Relational Parametricity and Separation Logic

Hongseok Yang, Queen Mary, Univ. of London
Lars Birkedal, IT Univ. of Copenhagen

Challenge

Develop a theory of data abstraction for pointer programs.

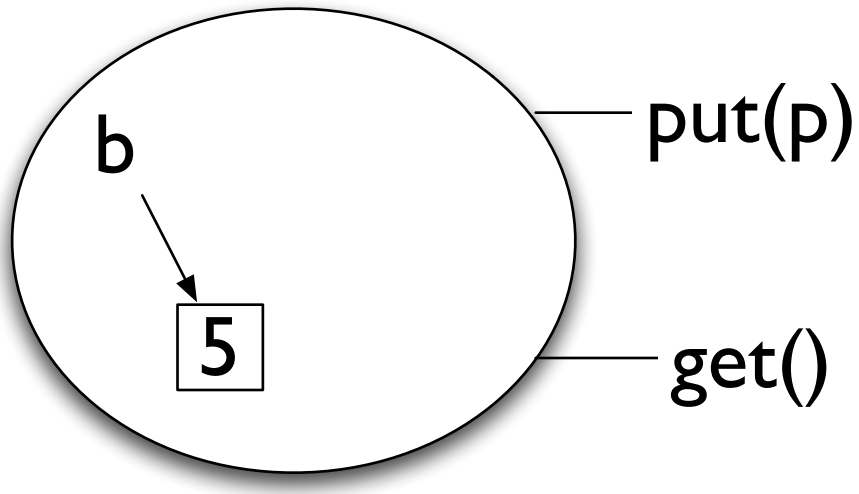
When are two modules equivalent in the presence of pointers?

Message

- Focus on good clients of modules.
- Provability in sep. logic is a good candidate for deciding good clients.

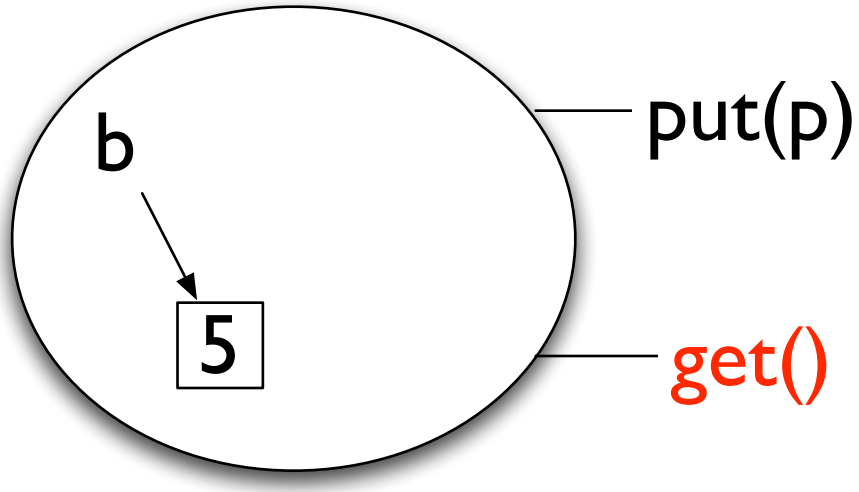
Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```



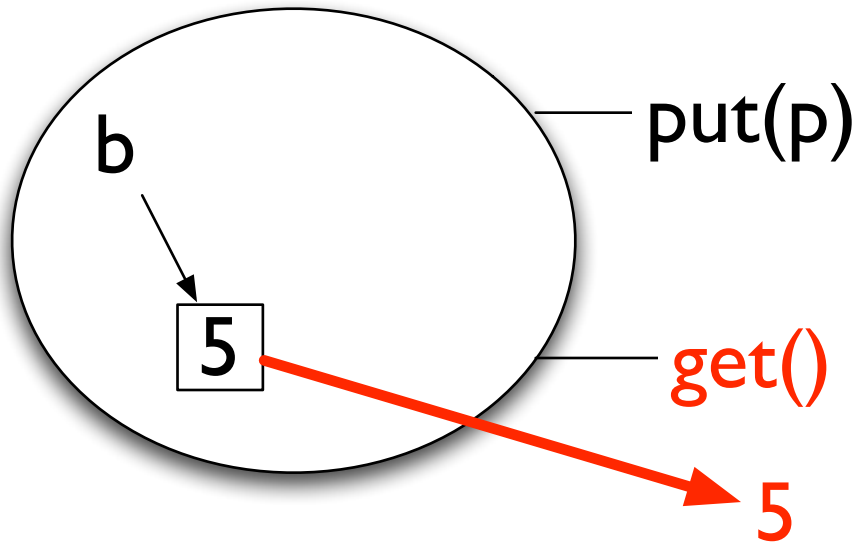
Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```



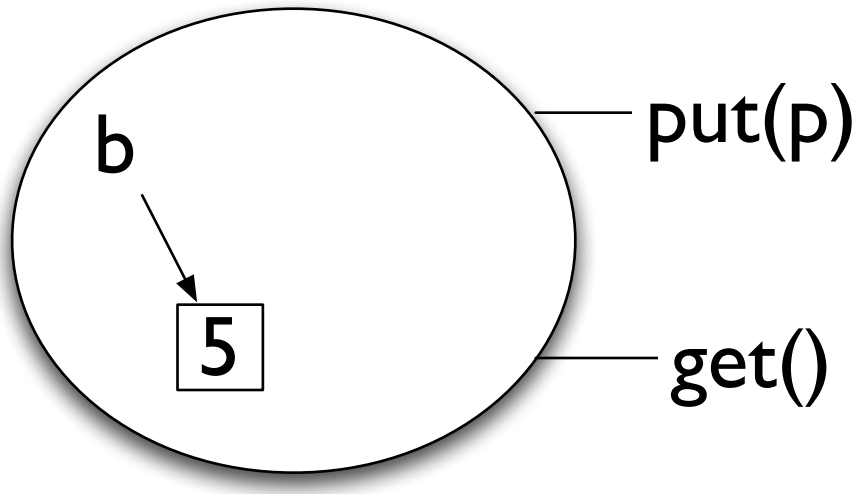
Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```



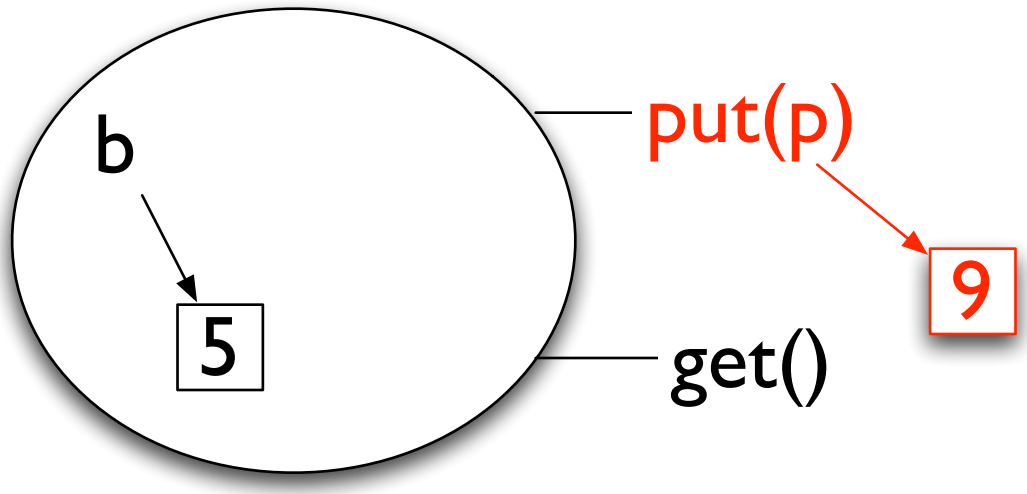
Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```



