathtt{blockv}(r)\}$$

Definition 4.8. Given an execution G, we write $G.\mathsf{MBR}$ for the set of coherence-maximal blocked read events.

$$G.\mathsf{MBR} \triangleq \left\{ r \in G.\mathsf{BR} \;\middle|\; rng([r]; G.\mathtt{fr}) = \emptyset \right\}$$

4.2 Confirmation

Definition 4.9. Given an execution G and a read $r \in G.\mathsf{R}^{\mathsf{spec}}$, we say that r is completed in G, and write $completed(G, r)$, if r is followed by a same-location confirmation write, i.e.,

$$completed(G, r) \triangleq \exists e \in \mathsf{W}^{\mathsf{conf}}_{\mathsf{loc}(r)}.\ \langle r, e\rangle \in G.\mathsf{po}$$

4.3 Lemmas

Lemma 4.10 (Progress). *Given an execution $G \sqsupseteq G_\emptyset$, if* $\mathsf{avail}(G) \neq \emptyset$ *then* $\mathsf{nextCandidates}(G) \neq \emptyset$.

Lemma 4.11 (Maximal Extensibility). *Given an execution $G \sqsupseteq G_\emptyset$ such that* $\neg\mathsf{racy}(G)$ *and* $e \in \mathsf{nextCandidates}(G)$, *there is a unique execution G' such that $G \overset{e}{\rightsquigarrow} G'$.*

Lemma 4.12 (Maximal Consistency). *Given two execution such that $G_\emptyset \sqsubseteq G \rightsquigarrow G'$, if G is consistent and* $\neg\mathsf{racy}(G)$, *then G' is consistent.*

Lemma 4.13 (Prefix Closedness). *Given two executions G and G' s.t. $G \sqsubseteq G \sqsubseteq G'$, if G' is consistent, then G is consistent.*

Lemma 4.14 (Prefix Extension). *Given two executions G and G' s.t. $G_\emptyset \sqsubseteq G \sqsubset G'$, and an event $e \in G'.\mathsf{E} \setminus G.\mathsf{E}$, there is a unique execution G_p s.t. $G \sqsubseteq G_p \sqsubseteq G'$ and $G_p.\mathsf{E} \setminus G.\mathsf{E} = dom(G'.\mathtt{porf}; [e]) \setminus G.\mathsf{E}$. We denote call this execution the* prefix-extension *of G until the event e of G', and denote is as $\textsc{PrefixExtension}(G, G', e)$.*

Lemma 4.15 (Next Prefix). *Given two executions G_s and G_t s.t. $G_\emptyset \sqsubseteq G_s \sqsubset G_t$, and an event $e \in \mathsf{avail}(G_s) \cap G_t.\mathsf{E}$, either there exists a step $t = e\ @\_$ and an execution G' s.t. $G_s \xrightarrow{t} G' \sqsubseteq G_t$, or one of the following hold:*

- *$e \in \mathsf{W}$ and there is a step $t = e\ @\_$ and executions G' and G_r s.t. $G_s \xrightarrow{t} G' \sqsubseteq G_r$, $full(G_r)$,* $\mathsf{consistent}(G_r)$, *and* $\mathsf{racy}(G_r)$.
- *$e \in \mathsf{R}$ and there is an execution G' s.t. $\textsc{PrefixExtension}(G_s, G_t, \mathtt{event}(t)) \xrightarrow{t}\xrightarrow{e\ @p} G' \sqsubseteq G_t$, where $\langle t, p\rangle = \textsc{GetRev}(G_s, G_t, e)$, and there is no execution G'' s.t. $e \in G''.\mathsf{E}$ and $G_s \sqsubset G'' \sqsubset G'$.*

4.4 Completeness

Lemma 4.16. *Given a consistent, full execution G_f of P such that $G_f.\mathsf{BR} = G_f.\mathsf{MBR}$, and every confirmation CAS reads from the same write as the preceding speculative read, it is $G_\emptyset \sqsubseteq G_f$.*

Lemma 4.17. *Given a consistent, full execution $G_f \sqsupseteq G_\emptyset$, a call to $\textsc{ProductionSequence}(\emptyset, G_f)$ will return a production sequence S such that either $\textsc{Apply}(S) = G_f$ or* $\mathsf{race}(S)$.

Proof. We will prove by induction that given a production sequence S such that $\neg\mathsf{race}(S)$ and a full, consistent execution $G_f \sqsupseteq G_\emptyset$ s.t. $\textsc{Apply}(S) \sqsubseteq G_f$, a call to $\textsc{ProductionSequence}(S, G_f)$ will return a production sequence S' and either $\textsc{Apply}(S) = G_f$ and $S' = S$, or $\textsc{Apply}(S) \neq G_f$, S' extends S, and $\textsc{Apply}(S') \vee \mathsf{race}(S')$. Then the lemma follows immediately by setting $S = \emptyset$.

The induction is on the length of S. The base case is $\textsc{Apply}(S) = G_f$ (there is no extension of S), which is trivial.

Algorithm 1 Production sequence

procedure ProductionSequence(S, G_f)
  while Apply(S) $\neq G_f$ **do**
    $S \leftarrow$ GetNext(S, G_f)
    if race(S) **then return** S
  return S

