$$
\begin{aligned}
&[\![\texttt{skip}]\!]_{ok} \triangleq \{(s,s) \mid s \in \text{World}\} && [\![\texttt{skip}]\!]_{er} \triangleq \emptyset \\
&[\![x := e]\!]ok \triangleq \{((\eta,\sigma),(\eta[x \mapsto v],\sigma)) \mid [\![e]\!]\eta = v\} && [\![x := e]\!]_{er} \triangleq \emptyset \\
&[\![\texttt{assume}(B)]\!]_{ok} \triangleq \{(s,s) \mid s{=}(\eta,\sigma) \land [\![B]\!]\eta \neq 0\} && [\![\texttt{assume}(B)]\!]_{er} \triangleq \emptyset \\
&[\![\texttt{error()}]\!]_{ok} \triangleq \emptyset && [\![\texttt{error()}]\!]_{er} \triangleq \{(s,s) \mid s \in \text{World}\}
\end{aligned}
$$

$$
[\![\mathsf{C}_1;\mathsf{C}_2]\!]_\epsilon \triangleq \left\{(s,s') \,\middle|\, \begin{array}{l} \epsilon = er \land (s,s') \in [\![\mathsf{C}_1]\!]_\epsilon \\ \lor\ \exists s''.\ (s,s'') \in [\![\mathsf{C}_1]\!]_{ok} \land (s'',s') \in [\![\mathsf{C}_2]\!]_\epsilon \end{array}\right\}
$$

$$
[\![\mathsf{C}_1 + \mathsf{C}_2]\!]_\epsilon \triangleq [\![\mathsf{C}_1]\!]_\epsilon \cup [\![\mathsf{C}_2]\!]_\epsilon
$$

$$
[\![\mathsf{C}^\star]\!]_\epsilon \triangleq \bigcup_{i \in \mathbb{N}} [\![\mathsf{C}^i]\!]_\epsilon \quad \text{with} \quad \mathsf{C}^0 \triangleq \mathsf{skip} \quad \text{and} \quad \mathsf{C}^{i+1} \triangleq \mathsf{C};\mathsf{C}^i
$$

$$
\begin{aligned}
[\![x := \texttt{malloc()}]\!]_{ok} \triangleq\ & \left\{(s,(\eta[x \mapsto l],\sigma[l \mapsto v])) \,\middle|\, \begin{array}{l} s{=}(\eta,\sigma) \land v \in \text{Val} \\ \land\ (l \notin \text{dom}(\sigma) \lor \sigma(l){=}\bot) \end{array}\right\} \\
& \cup \{(s,(\eta[x \mapsto \text{nil}],\sigma)) \mid s{=}(\eta,\sigma)\}
\end{aligned}
$$

$$
\begin{aligned}
&[\![x := \texttt{malloc()}]\!]_{er} \triangleq \emptyset \\
&[\![\texttt{free}(x)]\!]_{ok} \triangleq \{(s,(\eta,\sigma[\eta(x) \mapsto \bot])) \mid s{=}(\eta,\sigma) \land \sigma(\eta(x)) \in \text{Val}\} \\
&[\![\texttt{free}(x)]\!]_{er} \triangleq \{(s,s) \mid s{=}(\eta,\sigma) \land (\eta(x){=}\text{nil} \lor \sigma(\eta(x)){=}\bot)\} \\
&[\![x := [y]]\!]_{ok} \triangleq \{(s,(\eta[x \mapsto v],\sigma)) \mid s{=}(\eta,\sigma) \land \sigma(\eta(y)){=}v \in \text{Val}\} \\
&[\![x := [y]]\!]_{er} \triangleq \{(s,s) \mid \sigma{=}(s,\mathsf{h}()) \land (\eta(y){=}\text{nil} \lor \sigma(\eta(y)){=}\bot)\} \\
&[\![[x] := y]\!]_{ok} \triangleq \{(s,(\eta,\sigma[\eta(x) \mapsto \eta(y)])) \mid s{=}(\eta,\sigma) \land \sigma(\eta(x)) \in \text{Val}\} \\
&[\![[x] := y]\!]_{er} \triangleq \{(s,s) \mid \sigma{=}(\eta,\sigma) \land (\eta(x){=}\text{nil} \lor \sigma(\eta(x)){=}\bot)\} \\
&[\![\texttt{local}\ x.\mathsf{C}]\!]_\epsilon \triangleq \{((\eta,\sigma),(\eta'[x \mapsto \eta(x)],\sigma')) \mid ((\eta[x \mapsto \text{nil}],\sigma),(\eta',\sigma')) \in [\![\mathsf{C}]\!]_\epsilon\}
\end{aligned}
$$