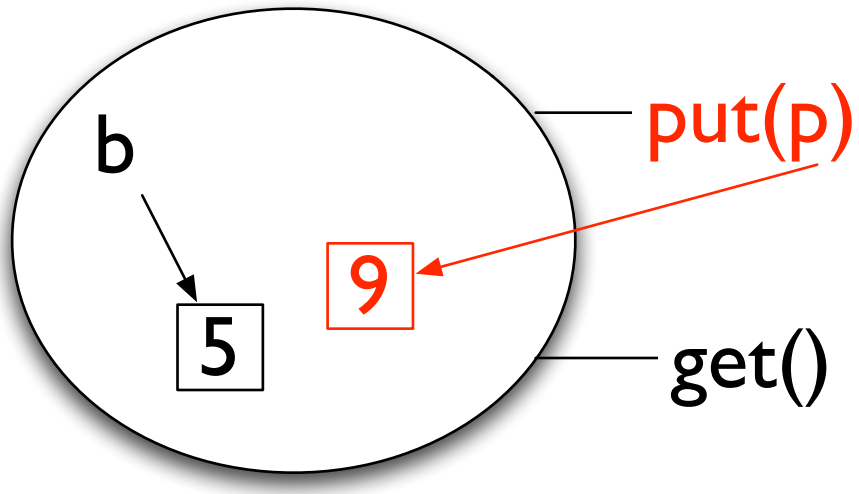
Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```



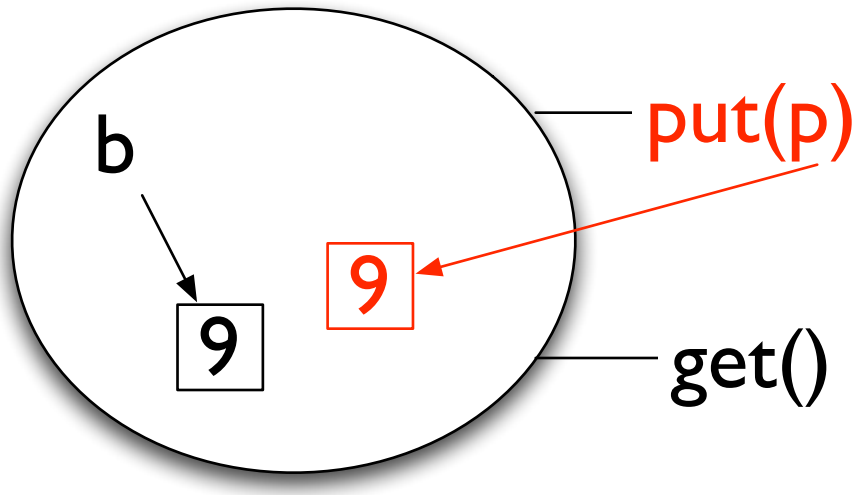
Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```



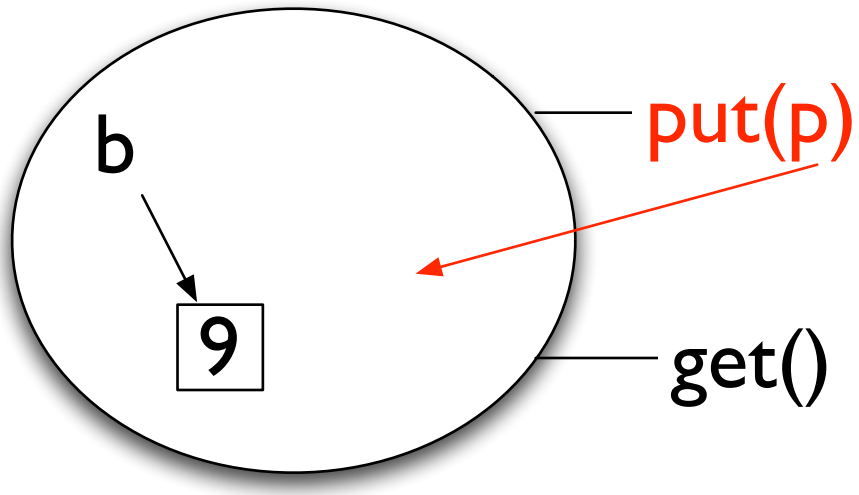
Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```



Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```

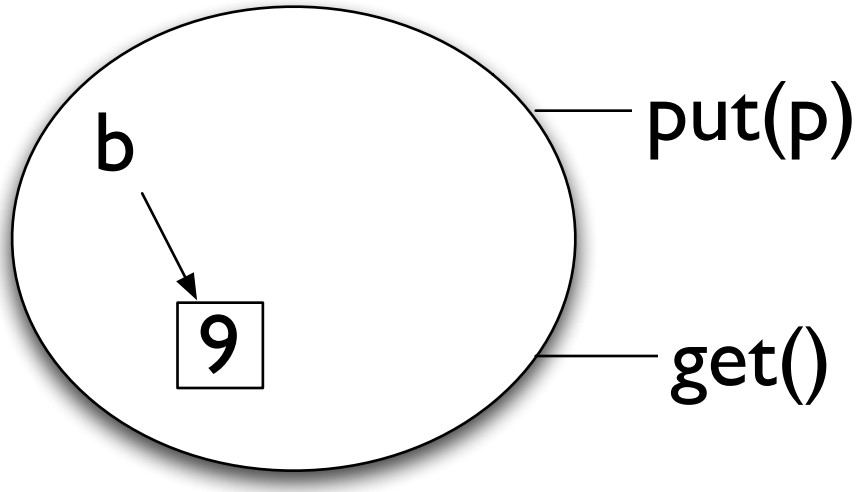


Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```

Module Buffer2

```
int* b=malloc();

void put(int* p) {
    free(b);
    b=p;
}

int get() { return *b; }
```

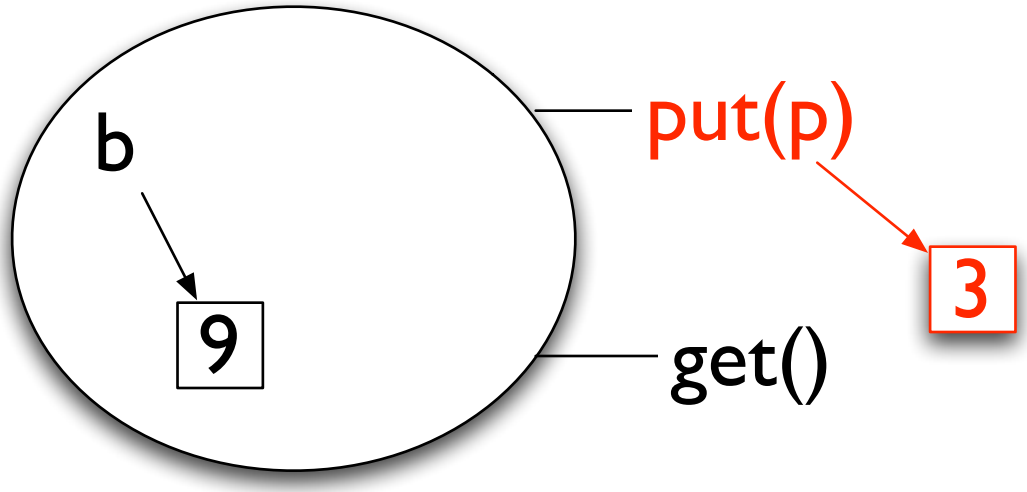


Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```

Module Buffer2

```
int* b=malloc();

void put(int* p) {
    free(b);
    b=p;
}

int get() { return *b; }
```

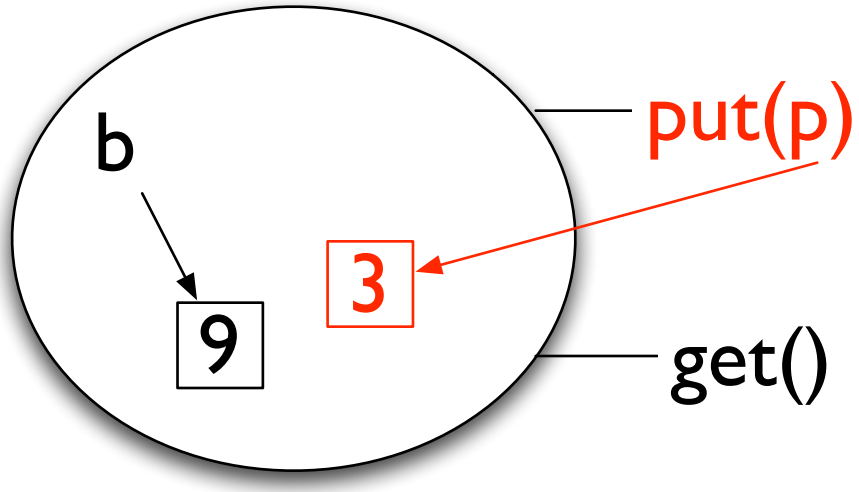


Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```

Module Buffer2

```
int* b=malloc();

void put(int* p) {
    free(b);
    b=p;
}

int get() { return *b; }
```