procedure GetNext(S_0, G_t)
  $G_s \leftarrow$ Apply(S_0)
  $e \leftarrow \mathsf{next}_P(G_s)$
  if racy(G_s, e) **then return** $S_0 +\!\!+ R(e)$
  if $\exists t = (e\,@p).\, G_s \xrightarrow{t} G' \wedge G' \sqsubseteq G_t$ **then return** $S_0 +\!\!+ \underset{nr}{\overset{t}{\Rightarrow}}$
  if $e \in \mathtt{W}$ **then**
    Let $G'', \hat{G}$ be executions s.t. $G_s \xrightarrow{e\,@p} G'' \sqsubseteq \hat{G}$, *full*($\hat{G}$), and racy($\hat{G}$)
    return ProductionSequence($S_0 +\!\!+ \underset{nr}{\overset{e\,@p}{\Rightarrow}}, \hat{G}$)
  $r_0 \leftarrow \mathsf{next}_P(G_s)$
  $\langle e_0\,@p, p_{r_0}\rangle \leftarrow$ GetRev(G_t, r_0)
  $G_p \leftarrow$ PrefixExtension(G_s, G_t, e_0)
  $A \leftarrow \emptyset$
  $S \leftarrow S_0$
  while true **do**
    $G \leftarrow$ Apply(S)
    $e \leftarrow \mathsf{next}_P(G)$
    if availPrefix$(G, e) \sqsubseteq G_p \wedge e \in G_p.\mathsf{E} \cup \{e_0\}$ **then**
      if $e = e_0$ **then**
        return $S +\!\!+ \underset{rv\ r_0}{\overset{e_0\,@p}{\Rightarrow}}$
      else
        $A \leftarrow A +\!\!+ e$
    $G_{MC} \leftarrow$ MaxCompletion(G_p, A)
    $S \leftarrow$ GetNext(S, G_{MC})
    if race(S) **then return** S

procedure GetRev(G_s, G_t, r)
  if $G_t.\mathtt{rf}(r) \in \mathtt{W}^{\mathsf{conf}} \wedge \mathsf{spec}(r, G_t.\mathtt{rf}(G_t.\mathsf{spec}(G_t.\mathtt{rf}(r))))$ **then**
    $r' \leftarrow G_t.\mathsf{spec}(G_t.\mathtt{rf}(r))$
    return $\langle r'\,@G_t.\mathtt{rf}(r'), \bot\rangle$
  else

    $w_t \leftarrow \max_{G.\mathtt{co}} \{w \in G.\mathtt{W}_{\mathtt{loc}(r)} \mid \langle w, G.\mathtt{rf}(r)\rangle \in G.\mathtt{co}? \wedge \mathsf{block}(r, G.\mathtt{co}|_{\mathsf{imm}}{}^{-1}(w))\}$
    $G_p \leftarrow$ PrefixExtension(G_s, G_t, w_t)
    $w_p \leftarrow \max_{G_p.\mathtt{co}}\{w \in G_p.\mathtt{W}_{\mathtt{loc}(w_t)}\}$
    return $\langle w_t\,@w_p, w_t\rangle$

For the inductive step, we assume a production sequence S such that $\neg$race(S), $G_\emptyset \sqsubseteq$ Apply(S) $\sqsubset G_f$, *full*(G_f), and consistent(G_f), and prove that ProductionSequence(S, G_f) will return a production sequence S' that extends S such that either Apply(S') $= G_f$ or race(S'). The inductive hypothesis states that this implication holds for any production sequence S'' that is longer than S.

To prove the inductive step, we will show that given a production sequence S such that $\neg\text{race}(S)$ and a consistent execution G_t s.t. $G_\emptyset \sqsubseteq G_s \sqsubset G_t$ and $\text{next}_P(G_s) \in G_t.\mathsf{E}$, where $G_s \triangleq \textsc{Apply}(S)$, a call to $\textsc{GetNext}(S, G_t)$ returns a production sequence S' that extends S such that either $\text{race}(S')$ or $\neg\text{race}(S')$, $G_s \sqsubset \textsc{Apply}(S') \sqsubseteq G_t$ and $S' \setminus S$ does not revisit events of G_s. The inductive step then follows immediately: as long as a race is not detected (i.e., $\neg\text{race}(S)$), the calls to $\textsc{GetNext}(S, G_f)$ repeatedly grow S until it leads to G_f. The precondition $G_\emptyset \sqsubseteq \textsc{Apply}(S) \sqsubseteq G_f$ holds because for a production sequence $S' = \textsc{GetNext}(S, G_f)$ such that $\neg\text{race}(S')$, it is $\textsc{Apply}(S) \sqsubset \textsc{Apply}(S') \sqsubseteq G_f$. The precondition $\text{next}_P(G) \in G_f$ holds trivially for any full execution G_f and $G \sqsubseteq G_f$.

We will prove the claim by induction on the size of the arguments at each $\textsc{GetNext}(S, G_t)$ call, with $|\langle S, G_t\rangle| = \langle -|\textsc{Apply}(S).\mathsf{E}|, |G_t, \mathsf{E}|\rangle$. Since G_s and G_t are consistent executions of a program with finite-size executions and $G_s \sqsubset G_t$, this measure is bounded below.

Consider a call $\textsc{GetNext}(S_0, G_t)$, where S_0 is a production sequence such that $\neg\text{race}(S_0)$, G_t is a consistent execution such that $G_\emptyset \sqsubseteq G_s \sqsubset G_t$, and $\text{next}_P(G_s) \in G_t.\mathsf{E}$, where $G_s \triangleq \textsc{Apply}(S_0)$.

If the test in Line 9 succeeds, then $\textsc{GetNext}S, G_t$ will return $S' = S_0 +\!\!+ R(e)$, which extends S_0 and $\text{race}(S')$. Otherwise, it is not $\neg\text{racy}(G_s, e)$.

If the test in Line 10 succeeds, we only need to show that if $G_s \xrightarrow{e\,@p} G'$, then $G_s \underset{nr}{\overset{e\,@p}{\Rightarrow}} G'$. If $e \in \mathsf{W}$, then $\textsc{Visit}_P(G)$ does consider every co placement p. If $e \in \mathsf{R}$, then the only rf choices not considered by the algorithm are the $\bot$, and writes w such that there is completed same-location speculative read that reads from w (if e is a speculative read). The latter cannot be the case by definition of $\rightarrow$. For the former, assume that $G_s \xrightarrow{e\,@\bot} G'$. Since $e = \text{next}_P(G_s)$, there cannot exist any same-location speculative read r' that is not completed and is not reading $\bot$. But is also cannot be that there is an r' that reads $\bot$: it is $G_\emptyset \sqsubseteq G_s$ and thus when r' was added to read $\bot$ in this $\rightarrow$ sequence, there was a same-location speculative r'' read that was reading from a write $w \neq \bot$. Since all speculative reads that do not read $\bot$ are completed in G_s, this r'' must be followed by a confirmation write, which in-place revisited r'. Therefore, all speculative reads in G_s are completed, contradicting that $G_s \xrightarrow{e\,@\bot} G'$.