Fig. 8. The Pulse-X denotational semantics

A SEMANTICS

The Pulse-X denotational semantic sis given in Fig. 8, and is analogous to that of ISL in [Raad et al. 2020].

B ERROR TRACE OF NPE IN LISTING 1

```
apps/lib/s_cb.c:959: error: Nullptr Dereference
  PISL found a potential null pointer dereference on line 959.

apps/lib/s_cb.c:957:23: in call to `app_malloc`
  955. static int ssl_excert_prepend(SSL_EXCERT **pexc)
  956. {
  957.     SSL_EXCERT *exc = app_malloc(sizeof(*exc), "prepend cert");
                          ^
  958.
  959.     memset(exc, 0, sizeof(*exc));

test/testutil/apps_mem.c:16:16: in call to `CRYPTO_malloc` (modelled)
    14. void *app_malloc(size_t sz, const char *what)
    15. {
    16.     void *vp = OPENSSL_malloc(sz);
                     ^
    17.
    18.     return vp;

test/testutil/apps_mem.c:16:16: is the null pointer
    14. void *app_malloc(size_t sz, const char *what)
    15. {
    16.     void *vp = OPENSSL_malloc(sz);
                     ^
    17.
    18.     return vp;

test/testutil/apps_mem.c:16:5: assigned
    14. void *app_malloc(size_t sz, const char *what)
    15. {
    16.     void *vp = OPENSSL_malloc(sz);
           ^
    17.
    18.     return vp;

test/testutil/apps_mem.c:18:5: returned
    16.     void *vp = OPENSSL_malloc(sz);
    17.
    18.     return vp;
           ^
    19. }

apps/lib/s_cb.c:957:23: return from call to `app_malloc`
  955. static int ssl_excert_prepend(SSL_EXCERT **pexc)
  956. {
  957.     SSL_EXCERT *exc = app_malloc(sizeof(*exc), "prepend cert");
                          ^
  958.
  959.     memset(exc, 0, sizeof(*exc));
```

```
apps/lib/s_cb.c:957:5: assigned
  955. static int ssl_excert_prepend(SSL_EXCERT **pexc)
  956. {
  957.     SSL_EXCERT *exc = app_malloc(sizeof(*exc), "prepend cert");
           ^
  958.
  959.     memset(exc, 0, sizeof(*exc));

apps/lib/s_cb.c:959:5: invalid access occurs here
  957.     SSL_EXCERT *exc = app_malloc(sizeof(*exc), "prepend cert");
  958.
  959.     memset(exc, 0, sizeof(*exc));
           ^
  960.
  961.     exc->next = *pexc;
```

Listing 10. Error trace of the bug in Listing 1.