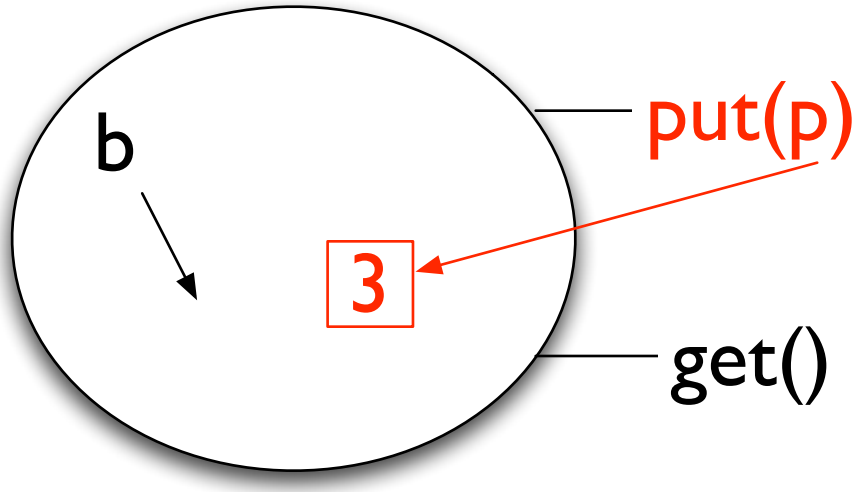


Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```

Module Buffer2

```
int* b=malloc();

void put(int* p) {
    free(b);
    b=p;
}

int get() { return *b; }
```

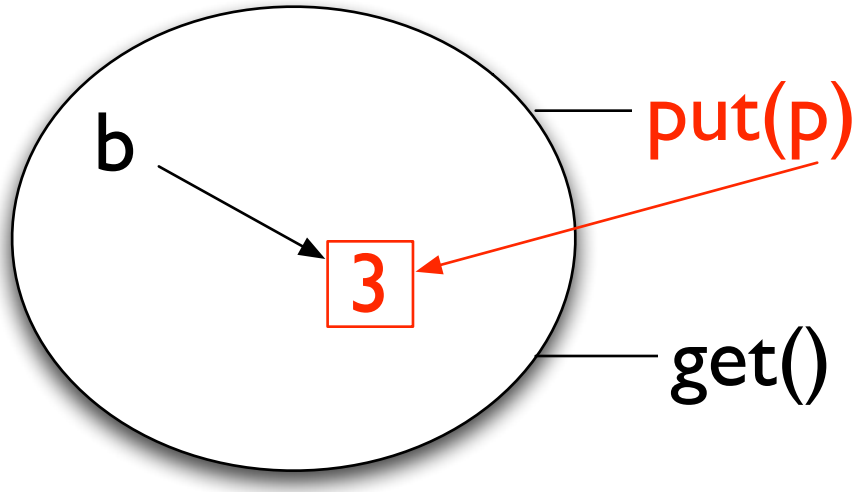


Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```

Module Buffer2

```
int* b=malloc();

void put(int* p) {
    free(b);
    b=p;
}

int get() { return *b; }
```

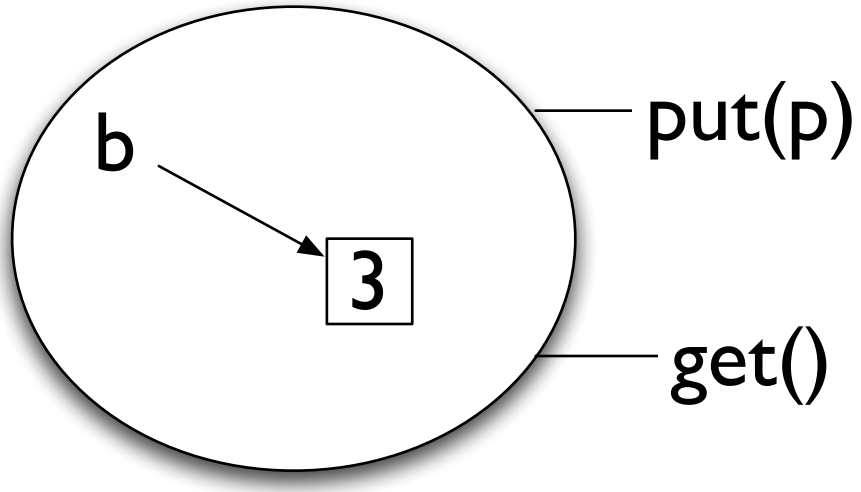


Module Buffer1

```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```

Module Buffer2

```
int* b=malloc();

void put(int* p) {
    free(b);
    b=p;
}

int get() { return *b; }
```



Module Buffer1

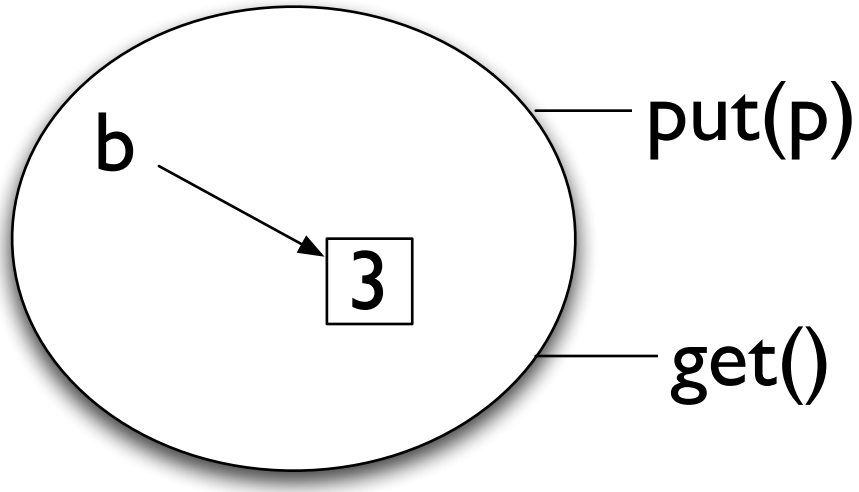
```
int* b=malloc();

void put(int* p) {
    *b=*p;
    free(p);
}

int get() { return *b; }
```

Module Buffer2

```
int* b=malloc();

void put(int* p) {
    free(b);
    b=p;
}

int get() { return *b; }
```

Client Program

```
p=malloc(); *p=5;

put(p);

n = read();

*p=3;
```



Module Buffer1

Module Buffer2

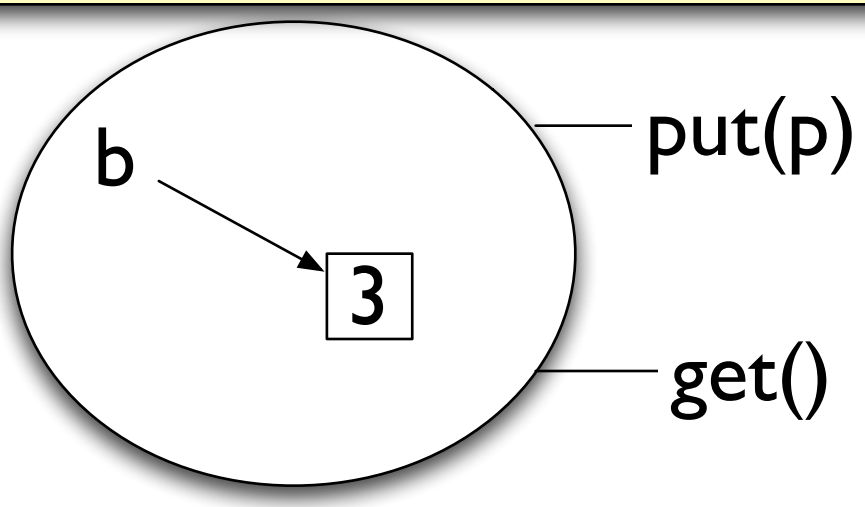
Client Program

```
p=malloc(); *p=5;

put(p);

n = read();