If the test in Line 10 does not succeed, from Lemma 4.15 there are two cases. The first case is that $e \in \mathsf{W}$ and there is a step $t = e\,@\_$ and executions G' and G_r such that $G_s \xrightarrow{t} G' \sqsubseteq G_r$, $\mathit{full}(G_r)$, $\text{consistent}(G_r)$, and $\text{racy}(G_r)$. Obviously, $\Rightarrow$ can match this $\rightarrow$ step and thus from the inductive hypothesis on the ProductionSequence calls (the first argument is at least one step longer), we have that Line 13 returns an extension S' of the first argument such that either $\textsc{Apply}(S') = \hat{G}$, or $\text{race}(S')$. The former cannot happen, since $\hat{G}$ is racy, and thus a race would have been detected. Therefore, $\textsc{GetNext}S_0, G_t$ in Line 13 will return an extension S' of S_0 such that $\text{race}(S')$. The second case is that $e \in \mathsf{R}$ and there is an execution G' s.t. $\textsc{PrefixExtension}(G_s, G_t, \mathsf{event}(t)) \xrightarrow{t} \xrightarrow{e\,@p} G_t$, where $\langle t, p\rangle = \textsc{GetRev}(G_s, G_t, e)$.

From Lemma 4.14, we have that $G_s \sqsubseteq G_p \sqsubseteq G_t$. We will show that in every iteration before the one where the test in Line 23 succeeds, either an extension S' of S_0 is returned such that $\text{race}(S')$ (Line 29), or if S' and A' are the new values of S and A, respectively, at the end of the loop, $G \triangleq \textsc{Apply}(S)$, and $G' \triangleq \textsc{Apply}(S')$, then $\textsc{MaxCompletion}(G_p, A') \neq \bot$, $G \sqsubset G' \sqsubseteq \textsc{MaxCompletion}(G_p, A')$, $A' \subseteq G'.\mathsf{E}$, and no event of G_s is revisited by $S' \setminus S_0$

In the first iteration, it is $G = G_s$, $\text{next}_P(G) = r_0$, $A = \emptyset$. By construction of G_p it is $G \sqsubseteq G_p$. The test in Line 22 will fail because $r_0 \notin G_p$, and thus it will be $A' = [r_0]$. From $r_0 \in \mathsf{nextCandidates}(G_s)$ and $G_s \sqsubseteq G_p$, we have $r_0 \in \mathsf{nextCandidates}(G_p)$: G_p can only have more enabled events (caused by an in-place revisit). From Lemmas 4.11 and 4.12, there is an execution G_{MC} such that $G \overset{r_0}{\rightsquigarrow} G_{MC}$. The first argument in the recursive call of Line 28 is S_0 and the second (G_{MC}) has at least one

fewer event than G_t (e_0). Since $r_0 \in G_{MC}$, we can use the inductive hypothesis and get that for the resulting extension S' of S_0, it is either $\mathsf{race}(S')$, in which case $\mathsf{next}_P(S_0, G_t)$ returns S' and fulfills its postcondition, or $\neg\mathsf{race}(S')$, $G \sqsubset G' \sqsubseteq G_{MC}$ and no step in $S' \setminus S_0$ revisits an event of G_s. Since $G' \sqsupset G$, $\mathsf{next}_P(G) = r_0$, and no event of G is revisited by $S' \setminus S_0$, it will be $r_0 \in G'.\mathsf{E}$ and thus $A' \subseteq G'.\mathsf{E}$.

For any next iteration, let S be the initial value of S and S' the value at the end of the iteration. Again, if a race is detected, the an extension S' of S_0 is returned which fulfills the postcondition of $\textsc{GetNext}S_0, G_t$. Otherwise, let $G \triangleq \textsc{Apply}(S)$, and $G' \triangleq \textsc{Apply}(S')$. From the previous iteration we have $G \sqsubseteq \textsc{MaxCompletion}(G_p, A)$, $A \subseteq G.\mathsf{E}$, and $S \setminus S_0$ does not revisit events of G_s. Additionally, since $G \sqsubseteq \textsc{MaxCompletion}(G_p, A)$, it is also $\mathsf{next}_P(G) \neq \bot$.

If the test in line Line 22 succeeds, we get that $\textsc{MaxCompletion}(G_p, A') \neq \bot$ from the previous iteration ($A' = A$). Otherwise, is suffices to show that for the execution $G_{MC} = \textsc{MaxCompletion}(G_p, A)$, it is $e \in \mathsf{nextCandidates}(G_{MC})$. Assume that $e \notin \mathsf{avail}(G_{MC})$. Since $e \in \mathsf{avail}(G)$ and $G \sqsubseteq \textsc{MaxCompletion}(G_p, A)$, it is $e \in G_p$: it cannot be that $e \in A$: every time an event e is added to A, in Line 28 we get an execution that includes e and hence it cannot be that $e \in \mathsf{avail}(G'')$ for a later G''. Since $G \sqsubseteq \textsc{MaxCompletion}(G_p, A)$ and $e \in \mathsf{avail}(G)$, it is $\mathsf{availPrefix}(G, e) \sqsubseteq \textsc{MaxCompletion}(G_p, A)$. Together with $e \in G_p$, we get that $\mathsf{availPrefix}(G, e) \sqsubseteq G_p$, which contradicts the fact that the test in line Line 22 fails. Thus, $e \in \mathsf{avail}(G_{MC})$. Assume that $e \notin \mathsf{nextCandidates}(G_{MC})$. Then is must be that $e \in \mathsf{R}_l^{\mathsf{spec}}$, and there are is a speculative read $r_b \in \mathsf{R}_l^{\mathsf{spec}}$ that is not completed. This contradicts that $e \in \mathsf{nextCandidates}(G)$ since no speculative read event in $G_p.\mathsf{E} \setminus G.\mathsf{E}$ can be not completed.