C MANIFEST ERRORS

THEOREM C.1. *For all manifest errors* $\models [p]\ \mathrm{C}\ [er\colon q]$*:*

$$\forall s.\ \exists s'.\ (s,s') \in [\![\mathrm{C}]\!]_{er} \wedge s' \in (\!|q * \mathsf{true}|\!)$$

PROOF. Pick a valid ISL triple $\models [p]\ \mathrm{C}\ [er\colon q]$ denoting a manifest error. Pick arbitrary s and let $r \triangleq \{s\}$. From Def. 3.2 we then know there exists f such that $p * f \vdash r$ and $\mathsf{sat}(q * f)$ holds. As $[p]\ \mathrm{C}\ [er\colon q]$ is a valid triple, using the Frame rule of ISL we can derive $\models [p * f]\ \mathrm{C}\ [er\colon q * f]$. Subsequently, since $p * f \vdash r$, from Def. 3.1 we also have $\models [r]\ \mathrm{C}\ [er\colon q * f]$. On the other hand, as $\mathsf{sat}(q * f)$ holds, we know there exists s' such that $s' \in (\!|q * f|\!)$ and thus $s' \in (\!|q * \mathsf{true}|\!)$. Moreover, from $\models [r]\ \mathrm{C}\ [er\colon q * f]$, Def. 3.1 and the definitions of r we know $(\!|q * f|\!) \subseteq [\![\mathrm{C}]\!]_{er}(\{s\})$; i.e., $\forall s_q \in (\!|q * f|\!).\ (s, s_q) \in [\![\mathrm{C}]\!]_{er}$, and thus $(s, s') \in [\![\mathrm{C}]\!]_{er}$, as required. □

Manifest Errors and Reverse Under-Approximate Triples. We next demonstrate that under certain conditions manifest errors coincide with *reverse* under-approximate triples. We write $\models \{\!|p|\!\}\ \mathrm{C}\ \{\!|\epsilon : q|\!\}$ to denote a reverse (under-approximate) triple that is valid. Intuitively, a valid reverse triple is the dual of a valid ISL triple (Def. 3.1): $\models \{\!|p|\!\}\ \mathrm{C}\ \{\!|\epsilon : q|\!\}$ denotes that executing C on each state in p reaches some state in q under ϵ, while preserving the frame property. Put formally:

$$\models \{\!|p|\!\}\ \mathrm{C}\ \{\!|\epsilon : q|\!\} \overset{\text{def}}{\Leftrightarrow} \forall r.\ (\!|p * r|\!) \subseteq [\![\mathrm{C}]\!]_{\epsilon}^{-1}((\!|q * r|\!))$$

where $[\![\mathrm{C}]\!]_{\epsilon}^{-1}$ denotes the inverse of the $[\![\mathrm{C}]\!]_{\epsilon}$ relation in Def. 3.1. The notion of reverse under-approximate triples corresponds to that of *total Hoare triples* in [de Vries and Koutavas 2011].

Interestingly, as we show in Theorem C.2 below, given a manifest error $T \triangleq\ \models[p]\ \mathrm{C}\ [er\colon q]$, if $p \equiv \mathsf{emp} \wedge \mathsf{true}$ (i.e., p imposes no spatial or pure constraints on the context), then T is also a valid reverse triple, i.e., $\models \{\!|p|\!\}\ \mathrm{C}\ \{\!|er : q|\!\}$ also holds.

THEOREM C.2. *Given a manifest error* $\models [p]\ \mathrm{C}\ [er\colon q]$*, if* $p \equiv \mathsf{emp} \wedge \mathsf{true}$*, then* $\models \{\!|p|\!\}\ \mathrm{C}\ \{\!|er : q|\!\}$.

PROOF. Pick a manifest error $T \triangleq \models[p]\ \mathrm{C}\ [er\colon q]$ such that $p \equiv \mathsf{emp} \wedge \mathsf{true}$. Pick arbitrary r and $s \in (\!|p * r|\!)$; it then suffices to show there exists $s' \in (\!|q * r|\!)$ such that $(s, s') \in [\![\mathrm{C}]\!]_{\epsilon}$.