*p=3;
```

Interface Spec

$\{p \mapsto -\}put(p)\{emp\}$

$\{emp\}get()\{emp\}$



Module Buffer1

Module Buffer2

Interface Spec

$\{p \mapsto -\}put(p)\{emp\}$

$\{emp\}get()\{emp\}$

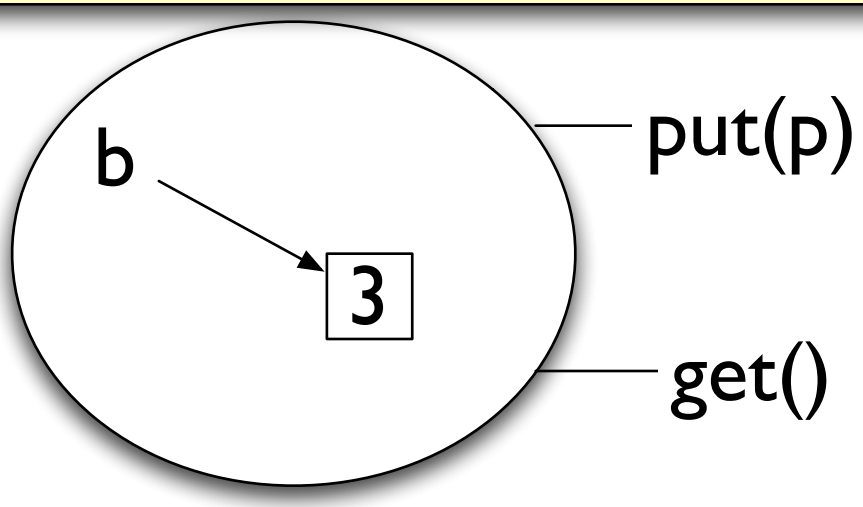
Client Program

{ emp }
p=malloc(); *p=5;

put(p);

n = read();

*p=3;



Module Buffer1

Module Buffer2

Interface Spec

$\{p \mapsto -\}put(p)\{emp\}$

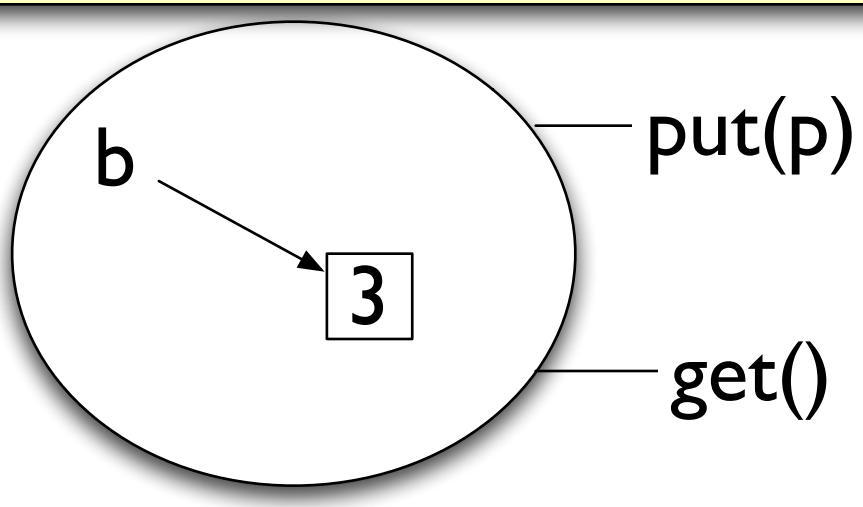
$\{emp\}get()\{emp\}$

Client Program

{ emp }
p=malloc(); *p=5;
{ p ↦ - }
put(p);

n = read();

*p=3;



Module Buffer1

Module Buffer2

Interface Spec

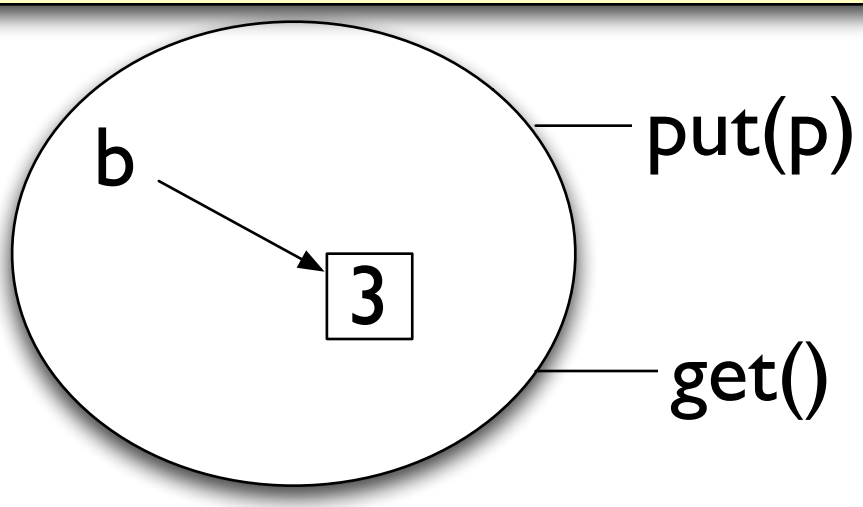
{p↦-}put(p){emp}

{emp}get(){emp}

Client Program

{ emp }
p=malloc(); *p=5;
{ p ↦ - }
put(p);
{ emp }
n = read();

*p=3;



Module Buffer1

Module Buffer2

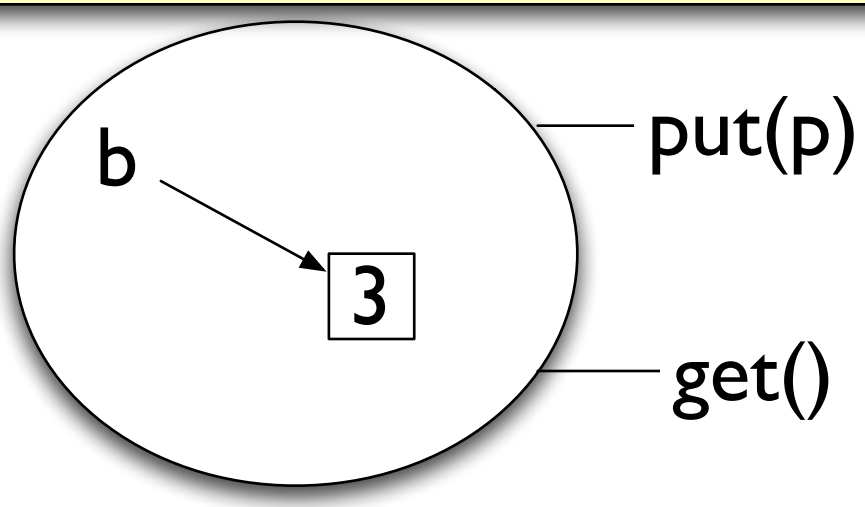
Interface Spec

$\{p \mapsto -\}put(p)\{emp\}$

$\{emp\}get()\{emp\}$

Client Program

{ emp }
p=malloc(); *p=5;
{ p ↦ - }
put(p);
{ emp }
n = read();
{ emp }
*p=3;



Module Buffer1

Module Buffer2

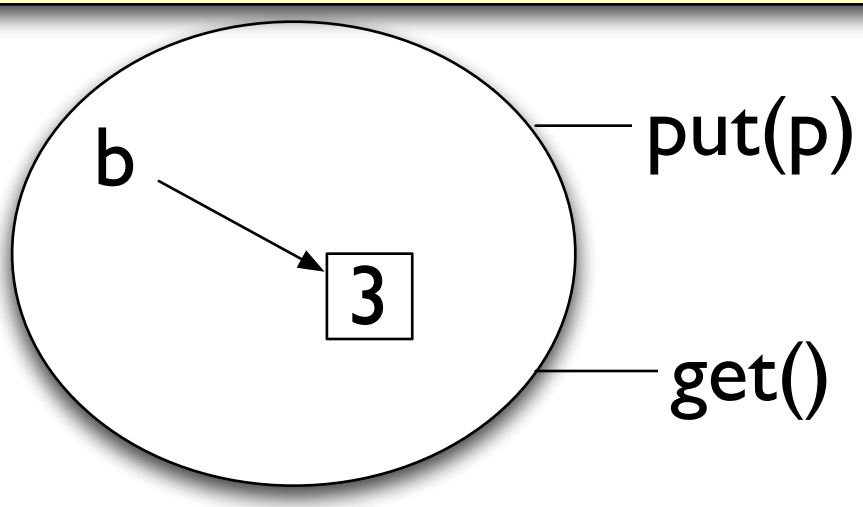
Interface Spec

{p↦-}put(p){emp}

{emp}get(){emp}

Client Program

{ emp }
p=malloc(); *p=5;
{ p ↦ - }
put(p);
{ emp }
n = read();
{ emp }
*p=3;
{ ??? }



Main Result Informally

Buffer1 and Buffer2 are observationally equivalent, if we consider only provable clients C in sep. logic: for some P and Q,

$$\Gamma \vdash_{\mathrm{SL}} \left(\{p \mapsto -\}\mathsf{put}(p)\{\mathsf{emp}\} \wedge \{\mathsf{emp}\}\mathsf{get}()\{\mathsf{emp}\}\right) \Rightarrow \{P\}C\{Q\}$$

Main Result Informally

Buffer1 and Buffer2 are observationally equivalent, if we consider only provable clients C in sep. logic: for some P and Q,

$$\Gamma \vdash_{\mathrm{SL}} \left(\boxed{\{p \mapsto -\}\mathsf{put}(p)\{\mathsf{emp}\} \wedge \{\mathsf{emp}\}\mathsf{get}()\{\mathsf{emp}\}}\right) \Rightarrow \{P\}C\{Q\}$$

Main Result Informally

Buffer1 and Buffer2 are observationally equivalent, if we consider only provable clients C in sep. logic: for some P and Q,

$$\Gamma \vdash_{\mathrm{SL}} \left(\{p \mapsto -\}\mathsf{put}(p)\{\mathsf{emp}\} \wedge \{\mathsf{emp}\}\mathsf{get}()\{\mathsf{emp}\}\right) \Rightarrow \{P\}C\{Q\}$$

Main Result Informally

Buffer1 and Buffer2 are observationally equivalent, if we consider only provable clients C in sep. logic: for some P and Q,

$$\Gamma \vdash_{\mathrm{SL}} \left(\{p \mapsto -\}\mathsf{put}(p)\{\mathsf{emp}\} \wedge \{\mathsf{emp}\}\mathsf{get}()\{\mathsf{emp}\}\right) \Rightarrow \{P\}C\{Q\}$$

Main Result Informally

Buffer1 and Buffer2 are observationally equivalent, if we consider only provable clients C in sep. logic: for some P and Q,

$$\Gamma \vdash_{\mathrm{SL}} \left(\{p \mapsto -\}\mathsf{put}(p)\{\mathsf{emp}\} \wedge \{\mathsf{emp}\}\mathsf{get}()\{\mathsf{emp}\}\right) \Rightarrow \{P\}C\{Q\}$$

Client Program