We thus have $\textsc{MaxCompletion}(G_p, A) \neq \bot$ in both cases. Since the first iteration is already executed, we have $G_s \sqsubset G$. Using the inductive hypothesis in Line 28 we get that $G \sqsubset G' \sqsubseteq \textsc{MaxCompletion}(G_p, A')$ and $S' \setminus S$ does not revisit events of G, and thus of G_s ($G_s \sqsubset G$). To see that the remaining invariant, i.e., $A' \subseteq G'.\mathsf{E}$, holds, we consider again two cases, depending on whether the test in Line 22 succeeds. If the test succeeds, we also get that $A' \subseteq G'.\mathsf{E}$, since $A' = A \subseteq G.\mathsf{E} \subseteq G'.\mathsf{E}$ ($G \sqsubset G'$). If the test fails, it suffices we show that $e \in G'.\mathsf{E}$. This follows from $\mathsf{next}_P(G) = e$, $G' \sqsupset G$, and no event of G is revisited by $S' \setminus S$.

If a race is not detected, the loop will eventually terminate by the call in Line 24. To see this, observe that at each iteration, G' is a consistent execution of the program that has at least one more event than the previous iteration ($G \sqsubset G'$). From the success of the test in Line 23, $G \sqsubseteq \textsc{MaxCompletion}(G_p, A)$, and $A \subseteq G.\mathsf{E}$, we have that $G = \textsc{MaxCompletion}(G_p, A)$ and $\mathsf{next}_P(G) = e_0$. From the invariants of the loop, we have that the events of A appear in the sequence in the same order as they were added (once added they are never deleted). The events that would be affected by a revisit step of e_0 to r_0 are the events of $\textsc{MaxCompletion}(G_p, A)$ that are not in G_p, thus the events of A. Therefore, that revisit step $\overset{e_0\,@p}{\underset{rv\ r_0}{\Rightarrow}}$ will be enabled and for the resulting execution G' we have $G_\emptyset \sqsubseteq G_s \overset{e_0\,@p r_0\,@p_{r_0}}{\longrightarrow} G' \sqsubseteq G_t$. Finally, the revisit step does not revisit any event of G_s ($r_0 \notin G_s$). □

4.5 Optimality

Lemma 4.18. *Given consistent execution G and G' s.t. $G_\emptyset \Rightarrow^* G \overset{e}{\underset{rv\ r}{\Rightarrow}} G'$, it is $G.\mathtt{rf}(r) \in G'.\mathsf{E}$.*

Proof. If e is a speculative read that revisited r, then in G' it is reading from $G.\mathtt{rf}(r)$ by definition of the revisit step. Otherwise, e is a write event. If e can also be added with a maximal step consistently, then it must be that $\neg\mathsf{racy}(G, e)$, since $G \Rightarrow G'$, which implies that all same-location writes, including $G.\mathtt{rf}(r)$ are in the $\mathtt{porf}$-prefix of e, and thus it would be $G.\mathtt{rf}(r) \in G'.\mathsf{E}$. If e

cannot be added with a maximal step, it must be a write that is part of an RMW that reads from a write w that is already read by another completed RMW $\langle r', w' \rangle$. Since G' is consistent, it must be that the revisit step either revisits or deletes r'. In the first case, i.e., $r = r'$, it is obvious that $w = G.\mathtt{rf}(r)$ is in $G'.\mathsf{E}$. In the second case, assume that $G.\mathtt{rf}(r) \notin G'.\mathsf{E}$. Then, it must be that the write that r is reading from in G was removed by a series of $\rightsquigarrow$, and thus in the execution $\hat{G}$ that results by removing the events of the deleted set, r is blocking. However, this contradicts that the revisit step takes place. □

Lemma 4.19. *Let $\langle G, < \rangle$ be a configuration s.t. $\langle G_\emptyset, \emptyset \rangle \Rightarrow^* \overset{e\ @-}{\underset{rv\ -}{\Rightarrow}} \langle G, < \rangle$ for some event e. Then, there is no algorithmic step after $\langle G, < \rangle$ that revisits or deletes e.*

Proof. Let S the production sequence of an execution G, i.e., $G = \textsc{Apply}(S)$, such that $\neg\mathsf{race}(S)$. We define $rv(S)$ as the set of events in G that revisited an event, and $mrv(S)$ the subset of $rv(S)$ of events that revisited an event that was not revisited or deleted later.

We will first prove by induction that given a production sequence $S +\!\!+ t$, t step cannot revisit or delete an event of $mrv(S)$. The base case is $S = \emptyset$, which is trivial. For the inductive case, we will prove that there is no step t that revisits or deletes an event of $mrv(S)$. Let $G = \textsc{Apply}(S)$, $e = \mathsf{next}_P(G)$, and t a step from S that revisits an event r and revisits or deletes an event $b \in mrv(S)$. For the event a that b revisited, it is $a < b$.

The first case is that b is a speculative read event, and thus a is another speculative read event. From the inductive hypothesis, b was not revisited or deleted after it was added and revisited a, and thus it is reading from the write that a was reading before the revisit from b. In execution G, b is completed: it does not read $\bot$, and for $\mathsf{next}_P(G)$ to pick another speculative read is for b to be followed by the matching confirmation write. Therefore, in execution G, b is completed and a reads from the matching confirmation CAS w of b. Because the revisit step t is enabled, there is an execution $\hat{G}$ s.t. $\hat{G} \rightsquigarrow^* G_b \overset{b}{\rightsquigarrow} G'_b \rightsquigarrow^+ G_w \overset{w}{\rightsquigarrow} G'_w \rightsquigarrow^* G$. In G'_w, a reads from w, since it does so in G and no $\rightsquigarrow$ would change this $\mathtt{rf}$ edge. In G_w, a reads $\bot$ since $G_w \overset{w}{\rightsquigarrow} G'_w$, and a reads from w in G'_w. Event a reads $\bot$ in G'_b, since it does so in G_w and no $\rightsquigarrow$ can change this $\mathtt{rf}$ edge. This leads to contradiction, since $G_b \overset{b}{\rightsquigarrow} G'_b$ but in G_b there is a same-location speculative read a that reads $\bot$.