Let $r' \triangleq \{s\}$; as $\mathsf{sat}(r')$ holds and T denotes a manifest error, from Def. 3.2 we know $\mathsf{sat}(q * r')$ holds. Moreover as $s \in (\!|p * r|\!)$ and $p \equiv \mathsf{emp} \wedge \mathsf{true}$ (and thus from the semantics of assertions $p * r \equiv r$), we know $s \in (\!|r|\!)$ and thus $r' \vdash r$. Moreover, as $\mathsf{sat}(q * r')$ holds, we know there exists s' such that $s' \in (\!|q * r'|\!)$ and thus $s' \in (\!|q * r|\!)$ since $r' \vdash r$. On the other hand, as T is a valid triple, from Def. 3.1 we know $(\!|q * r'|\!) \subseteq [\![\mathrm{C}]\!]_{\epsilon}((\!|p * r'|\!))$ and thus $(\!|q * r'|\!) \subseteq [\![\mathrm{C}]\!]_{\epsilon}((\!|r'|\!))$ since $p \equiv \mathsf{emp} \wedge \mathsf{true}$. Consequently, as $s' \in (\!|q * r'|\!)$, we know $s' \in [\![\mathrm{C}]\!]_{\epsilon}((\!|r'|\!))$, i.e., $(s, s') \in [\![\mathrm{C}]\!]_{\epsilon}$ since $r' \triangleq \{s\}$. That is, there exists $s' \in (\!|q * r|\!)$ such that $(s, s') \in [\![\mathrm{C}]\!]_{\epsilon}$, as required. □

Definition C.3 (Path-manifest errors). An error triple $\models [p]\ \mathrm{C}\ [er\colon q]$ denotes a *path-manifest error* iff for all r, if $\mathsf{sat}(p * r)$ holds then $\mathsf{sat}(q * r)$ also holds.

THEOREM C.4 (PATH-MANIFEST ERRORS). *An error triple* $\models [p]\ \mathrm{C}\ [er\colon q]$ *with* $p \triangleq \exists \overrightarrow{X_p}.\ \kappa_p \wedge \pi_p$ *and* $q \triangleq \exists \overrightarrow{X_q}.\ \kappa_q \wedge \pi_q$ *denotes a* path-manifest error *if:*

(1) $\mathsf{sat}(q)$ *holds;*
(2) $\mathsf{locs}(\kappa_q) \setminus \overrightarrow{X_q} \subseteq \mathsf{locs}(\kappa_p) \setminus \overrightarrow{X_p}$*;*
(3) for all $\overrightarrow{v}$, $\mathsf{sat}(\pi_q[\overrightarrow{v} / \overrightarrow{Y} \cup \mathsf{locs}(\kappa_q)])$ *holds, where* $\overrightarrow{Y} = \mathsf{flv}(q)$ *and:*

$$\mathsf{locs}(\mathsf{emp}) \triangleq \emptyset \quad \mathsf{locs}(x \mapsto X) \triangleq \{x\} \quad \mathsf{locs}(X \mapsto V) = \mathsf{locs}(X \not\mapsto) \triangleq \{X\} \quad \mathsf{locs}(\kappa_1 * \kappa_2) \triangleq \mathsf{locs}(\kappa_1) \cup \mathsf{locs}(\kappa_2)$$

Proof. Pick arbitrary p, q, r, C, $\overrightarrow{X_p}$, κ_p, π_p, $\overrightarrow{X_q}$, κ_q, π_q and $\overrightarrow{Y}$ such that $p \triangleq \exists \overrightarrow{X_p}.\ \kappa_p \wedge \pi_p$, $q \triangleq \exists \overrightarrow{X_q}.\ \kappa_q \wedge \pi_q$, $\overrightarrow{Y} = \mathsf{flv}(q)$, $\mathsf{sat}(q)$ holds, $\mathsf{locs}(\kappa_q) \setminus \overrightarrow{X_q} \subseteq \mathsf{locs}(\kappa_p) \setminus \overrightarrow{X_p}$, for all $\overrightarrow{v}$, $\mathsf{sat}(\pi_q[\overrightarrow{v}/\overrightarrow{Y} \cup \mathsf{locs}(\kappa_q)])$ holds and $\mathsf{sat}(p * r)$ holds.