```
{ emp }
p=malloc(); *p=5;
{ p ↦ - }
put(p);
{ emp }
n = read();
{ emp }
*p=3;
{ ??? }
```

Main Result Informally

Buffer1 and Buffer2 are observationally equivalent, if we consider only provable clients C in sep. logic: for some P and Q,

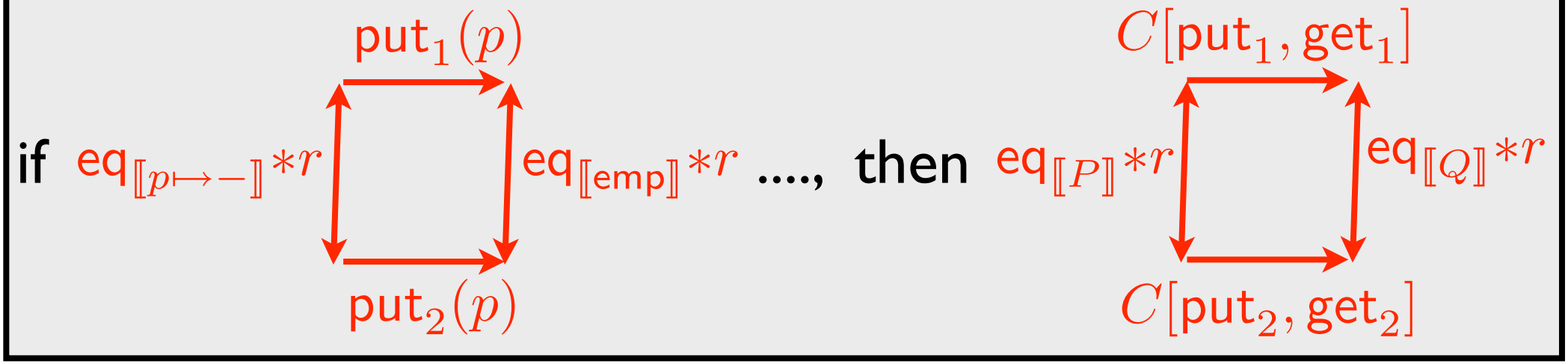
$$\Gamma \vdash_{\mathrm{SL}} \left(\{p \mapsto -\}\mathsf{put}(p)\{\mathsf{emp}\} \wedge \{\mathsf{emp}\}\mathsf{get}()\{\mathsf{emp}\}\right) \Rightarrow \{P\}C\{Q\}$$

Main Result Informally

Buffer1 and Buffer2 are observationally equivalent, if we consider only provable clients C in sep. logic: for some P and Q,

$$\Gamma \vdash_{\mathrm{SL}} \left(\{p \mapsto -\}\mathsf{put}(p)\{\mathsf{emp}\} \wedge \{\mathsf{emp}\}\mathsf{get}()\{\mathsf{emp}\}\right) \Rightarrow \{P\}C\{Q\}$$

for all r,



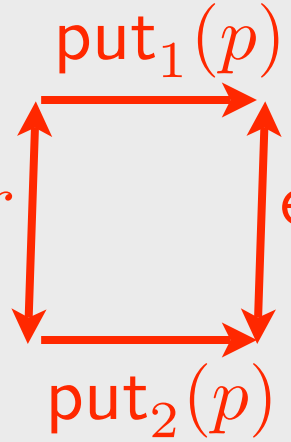
By Relational Semantics of Specifications in Sep. Logic

Bu
we consider only provable clients C in sep. logic. for some P and Q,

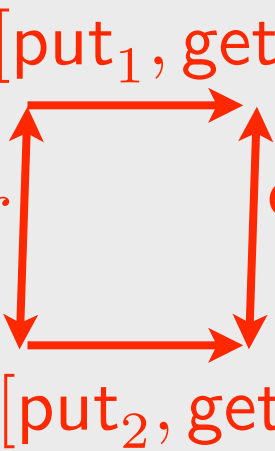
$$\Gamma \vdash_{\mathrm{SL}} \left(\{p \mapsto -\}\mathsf{put}(p)\{\mathsf{emp}\} \wedge \{\mathsf{emp}\}\mathsf{get}()\{\mathsf{emp}\}\right) \Rightarrow \{P\}C\{Q\}$$

for all r,

if $\mathsf{eq}_{[\![p \mapsto -]\!]} * r$



$\mathsf{eq}_{[\![\mathsf{emp}]\!]} * r$ ...., then $\mathsf{eq}_{[\![P]\!]} * r$



$\mathsf{eq}_{[\![Q]\!]} * r$

Sample Specification

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

Syntax

$$\boxed{k}\text{: com} \quad \vdash \quad \{l \mapsto -\}\boxed{k}\{\text{emp}\} \Rightarrow \{P\}\boxed{C}\{Q\}$$

- Commands. (e.g. free(x), *x=y.)

Syntax

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}\, k\, \{\mathsf{emp}\} \Rightarrow \{P\}\, C\, \{Q\}$$

- Commands. (e.g. free(x), *x=y.)
- Assertions. (e.g. emp, l↦-.)

Syntax

$$k : \text{com} \quad \vdash \quad \boxed{\{l \mapsto -\}k\{\text{emp}\} \Rightarrow \{P\}C\{Q\}}$$