The second case is that b is a write event, and thus a is a read event. Let G_1 and G_2 be the executions before and after the revisit step t' of b to a, i.e., $G_1 \overset{b}{\underset{rv\ a}{\Rightarrow}} G_2$. From Lemma 4.18, it is $\hat{w} \triangleq G_1.\mathtt{rf}(a) \in G_2.\mathsf{E}$, and since G_2 is not racy, it is $\langle \hat{w}, b \rangle \in G_2.\mathtt{porf}$. From the induction hypothesis, b is in G and a has not been revisited. Since G_2 is not racy, all writes W_b of G_2 to $\mathtt{loc}(b)$ are in the $\mathtt{porf}?$-prefix of b in G_2, and thus the writes in W_b are also in the $\mathtt{porf}?$-prefix of b in G. Additionally, since the revisit step t' was enabled from G_1, $\hat{w}$ is the $\mathtt{co}$-maximal of the writes in $W_b \setminus \{b\}$ in G_2. It is easy to see that there are no other writes in the $\mathtt{porf}$-prefix of b in G apart from those in W_b, since they would have to revisit an event in the $\mathtt{porf}$-prefix of b, and thus delete b. Execution G is also not racy, and thus a reads from a write b' that is $\mathtt{porf}?$-after b (a might have been blocked by reading from b, and was in-place revisited later). The revisit step t is enabled in G, and hence there is an execution $\hat{G}$ s.t. $\hat{G} \rightsquigarrow^* G_b \overset{b}{\rightsquigarrow} G'_b \rightsquigarrow^* G$. Execution G'_b includes a; otherwise, a is in the $\mathtt{porf}$-prefix of e, and hence the same holds for b since it is $\langle b, a \rangle \in \mathtt{porf}?; [b']; \mathtt{rf}$. Any write $\mathtt{porf}$-after b is not in G'_b, and thus $G'_b.\mathtt{rf}(a) = b$. From $G_b \overset{b}{\rightsquigarrow} G'_b$, we have that b is blocking by reading from the $\mathtt{co}$-maximal write of G_b. It is easy to see that the writes of G_b to $\mathtt{loc}(b)$ are exactly the writes in $W_b \setminus \{b\}$, and a does not block when reading from the $\mathtt{co}$-maximal write $\hat{w}$ from this set. Therefore, we reached a contradiction.

Having proved that no event of $mrv(S)$ can be revisited or deleted, we will show that the same holds for $rv(S)$ by proving that every event of $rv(S)$ is either in $mrv(S)$ or in the porf-prefix of an event in $mrv(S)$; in the latter case the event of $rv(S)$ cannot be revisited or deleted because an event of $mrv(S)$ would also have to be revisited or deleted. We will prove our claim by induction on the production sequence S. The base case is $S = \emptyset$, which is trivial.

Let $G = \textsc{Apply}(S)$, $e = \mathsf{next}_P(G)$, t a step from S, G' the resulting execution, and b an event of $rv(S)$. Assume that t is a non-revisit step. The first case is $b \in mrv(S)$. Then it follows that it is also $b \in mrv(S +\!\!+ t)$. The second case is $\langle b, e' \rangle \in G.\mathsf{porf}$, for an event $e' \in mrv(S)$. Even if t includes an in-place revisit, it is easy to see that $\langle b, e' \rangle \in G'.\mathsf{porf}$. In both cases $b \in mrv(S +\!\!+ t)$ or b is in the $G'.\mathsf{porf}$-prefix of an event in $mrv(S +\!\!+ t)$.

Assume now that t is a revisit step $\underset{rv\ r}{\overset{e}{\Rightarrow}}$. The first case is $b \in mrv(S)$. We will show that if $b \notin mrv(S +\!\!+ t)$, it is $\langle b, e \rangle \in G'.\mathsf{porf}$. To see this, observe that b revisited an event $a \in G.\mathsf{E}$ that was either revisited or deleted by t. Since $a < b$ in G, but b was not deleted (events of $mrv(S)$ cannot be revisited or deleted), b is in the porf-prefix of e in G', and $e \in mrv(S +\!\!+ t)$. The second case is $\langle b, b' \rangle \in G.\mathsf{porf}$, for an event $b' \in mrv(S)$ that revisited $a' \in G$. If $b' \in mrv(S +\!\!+ t)$, it must be that a' was either revisited or deleted by t. For the same reason as in the previous case, b' must be in the porf-prefix of e in G', and thus it is also that b is in the porf-prefix of e in G'. □

Lemma 4.20. *Given an execution G reachable by the algorithm, G has at most one pending read event.*

Proof. Given a configuration $\langle G, < \rangle$ reachable by the algorithm, we will prove by induction on the length of the production sequence S of $\langle G, < \rangle$ that (1) G has at most one pending read event and it was added or revisited in the exact previous step; and (2) if there are events $r, w, e \in G.\mathsf{E}$ such that $\langle r, w \rangle \in G.RMW$, $e \in \mathsf{R}$, and $r < e < w$, then e is in the porf prefix of an event of G that revisited in S.

The base case is the initial execution $G_\emptyset$, which is trivial. For the inductive case, we will prove that if $\langle G_\emptyset, \emptyset \rangle \Rightarrow^* \langle G, < \rangle \underset{t}{\overset{e}{\Rightarrow}} \langle G', <' \rangle$ and every configuration up to $\langle G, < \rangle$ satisfies properties (1) and (2), then so does $\langle G', <' \rangle$.