As $\mathsf{sat}(p * r)$ holds, we know there exist $\eta, \eta_p, \sigma_p, \sigma_r, \overrightarrow{v_p}$ such that $\eta_p = \eta[\overrightarrow{X_p \mapsto v_p}]$, $\eta_p, \sigma_p \models \kappa_p$, $\eta_p \models \pi_p$, $\eta, \sigma_r \models r$ and $\sigma_p \# \sigma_r$, i.e., $(\eta(\mathsf{locs}(\kappa_p)) \setminus \overrightarrow{X_p}) \cap \mathsf{dom}(\sigma_r) = \emptyset$.

As $\mathsf{locs}(\kappa_q) \setminus \overrightarrow{X_q} \subseteq \mathsf{locs}(\kappa_p) \setminus \overrightarrow{X_p}$, we know there exist $\overrightarrow{Z_1}, \overrightarrow{Z_2}$ such that $\mathsf{locs}(\kappa_q) = \overrightarrow{Z_1} \uplus \overrightarrow{Z_2}$, $\overrightarrow{Z_1} \cap \overrightarrow{X_q} = \emptyset$ (i.e., $\overrightarrow{Z_1} \subseteq \overrightarrow{Y}$), $\overrightarrow{Z_1} \subseteq \mathsf{locs}(\kappa_p) \setminus \overrightarrow{X_p}$ and $\overrightarrow{Z_2} \subseteq \overrightarrow{X_q}$. Note that as $\overrightarrow{Z_1} \subseteq \mathsf{locs}(\kappa_p) \setminus \overrightarrow{X_p}$ and $\eta_p, \sigma_p \models \kappa_p$, from the definition of η we know that $\eta \models \mathsf{disjoint}(\overrightarrow{Z_1})$ holds.

Pick $\overrightarrow{v_2}$ such that $\overrightarrow{v_2} \cap \mathsf{dom}(\sigma_r) = \emptyset$ and $\overrightarrow{v_2} \cap \eta(\overrightarrow{Z_1}) = \emptyset$. Let $\pi_1 = \pi[\overrightarrow{Y} \mapsto \overrightarrow{\eta(Y)}]$ and $\pi_2 = \pi_1[\overrightarrow{Z_2} \mapsto \overrightarrow{v_2}]$. As for all $\overrightarrow{v}$, $\mathsf{sat}(\pi_q[\overrightarrow{v}/\overrightarrow{Y} \cup \mathsf{locs}(\kappa_q)])$ holds, we know that $\mathsf{sat}(\pi_2)$ holds and thus there exists $\eta_q, \overrightarrow{v_3}$ such that $\eta_q{=}\eta[\overrightarrow{Z_2} \mapsto \overrightarrow{v_2}][(\overrightarrow{X_q} \setminus \overrightarrow{Z_2}) \mapsto \overrightarrow{v_3}]$ and $\eta_q \models \pi_q$. That is, there exist $\overrightarrow{v_q}$ such that $\eta_q{=}\eta[\overrightarrow{X_q} \mapsto \overrightarrow{v_q}]$, $\eta_q(\overrightarrow{Y}){=}\eta(\overrightarrow{Y})$ and $\eta_q(\overrightarrow{Z_2}){=}\overrightarrow{v_2}$. Moreover, since $\eta \models \mathsf{disjoint}(\overrightarrow{Z_1})$, $\overrightarrow{Z_1} \subseteq \overrightarrow{Y}$, $\eta_q(\overrightarrow{Y}) = \eta(\overrightarrow{Y})$, $\overrightarrow{v_2} \cap \eta(\overrightarrow{Z_1}) = \emptyset$ and $\eta_q(\overrightarrow{Z_2}) = \overrightarrow{v_2}$, we also have $\eta_q \models \mathsf{disjoint}(\overrightarrow{Z_1} \cup \overrightarrow{Z_2})$. As such, since $\mathsf{locs}(\kappa_q) = \overrightarrow{Z_1} \uplus \overrightarrow{Z_2}$, from Proposition C.9 we have $\eta_q, \sigma_q \models \kappa_q$, where $\sigma_q = [\kappa_q]_{\eta_q}$. Consequently, as $\eta_q = \eta[\overrightarrow{X_q} \mapsto \overrightarrow{v_q}]$, $\eta_q, \sigma_q \models \kappa_q$ and $\eta_q \models \pi_q$ we have $\eta, \sigma_q \models \exists \overrightarrow{X_q}.\ \kappa_q \wedge \pi_q$ and thus $\eta, \sigma_q \models q$.

