Return

$$\overline{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}$$

$$\Gamma; V; \{f_1\} \leadsto \{\exists \vec{z}. (f_2 \wedge res = e)\} \mid return\ e;$$

$$f_1 \vdash \exists \vec{z}. f_2, \quad fv(e) \subseteq V$$

$$\overline{\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad}$$

$$\Gamma; V; [f_1] \leadsto [ok: \exists \vec{z}. (f_2 \wedge res = e)] \mid return\ e;$$

? $f_1 \vdash \exists \vec{z}. f_2, \quad fv(e) \subseteq V$

Read

$$\frac{\Gamma; V \cup \{v\}; \{\Sigma_1 * u \xmapsto{\iota} (fld : t) \wedge \Pi_1 \wedge v = t\} \leadsto \{f_2\} \mid C}{\Gamma; V; \{\Sigma_1 * u \xmapsto{\iota} (fld : t)\} \wedge \Pi_1 \leadsto \{f_2\} \mid typ\ v = u \to fld = t;}$$

(fld → field of data)

(typ → variable type)

$u \in V$

$v \notin V$

Call

$$\Gamma \cup \{\{q_1\}\ \text{frame}(\vec{u})\{q_2\}\}; V \cup \{r\} \cdot \{F_1 * q_2\theta[v/res]\} \rightsquigarrow \{F_2\} \mid C$$

$$\Gamma \cup \{\{q_1\}\ \text{frame}(\vec{u})\{q_2\}\}; V; \{F_1 * F\} \rightsquigarrow \{F_2\} \mid \text{typ}\ v = \text{frame}(\vec{u}\theta); C$$

$$\boxed{\begin{array}{c} q_1\theta = F, \\ fv(\vec{u}\theta) \in V \end{array}}$$

< same for (SL)

Examples (attempts)

from paper

$(\exists l .\ x \mapsto L * L \mapsto nil \wedge X = nil \wedge L \mathrel{!}= nil]$

from C (set(x,y))

er: $(\exists L .\ x \mapsto L * L \mapsto nil \wedge X = nil \wedge L \mathrel{!}= nil]$

er: $[y \mapsto Y * v \mapsto V * z \mapsto W * Y \mapsto W \wedge Z = nil \wedge w = nil]$ error

([z] := y write)

ok version?

ok: $[z \mapsto Z * Z \mapsto V * y \mapsto Y]\ ([z] := y)\ [z \mapsto X * X \mapsto Y * y \mapsto Y]$

1 → Replace error triple with ok version of the triple?

2 → Negate discovered frame that forces error?

---

1) $[y \mapsto Y * v \mapsto V * z \mapsto W * Y \mapsto W \wedge \cancel{Z} = nil]$

replace

C_{write}

$[y \mapsto Y * v \mapsto V * z \mapsto W * Y \mapsto W \wedge W \mapsto V \wedge Z = nil$

2) $[y \mapsto Y * v \mapsto V * z \mapsto W * Y \mapsto W \wedge Z = nil \wedge W \,!= nil]$

negate frame

C_{write}

$[y \mapsto Y * v \mapsto V * z \mapsto W * Y \mapsto W \wedge Z = nil \wedge W \mapsto V]$

- Can the above be used as VC's?
- Can they be directly used as pre and post?
-

→ more rules? (Early return , error out , assume false
if (π) return ; error() (remove path)

all of this possible because "under-approximate"

no-op delete ??
skip() (if post condition still satisfied)

"Repair" ← z = y → pre
"HOLE"
post

Inference step

$P(x, y, z) \vdash \boxed{pre}$

$\boxed{post} \vdash Q(x, y, z)$

(replace with "ok" version)

$[ok : y \mapsto Y * v \mapsto V * z \mapsto W * Y \mapsto W \wedge z = nil \wedge W \mapsto V]$

$[ok : y \mapsto Y * v \mapsto V * z \mapsto W * Y \mapsto W \wedge z = nil \wedge W = nil]$

OK:

unknown

$$[y \vdash Y * v \vdash V * z \vdash W * Y \vdash W * W \vdash V \wedge z = nil] \vdash Q * K \quad \ldots (1)$$

$$P * K \vdash [y \vdash Y * v \vdash V * z \vdash W * Y \vdash W \wedge z = nil \wedge W = nil] \quad \ldots (2)$$

OK:

unknown

$$Q \stackrel{def}{=} (y \vdash Y * v \vdash V * z \vdash W * Y \vdash W * W \vdash V \wedge z = nil)$$

U_R ?

$$K \stackrel{def}{=} [emp \wedge z = nil]$$

substituting K in (2)

$$P \wedge \underline{z = nil} \vdash [y \vdash Y * v \vdash V * z \vdash W * Y \vdash W \wedge \underline{z = nil \wedge W = nil}]$$

$$\downarrow$$

$$z = nil \nvdash z = nil \wedge W = nil \quad ?$$

$$\downarrow \quad U_2 \quad ?$$

$$P \stackrel{def}{=} [y \vdash Y * v \vdash V * z \vdash W * Y \vdash W \wedge z = nil \wedge W = nil]$$

Synthesis Step

general

Goal : $\Gamma ; V ; [F_1] \rightsquigarrow F_2 \mid C$

available vars

program statements to synth.

declared funcs

pre

post

such that

$[F_1] \; C \; [F_2]$

should hold

---

from previous page.

starting point

$\Gamma ; V ; [y \mapsto Y * v \mapsto V * z \mapsto W * Y \mapsto W \wedge z = nil \wedge w = nil]$

$\rightsquigarrow [y \mapsto Y * v \mapsto V * z \mapsto W * Y \mapsto W * w \mapsto V \wedge z = nil] \mid C$

alloc ?

C : struct$_1$ u = malloc(sizeof(struct$_1$)); C

y

int *

type

$\boxed{u \mapsto \{\bar{v}\} \quad \bar{v} \text{ are fresh}}$

appended to Γ

write ($y = nil$ to $y = v$)

y

$u \to fld = r$ v

skip

done.

Candidate for

```
int *y = malloc(sizeof(int *));
y = &v;   (w/o V)
```

in setC()

to be satisfied?

↳ verify with Infer.

technically correct

Question

- what if NPE is expected?
  - Do early-return fixes work?