We first consider the case where $t = nr$. If G has no pending read, then the added event is not a write that is a part of an RMW, and thus condition (2) holds for $\langle G', <' \rangle$. Condition (1) also holds since G' can have at most one pending read: the event e that is just added. If G has a pending read a, then e is the matching write, by definition of $\mathsf{next}_P(G)$. Thus, execution G' has no pending read and condition (1) holds. Note that there are no await CASes, and thus adding a write cannot introduce any pending reads. It remains to show that if G has a pending read a, condition (2) holds as well. From the inductive hypothesis, a was just added or revisited in G. If a was just added, then condition (2) holds trivially by the inductive hypothesis. If a was just revisited, then any event $<'$-after a is the porf-prefix of the write that revisited, and condition (2) still holds.

The other case is $t = rv\ a$. We will show that, if G' has a pending read, it must be a. If G has a pending read, then e is the matching RMW. Thus, is suffices to show that the only pending read of G' can be the read a. Suppose that this is not the case. Since there are no await CASes, the only way for G' to have a pending read is by deleting the write of an RMW, but not the read, i.e., there are event $r, w \in G.\mathsf{E}$ s.t. $\langle r, w \rangle \in G.RMW$, $r \neq a$, and G' contains r but not w. If r was in the prefix of e, then so would w. Therefore, r is not in the prefix of e, but instead $r < a$. Since w was deleted, it is $a < w$. From condition (2) for G, a is in the prefix of a event that revisited, which contradict that a is revisited in t (Lemma 4.19). Therefore condition (1) holds for G', since it has at most one pending read. Condition (2) also holds since we showed that if there is a pending read in G', it must be a, which was the revisited event in t. □

Lemma 4.21. *Let S be a a production sequence such that* $\neg\mathsf{race}(S)$ *and* $\textsc{Apply}(S) = G_t$*. Then there is no execution G in S s.t.*

- *G results from a $\overset{e}{\underset{rv\ r}{\Rightarrow}}$ step*
- $G \setminus \{r\} \sqsubseteq G_t$
- $e \in \mathtt{W} \Rightarrow \langle G_t.\mathtt{rf}(r), e\rangle \in G.\mathtt{porf}$
- $e \in \mathsf{R}^{\mathsf{spec}} \Rightarrow G_t.\mathtt{rf}(r) = G.\mathtt{rf}(e)$

Proof. Assume the opposite and let G be the last such execution in S. In both cases, $G_t.\mathtt{rf}(r)$ is in the porf-prefix of e, and thus it cannot be deleted in a later step (Lemma 4.19). The only way for G_t to be reached from G is for another read of G to be revisited so that r is deleted. Let G' be the first execution in S after G that does not include r, resulting from a step t that revisits a read r' of an execution $\hat{G}$ by an event e', i.e., $\hat{G} \overset{e'}{\underset{rv\ r'}{\Rightarrow}} G'$. In G, all events $<$-after r are in the porf-prefix of e, thus the only events of G that can be revisited are $<$-before r or r itself. Additionally, no events of G are deleted until $\hat{G}$. Since t is the first step that deletes r ($r' < r$), the are no revisits to events of G between G and $\hat{G}$ in S: any other revisit by an event a would have to not delete r, and thus a must be porf-after r, which would imply that r cannot be deleted later, contradicting the assumption about t deleting r. Let $w_t = G_t.\mathtt{rf}(r')$ and $\hat{w} = \hat{G}.\mathtt{rf}(r')$.

Assume that $e' \in \mathtt{W}$. From $G \setminus \{r\} \sqsubseteq G_t$, we have that either $G.\mathtt{rf}(r') = w_t$, or r' reads from a write w' in G and blocks, and all writes porf-after w' and before $w_t = G_t.\mathtt{rf}(r')$ in G_t block r', so that it can be in-placed revisited by all of them and read from w_t in G_t. From Lemma 4.18, it is $\hat{w} \in G'.\mathtt{E}$, and since G' is not racy, it is $\langle \hat{w}, e'\rangle \in G'.\mathtt{porf}$.

Let $w' \triangleq G.\mathtt{rf}(r')$. Since no revisit happened from G to $\hat{G}$ to the events of $G.\mathtt{E}$, $\hat{G}.\mathtt{rf}(r') = \hat{w}$ is either w', or a write porf-after w' (otherwise $\hat{G}$ is racy). From Lemma 4.18, $\hat{w}$ is in $G'.\mathtt{E}$, and thus it is porf-before e' (otherwise G' is racy). Therefore, $\langle w', e'\rangle \in G'.\mathtt{porf}$. Assume that $w' \neq w_t$. Since r' was reading from $\hat{w}$ in $\hat{G}$, $\hat{w} \in G'.\mathtt{E}$, and the revisit step was taken from $\hat{G}$ to G', $\hat{w}$ was co-maximal in $\hat{G}$ and did not block r'. Since G' eventually leads to G_t, Lemma 4.19 implies that G_t also includes e' and its porf-prefix. If $w_t \in G'$, then $\langle w_t, e'\rangle \in G'.\mathtt{porf}$. Otherwise, w_t is in the porf-suffix of e' in G_t, and we have thus reached a contradiction: $G \setminus \{r\}G_t$, $G.\mathtt{rf}(r') = w'$, but the writes in the porf path from w' to w_t in G_t do not all block r' ($\hat{w}$ does not). Therefore, we have $\langle w_t, e'\rangle \in G'.\mathtt{porf}$.

Assume now that $e' \in \mathsf{R}^{\mathsf{spec}}$, which implies $r' \in \mathsf{R}^{\mathsf{spec}}$. From $G \setminus \{r\}G_t$, we have that $w_t = G.\mathtt{rf}(r')$. Since no events of G are deleted or revisited until $\hat{G}$, it is $w_t = \hat{w}$. Since r' is revisited from e', it is $G'.\mathtt{rf}(e') = w_t$, which gives us that $\langle w_t, e'\rangle \in G'.\mathtt{porf}$.

To reach a contradiction, it remains to be shown that $G' \setminus \{r'\} \sqsubseteq G_t$, which would imply that G' is an execution after G in S that also satisfies the properties in question. To see that $G' \setminus \{r'\}$, notice that $G \setminus \{r\} \sqsubseteq G_t$, no event of G is revisited or deleted until $\hat{G}$, G' does not contain r, and all events in G' that are not in G also exist in G_t, since they are events in the porf?-prefix of e', e' cannot be deleted by a later step in S, and S reaches G_t.