- Commands. (e.g. free(x), *x=y.)
- Assertions. (e.g. emp, l↦-.)
- Specifications. (e.g. {l↦-}k{emp}, {emp}C{emp}.)

Semantics of Commands and Assertions

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

$$\begin{aligned}
\mathsf{Heaps} &\stackrel{\mathrm{def}}{=} \mathsf{Locations} \rightharpoonup_{\mathrm{fin}} \mathsf{Values} \\
[\![\mathsf{com}]\!] &\stackrel{\mathrm{def}}{=} \mathsf{Heaps} \rightarrow_{\mathrm{local}} (\mathsf{Heaps} \cup \{\mathsf{fault}\}) \\
[\![\mathsf{assert}]\!] &\stackrel{\mathrm{def}}{=} \mathcal{P}(\mathsf{Heaps})
\end{aligned}$$

Relational Semantics of Specifications

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

$$\eta_1, \eta_2, r \models \varphi$$

Relational Semantics of Specifications

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

Environments
E.g. $[k \rightarrow [\![\mathsf{free}(l)]\!]]$

$$\eta_1, \eta_2, r \models \varphi$$

Relational Semantics of Specifications

$$k : \mathsf{com} \;\vdash\; \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

Environments
E.g. $[k \rightarrow [\![\mathsf{free}(l)]\!]]$

$$\eta_1, \eta_2, r \models \varphi$$

Relations on Heaps
E.g. $h_1[\mathsf{eq}_H]h_2$ iff $h_1 = h_2 \wedge h_2 \in q$
$h_1[\mathsf{allocEmp}_l]h_2$ iff $\exists v.\ h_1{=}[l{\rightarrow}v] \wedge h_2{=}[]$

Semantics of Hoare Triples

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

$$\eta_1, \eta_2, r \quad \models \quad \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \rightarrow \mathsf{eq}_{[\![Q]\!]} * r \right] [\![C]\!]_{\eta_2}$$

Semantics of Hoare Triples

Separating Conj. for Relations: $h_1[r_1 * r_2]h_2$ holds iff

$$h_1 = n_1 \bullet m_1 \ \wedge \ h_2 = n_2 \bullet m_2 \ \wedge \ n_1[r_1]m_1 \ \wedge \ n_2[r_2]m_2$$

$$\eta_1, \eta_2, r \ \models \ \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\boxed{\mathsf{eq}_{[\![P]\!]} * r} \rightarrow \boxed{\mathsf{eq}_{[\![Q]\!]} * r} \right] [\![C]\!]_{\eta_2}$$