Lastly, since $\mathsf{dom}(\sigma_q) = \eta_q(\overrightarrow{Z_1}) \uplus \eta_q(\overrightarrow{Z_2}) = \eta(\overrightarrow{Z_1}) \uplus \overrightarrow{v_2}$, $\overrightarrow{v_2} \cap \mathsf{dom}(\sigma_r) = \emptyset$, $(\eta(\mathsf{locs}(\kappa_p)) \setminus \overrightarrow{X_p}) \cap \mathsf{dom}(\sigma_r) = \emptyset$ and thus $\eta(\overrightarrow{Z_1}) \cap \mathsf{dom}(\sigma_r) = \emptyset$ (since $\overrightarrow{Z_1} \subseteq \mathsf{locs}(\kappa_p) \setminus \overrightarrow{X_p}$), we also know that $\mathsf{dom}(\sigma_q) \cap \mathsf{dom}(\sigma_r) = \emptyset$ and thus $\sigma_q \# \sigma_r$. Consequently, as $\eta, \sigma_q \models q$, $\eta, \sigma_r \models r$ and $\sigma_q \# \sigma_r$, we know $\eta, \sigma_q \uplus \sigma_r \models q * r$ and thus $\mathsf{sat}(q * r)$, as required. □

Definition C.5 (Resource-manifest errors). An error triple $\models [p]\ \mathsf{C}\ [er{:}\ q]$ denotes a *resource-manifest error* iff:

- $\models [p]\ \mathsf{C}\ [er{:}\ q]$ denotes a path-manifest error; and
- $\mathsf{pheap}(p)$ holds, where

$$\mathsf{pheap}(\exists \overrightarrow{X}.\ \kappa \wedge \pi) \overset{\text{def}}{\Leftrightarrow} \mathsf{nlocs}(\kappa) = \emptyset \wedge \pi \equiv \mathsf{true}$$
$$\mathsf{nlocs}(\mathsf{emp}) \triangleq \emptyset \qquad \mathsf{nlocs}(x \mapsto V) \triangleq \emptyset \qquad \mathsf{nlocs}(X \mapsto V) \triangleq \emptyset$$
$$\mathsf{nlocs}(X \not\mapsto) \triangleq \{X\} \qquad \mathsf{nlocs}(\kappa_1 * \kappa_2) = \mathsf{nlocs}(\kappa_1) \cup \mathsf{nlocs}(\kappa_2)$$

Theorem C.6 (Resource-manifest errors). *An error triple* $\models [p]\ \mathsf{C}\ [er{:}\ q]$ *with* $p \triangleq \exists \overrightarrow{X_p}.\ \kappa_p \wedge \pi_p$ *and* $q \triangleq \exists \overrightarrow{X_q}.\ \kappa_q \wedge \pi_q$ *denotes a* resource-manifest error *if:*

(1) $\mathsf{sat}(q)$ *holds;*
(2) $\mathsf{locs}(\kappa_q) \setminus \overrightarrow{X_q} \subseteq \mathsf{locs}(\kappa_p) \setminus \overrightarrow{X_p}$*;*
(3) for all $\overrightarrow{v}$, $\mathsf{sat}(\pi_q[\overrightarrow{v}/\overrightarrow{Y} \cup \mathsf{locs}(\kappa_q)])$ *holds, where* $\overrightarrow{Y} = \mathsf{flv}(q)$*;*
(4) $\mathsf{pheap}(p)$ *holds.*

Proof. Follows from Theorem C.4 and the definitions of $\mathsf{pheap}(.)$ and resource-manifest errors. □

Definition C.7 (Manifest errors). An error triple $\models [p]\ \mathsf{C}\ [er{:}\ q]$ denotes a *manifest error* iff for all r, if $\mathsf{sat}(r)$ holds, then there exists f such that $p * f \vdash r$ and $\mathsf{sat}(q * f)$ also holds.