We have now reached a contradiction: G' is an execution after G in S that results from a step $\overset{e'}{\underset{rv\ r'}{\Rightarrow}}$ and satisfies the respective properties. □

Lemma 4.22. *Let S be a production sequence that reaches a consistent execution G_t from $G_\emptyset$, and G be an execution in S such that $G \sqsubseteq G_t$. Then there is no algorithm step in S after G that revisits an event of G.*

Proof. Assume the opposite and take t to be the first step after G in S that revisits a read r of G. Let $\hat{G}$ and G' be the executions before and after t, respectively, i.e., $G \Rightarrow^* \hat{G} \overset{e\ @-}{\underset{rv\ r}{\Rightarrow}} G'$.

Assume that $e \in \mathtt{W}$, and let $\hat{w} = \hat{G}.\mathtt{rf}(r)$. From Lemma 4.18, $\hat{w}$ is also in G', and thus in the $\mathtt{porf}$-prefix of e (G' is not racy). Since $G \sqsubseteq G_t$, r is either reading from $w_t = G_t.\mathtt{rf}(r)$ in G, or is reading from a $\mathtt{co}$ maximal write $w_b \neq w_t$ and blocks. In the first case, assume that $w_t \neq \hat{w}$. This implies that r blocks by reading from w_t and was in-place revisited by the $\mathtt{co}$-later $\hat{w}$. From Lemma 4.19 and the fact that S reaches G_t, $\hat{w}$ also exists in G_t, and therefore in G_t r reads from a write w_t that blocks it but it is not $\mathtt{co}$-maximal, which is a contradiction. Therefore, we have $w_t = \hat{w}$ in the first case. In the second case, in $\hat{G}$ r is also reading from a $\mathtt{co}$-maximal write $\hat{w}$, (either w_b, or a $\mathtt{co}$-later write that in-placed revisited r). Since the revisit step t is enabled, r is not blocking in $\hat{G}$. Execution G' includes $\hat{w}$ (Lemma 4.18), and is in the $\mathtt{porf}$-prefix of e (G' is not racy). Execution G_t can be reached from G' and e must also exist in G_t. Therefore, it must be $\hat{w} = w_t$: it is $G \sqsubseteq G_t$, and $\hat{w}$ is the $\mathtt{porf}$-last write that r can be in-place revisited from, when it initially reads from a write w_b in G that blocks r. We thus have $\langle G_t.\mathtt{rf}(r), e\rangle \in G'.\mathtt{porf}$ in either case.

Assume that $e \in \mathsf{R}^{\mathsf{spec}}$. Then it must be that $r \in \mathsf{R}^{\mathsf{spec}}$ and r is either reading $\bot$ or from $G_t.\mathtt{rf}(r)$ in G, since it is $G \sqsubseteq G_t$. In the former case, since execution G is reachable from $G_\emptyset$, and thus $G_\emptyset \sqsubseteq G$, G contains a speculative read r_s in the same location as r that has no matching confirmation CAS, and in G_t, r reads from this confirmation CAS w_t, i.e., $G_t.\mathtt{rf}(r) = w_t$. Execution $\hat{G}$ also contains r_s, and since $\mathsf{next}_P(\hat{G}) = e \in \mathsf{R}^{\mathsf{spec}}$, the matching confirmation CAS w_t of r_s is in $\hat{G}$ and r reads from w_t, i.e., $\hat{G}.\mathtt{rf}(r) = w_t$. In the latter case, r continues reading from w_t in $\hat{G}$. In both cases, $\hat{G}.\mathtt{rf}(r) = G_t.\mathtt{rf}(r)$, and since t revisits r from e, it is $G'.\mathtt{rf}(e) = G_t.\mathtt{rf}(r)$.

Finally, it is easy to see that $G' \setminus \{r\} \sqsubseteq G_t$: the $G'.\mathtt{porf}$-prefix of e is included in G_t, $G \sqsubseteq G_t$, and no event of G is revisited in S between G and $\hat{G}$. We have now reached a contradiction due to Lemma 4.21. □

Theorem 4.23. *Given a full, consistent execution $G_f \sqsupseteq G_\emptyset$, there is at most one production sequence that reaches G_f.*

Proof. We will show that if there is a production sequence that leads to G_f, then this production sequence can only be ProductionSequence$(\emptyset, G_f)$. We will prove by induction (with the same measure as in Lemma 4.17) that given a production sequence S that reaches G_s (i.e., $G_s = \textsc{Apply}(S)$) and a consistent execution G_t s.t. $G_\emptyset \sqsubseteq G_s \sqsubset G_t$ and $\mathsf{next}_P(G_s) \in G_t.\mathsf{E}$, any production sequence that extends S and reaches an execution $\hat{G} \sqsupseteq G_t$, (1) does not delete the first event added (i.e., $\mathsf{next}_P(G_s)$) and (2) must be a suffix of GetNext(S, G_t). Then our result follows by induction on the number of iterations of the loop in ProductionSequence$(\emptyset, G_f)$. If at any step $\mathsf{next}_P(G_s)$ returns a production sequence such that $\mathsf{race}(S)$, there no production sequence that reaches G_f exists (since it cannot be S). Termination of the loop is proven in Lemma 4.17.