Semantics of Hoare Triples

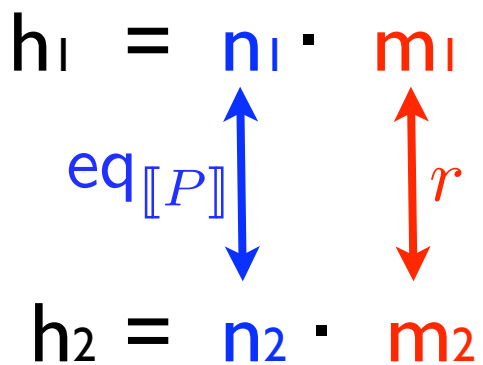
Separating Conj. for Relations: $h_1[r_1 * r_2]h_2$ holds iff

$$h_1 = n_1 \bullet m_1 \ \wedge \ h_2 = n_2 \bullet m_2 \ \wedge \ n_1[r_1]m_1 \ \wedge \ n_2[r_2]m_2$$

$$\eta_1, \eta_2, r \ \models \ \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \rightarrow \mathsf{eq}_{[\![Q]\!]} * r \right] [\![C]\!]_{\eta_2}$$



Semantics of Hoare Triples

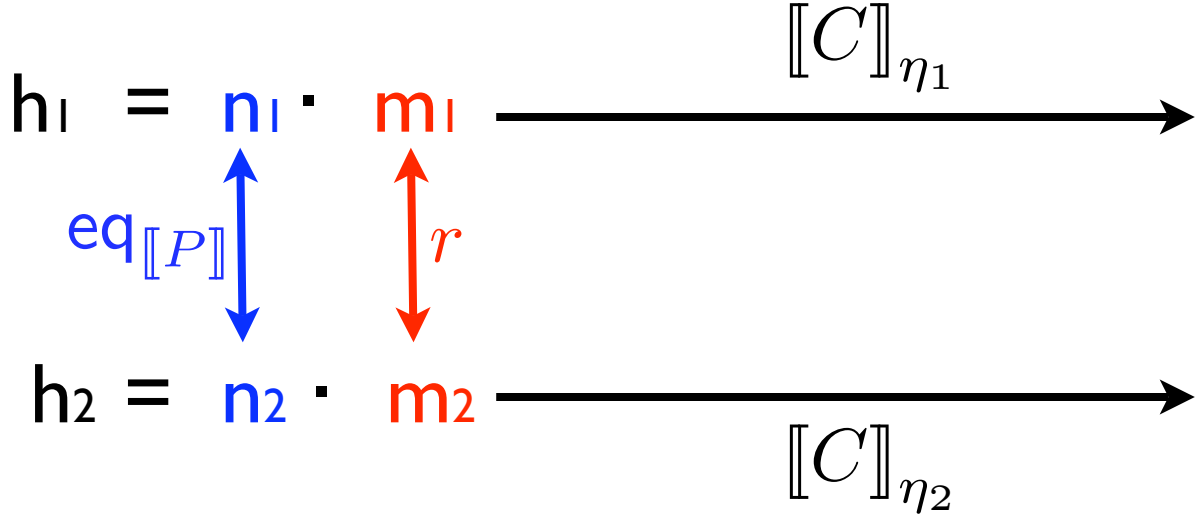
Separating Conj. for Relations: $h_1[r_1 * r_2]h_2$ holds iff

$$h_1 = n_1 \bullet m_1 \ \wedge \ h_2 = n_2 \bullet m_2 \ \wedge \ n_1[r_1]m_1 \ \wedge \ n_2[r_2]m_2$$

$$\eta_1, \eta_2, r \ \models \ \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \rightarrow \mathsf{eq}_{[\![Q]\!]} * r \right] [\![C]\!]_{\eta_2}$$



Semantics of Hoare Triples

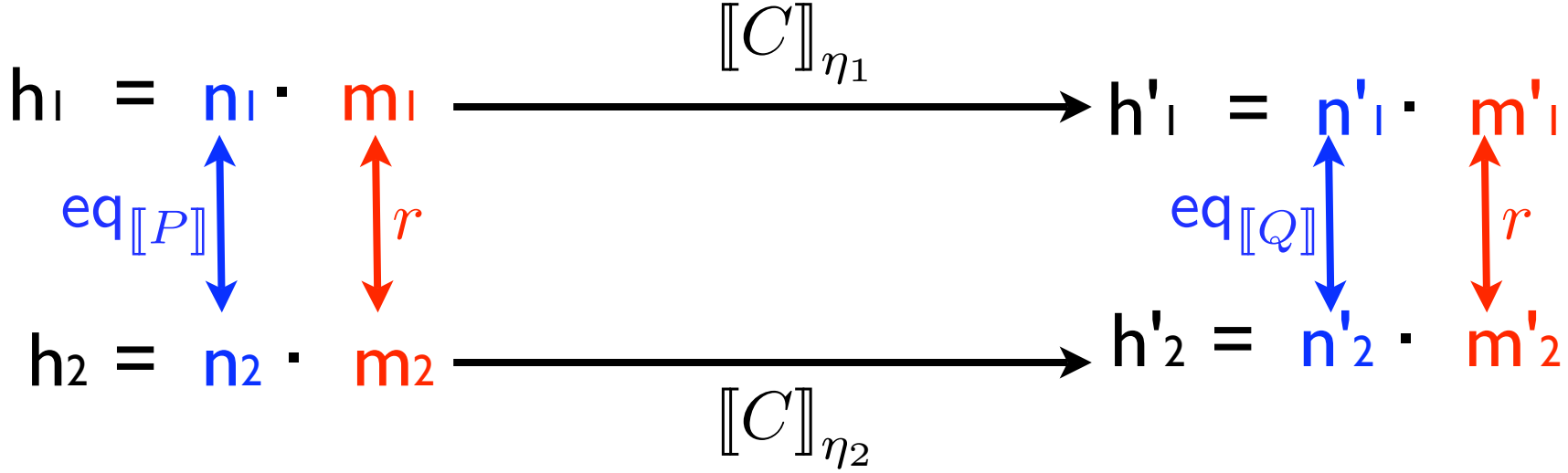
Separating Conj. for Relations: $h_1[r_1 * r_2]h_2$ holds iff

$$h_1 = n_1 \bullet m_1 \ \wedge \ h_2 = n_2 \bullet m_2 \ \wedge \ n_1[r_1]m_1 \ \wedge \ n_2[r_2]m_2$$

$$\eta_1, \eta_2, r \ \models \ \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \rightarrow \mathsf{eq}_{[\![Q]\!]} * r \right] [\![C]\!]_{\eta_2}$$



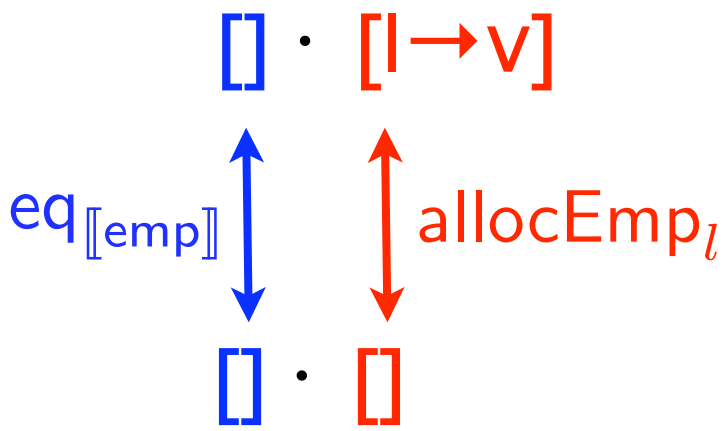
Semantics of Hoare Triples

$$[k \to [\![*l = 0]\!]],\ [k \to [\![\mathsf{skip}]\!]],\ \mathsf{allocEmp}_l \models \{\mathsf{emp}\}k\{\mathsf{emp}\}$$

$$\eta_1, \eta_2, r \models \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \longrightarrow \mathsf{eq}_{[\![Q]\!]} * r\right] [\![C]\!]_{\eta_2}$$



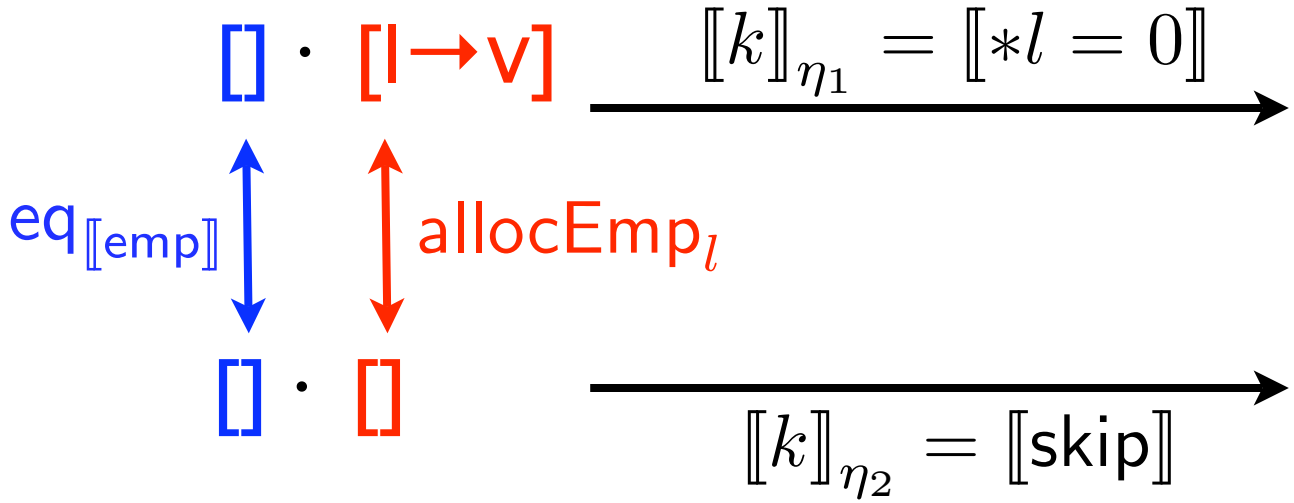
Semantics of Hoare Triples

$$[k \to [\![*l = 0]\!]],\ [k \to [\![\mathsf{skip}]\!]],\ \mathsf{allocEmp}_l \models \{\mathsf{emp}\}k\{\mathsf{emp}\}$$