Theorem C.8 (Manifest errors). *An error triple* $\models [p]\ \mathsf{C}\ [er{:}\ q]$ *with* $q \triangleq \exists \overrightarrow{X_q}.\ \kappa_q \wedge \pi_q$ *denotes a* manifest error *if:*

(1) $p \equiv \mathsf{emp} \land \mathsf{true}$*;*
(2) $\mathsf{sat}(q)$ *holds;*
(3) $\mathsf{locs}(\kappa_q) \setminus \overrightarrow{X_q} \subseteq \mathsf{locs}(\kappa_p) \setminus \overrightarrow{X_p}$*;*
(4) for all $\overrightarrow{v}$, $\mathsf{sat}(\pi_q[\overrightarrow{v} / \overrightarrow{Y} \cup \mathsf{locs}(\kappa_q)])$ *holds, where* $\overrightarrow{Y} = \mathsf{flv}(q)$.

Proof. Pick an arbitrary error triple $\models [p]\ \mathsf{C}\ [er\colon q]$ such that conditions (1)-(4) above hold. Pick an arbitrary r such that $\mathsf{sat}(r)$ holds. As $p \equiv \mathsf{emp} \land \mathsf{true}$, we then have $p * r \equiv r$, and thus $p * r \vdash r$. Moreover, from conditions (2)-(4) and Theorem C.4 we know $\models [p]\ \mathsf{C}\ [er\colon q]$ is a path-manifest error. On the other hand, as $\mathsf{sat}(r)$ holds and $p * r \equiv r$, we know $\mathsf{sat}(p * r)$ and thus from the definition of path-manifest errors we have $\mathsf{sat}(q * r)$, as required. □

Proposition C.9. *For all* $\kappa, \pi, \eta, \sigma, \sigma_1, \sigma_2, \kappa_1, \kappa_2$*:*

(1) if $\eta, \sigma_1 \models \kappa$ *and* $\eta, \sigma_2 \models \kappa$*, then* $\sigma_1 = \sigma_2$*;*
(2) $\kappa \equiv \kappa \land \mathsf{disjoint}(\mathsf{locs}(\kappa))$*, where:*

$$\mathsf{disjoint}(\emptyset) = \mathsf{disjoint}(\{X\}) \triangleq \mathsf{true} \qquad \mathsf{disjoint}(\{X\} \uplus S) \triangleq \bigwedge_{Y \in S} X \neq Y \land \mathsf{disjoint}(S)$$

(3) if $\eta, \sigma \models \kappa$*, then* $\eta \models \mathsf{disjoint}(\mathsf{locs}(\kappa))$ *(follows from the previous part);*
(4) if $\eta \models \mathsf{disjoint}(\mathsf{locs}(\kappa))$*, then* $\eta, [\kappa]_\eta \models \kappa$*, where:*

$$[\mathsf{emp}]_\eta = \emptyset \qquad [x \mapsto X]_\eta = [\eta(x) \mapsto \eta(V)] \qquad [X \mapsto Y]_\eta = [\eta(X) \mapsto \eta(Y)]$$
$$[X \not\mapsto]_\eta = [\eta(X) \mapsto \bot] \qquad [\kappa_1 * \kappa_2]_\eta = [\kappa_1]_\eta \uplus [\kappa_2]_\eta$$

(5) if $\eta, \sigma \models \kappa$*, then* $\sigma = [\kappa]_\eta$ *(follows from parts 1, 3 and 4).*