Consider a production sequence S_0 and a consistent execution G_t such that $G_\emptyset \sqsubseteq G_s \sqsubset G_t$ and $\mathsf{next}_P(G_s) \in G_t.\mathsf{E}$, where $G_s = \textsc{Apply}(S_0)$. Also consider a production sequence $\hat{S}$ that extends S_0 and reaches an execution $\hat{G} \sqsupseteq G_t$. The test in Line 9 cannot succeed, otherwise G_t is racy and thus there is no production sequence that reaches $\hat{G}$. If the test in Line 10 succeeds, it must be that $\hat{S} \setminus S_0$ starts with the $\overset{t}{\underset{nr}{\Rightarrow}}$. Otherwise, to reach an execution $\hat{G} \sqsupset G_s$, $\hat{S}$ must revisit an event of G_s. To see this, consider the case where the added event is a read and assume that another step t' is taken and examine two cases, depending on whether $w \triangleq G_t.\mathtt{rf}(r)$ is in $G_s.\mathsf{E}$. If $w \in G_s.\mathsf{E}$, t is the step that adds r to read from w. A step t' would add r to read from a $w' \neq w$ and thus a later step needs to delete r or w', which means that an event of G_s is revisited. If $w \notin G_s.\mathsf{E}$, it cannot be that $r \in \mathsf{R}^{\mathsf{spec}}$: no read option for r will allow it to read from another write, without a revisit. Hence $r \notin \mathsf{R}^{\mathsf{spec}}$, t is the step that adds r to read from the $\mathtt{co}$-maximal write w_m and block. Any other step t' would either end with r reading from a non $\mathtt{co}$-maximal write and either block or not block. In the former case, the execution would be dropped. In the latter case, r cannot be eventually revisited by w since

w_m would be in the execution and thus r would not be reading from a co-maximal write. Thus an event of G_s must be revisited in $\hat{S}$, which leads to a contradiction from Lemma 4.22.

If the test in Line 10 fails, we will show that $\hat{S}$ must reach execution G_p and first follow the steps returned in Line 24. The production sequence $\hat{S}$ will first add the read event r_0. Since $G_s.\texttt{rf}(r_0) \notin G_s.\mathsf{E}$ (the test in Line 10 fails), the event r_0 will read from another write and must either be deleted and added again, or be revisited. It cannot be that r_0 eventually reads from $G_s.\texttt{rf}(r_0)$ by a series of in-place revisits, again because the test in Line 10 fails. From Lemma 4.22, it cannot be that r_0 is deleted (it would require a read of G_s being revisited). Thus r_0 will be eventually revisited, which will lead to execution G_p.

We will show that as soon as a configuration $\langle G', <' \rangle$ is reached s.t. $\mathsf{availPrefix}(G', e_0) \sqsubseteq G_p$ and $\mathsf{next}_P(G') = e_0$, $\hat{S}$ will perform a revisit step and reach G_p. Assume that this is not the case: such an execution G' is reached but $\hat{S}$ does not proceed with a revisit of r_0.

We first consider the case where $\hat{S}$ proceeds with a revisit step. Assume that e_0 is a speculative read, and thus r_0 is a speculative read. If e_0 reads from a write that is porf-after the one r_0 is reading, and thus porf after r_0 since r_0 is a completed speculative read ($\mathsf{next}_P(.)$ picks another speculative read), then r_0 will keep reading from $G'.\texttt{rf}(r_0)$ in any later execution in $\hat{S}$, since it will be in the porf-prefix of an event that revisited, which contradicts that $\hat{S}$ reaches an execution $\hat{G} \sqsupseteq G_t$. If e_0 read from a write that is porf-before the one r_0 is reading, $\hat{S}$ starts with a revisit of an event of G_s, which contradicts Lemma 4.22. The other case is that e_0 is a write w_0. If w_0 performs another revisit step, it will never be deleted in a later step (Lemma 4.19). Also, r_0 will not be deleted since no event of G_s will be revisited (Lemma 4.22), and thus $\hat{S}$ cannot reach an execution $\hat{G} \sqsupseteq G_t$. To see this, observe that r_0 does not block in G' (since the revisit to r_0 is enabled), and in the execution after a different revisit step it is not co-maximal and therefore its rf edge cannot change by an in-place revisit.

The other case is for e_0 to be added with a *nr* step and later to be deleted and eventually added again to revisit e_0. Let $\overset{e' @ -}{\underset{rv\ -}{\Rightarrow}}$ be the first revisit step to events of G' after this *nr* step. If e' is in the porf-suffix of e_0, then e_0 cannot be later deleted, and thus r_0 will never be revisited from e_0. If e' is not in the porf-suffix of e_0, then r_0 cannot be later revisited by e_0: the revisit would have to delete e', which contradicts Lemma 4.19. Note that e' cannot have been deleted in a step before the hypothetical revisit of e_0, again due to Lemma 4.19.

It remains to show that $\hat{S}$ necessarily reaches the executions in Line 28 (for each iteration), using the corresponding steps returned by the recursive call, before ending with the revisit step in Line 24. We will show that at each iteration, $\hat{S}$ must follow exactly the steps in S and the events in A cannot be deleted by a later step in $\hat{S}$. Initially this holds since $S = \emptyset$, $A = \emptyset$, and no events of G_s can be revisited (Lemma 4.22) in $\hat{S}$.

Consider an iteration of the loop. The first case is for the test in Line 22 to succeed. Then A remains the same. Since the events in A cannot be deleted by a later step in $\hat{S}$, it must be that an execution G' is reached such that $G' \sqsupseteq \textsc{MaxCompletion}(G_p, A)$; otherwise, the maximality check will fail. From the inductive hypothesis, $\hat{S}$ must match the steps returned by the recursive call. The second case is for the test in Line 22 to fail. Since the events in A (before adding e) cannot be deleted by a later step in $\hat{S}$, it must again be that an execution G' is reached such that $G' \sqsupseteq \textsc{MaxCompletion}(G_p, A)$. From the inductive hypothesis, the first event added, i.e., e, cannot be deleted by a later step in $\hat{S}$. Thus, it must be that an execution G'' is reached such that $G' \sqsupseteq \textsc{MaxCompletion}(G_p, A +\!\!+ e)$. From the inductive hypothesis, $\hat{S}$ must again match the steps returned by the recursive call.

We have thus shown that at each iteration, $\hat{S}$ matches the steps of S. We note that at no point a race can be detected, because $\hat{S}$ does reach G_t (from the hypothesis), and any $\hat{S}$ matches the steps of

S. The proof is concluded by observing that the first event added after G_s in the unique production sequence that reaches an execution $G' \sqsupseteq G_t$, i.e., r_0, is not deleted by any step in the production sequence (Lemma 4.22).

□