$$\eta_1, \eta_2, r \models \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \longrightarrow \mathsf{eq}_{[\![Q]\!]} * r\right] [\![C]\!]_{\eta_2}$$



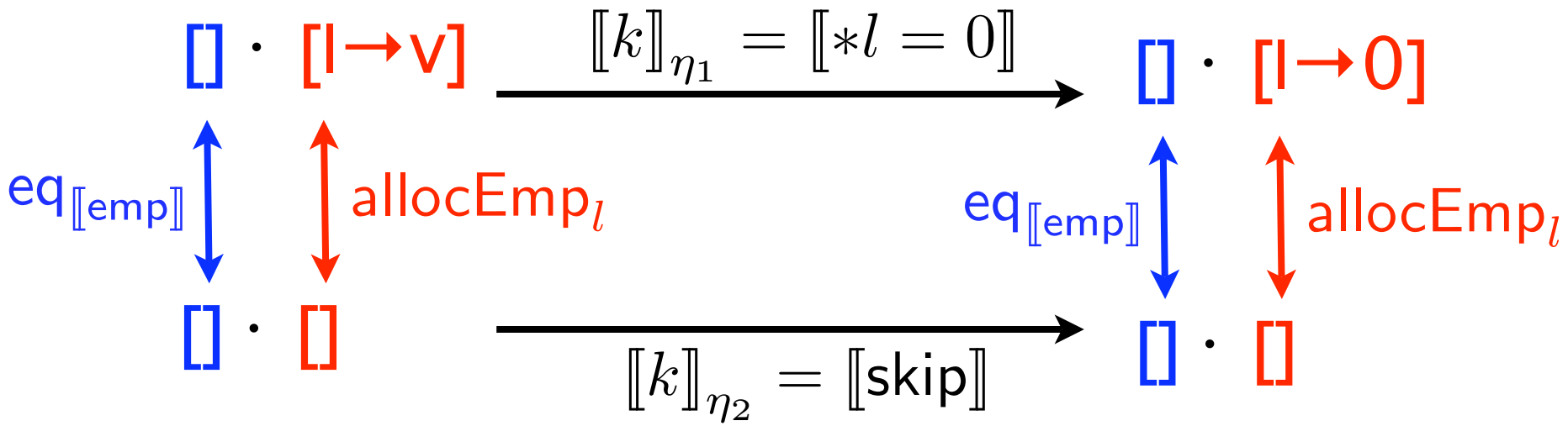
Semantics of Hoare Triples

$$[k \to [\![*l = 0]\!]],\ [k \to [\![\mathsf{skip}]\!]],\ \mathsf{allocEmp}_l \models \{\mathsf{emp}\}k\{\mathsf{emp}\}$$

$$\eta_1, \eta_2, r \models \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \to \mathsf{eq}_{[\![Q]\!]} * r\right] [\![C]\!]_{\eta_2}$$



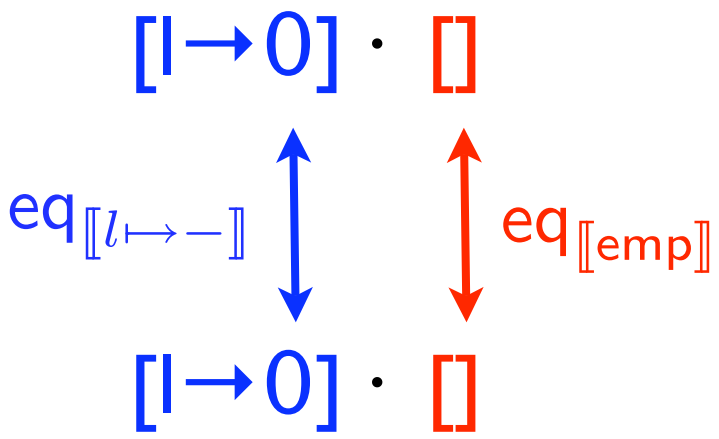
Semantics of Hoare Triples

$$\eta_1, \eta_2, \mathsf{eq}_{[\![\mathsf{emp}]\!]} \models \{l \mapsto -\}\mathsf{dispose}(l)\{\mathsf{emp}\}$$

$$\eta_1, \eta_2, r \models \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \rightarrow \mathsf{eq}_{[\![Q]\!]} * r\right] [\![C]\!]_{\eta_2}$$



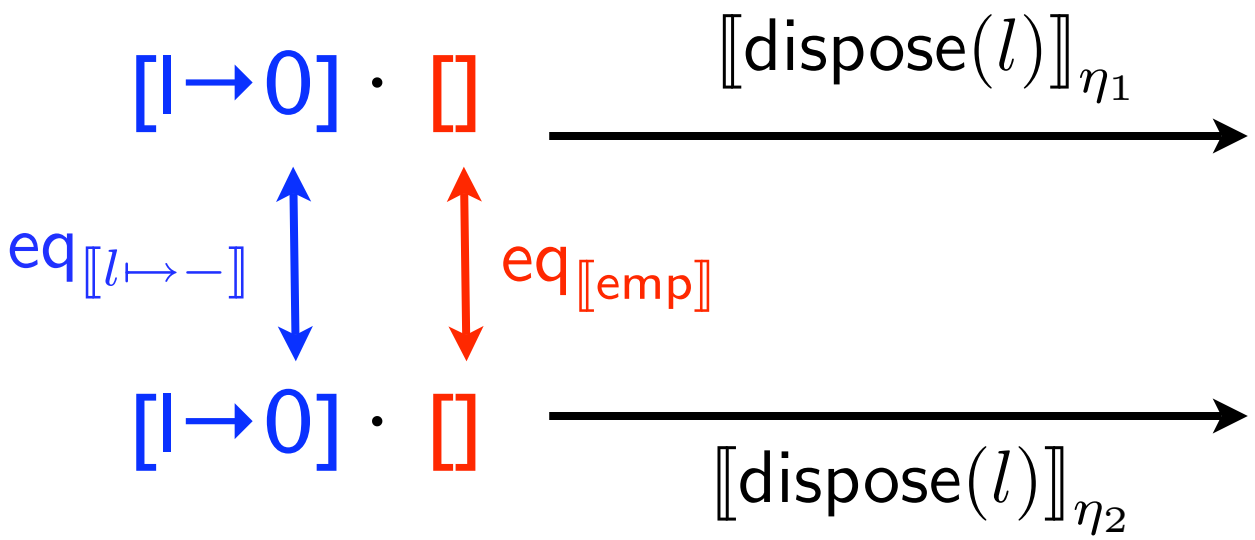
Semantics of Hoare Triples

$$\eta_1, \eta_2, \mathsf{eq}_{[\![\mathsf{emp}]\!]} \models \{l \mapsto -\}\mathsf{dispose}(l)\{\mathsf{emp}\}$$

$$\eta_1, \eta_2, r \models \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \rightarrow \mathsf{eq}_{[\![Q]\!]} * r\right] [\![C]\!]_{\eta_2}$$



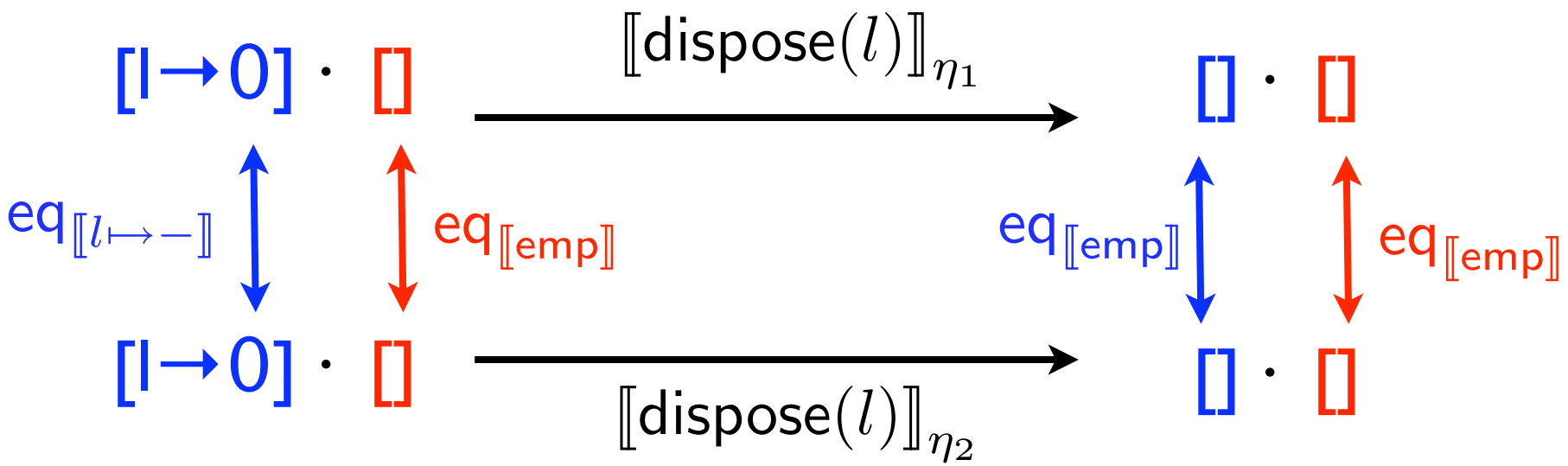
Semantics of Hoare Triples

$$\eta_1, \eta_2, \mathsf{eq}_{[\![\mathsf{emp}]\!]} \models \{l \mapsto -\}\mathsf{dispose}(l)\{\mathsf{emp}\}$$

$$\eta_1, \eta_2, r \models \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \rightarrow \mathsf{eq}_{[\![Q]\!]} * r\right] [\![C]\!]_{\eta_2}$$

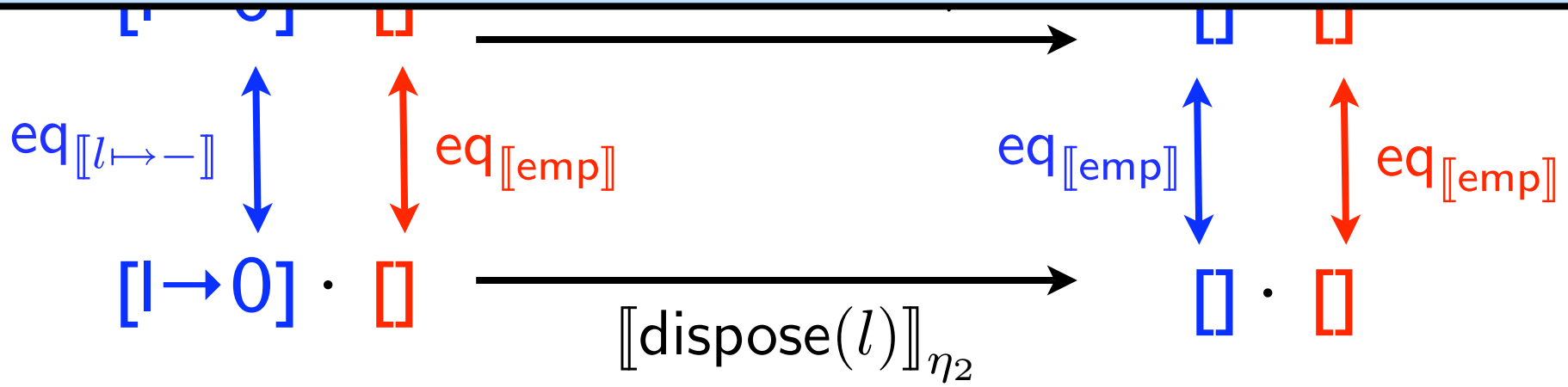


Semantics of Hoare Triples

$$\eta_1, \eta_2, \mathsf{eq}_{[\![\mathsf{emp}]\!]} \models \{l \mapsto -\}\mathsf{dispose}(l)\{\mathsf{emp}\}$$

Lemma: Suppose C does not call functions.

$$\models_{\text{usual}} \{P\}C\{Q\} \quad \text{if and only if} \quad \models_{\text{new}} \{P\}C\{Q\}$$



Semantics of Implications

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

$$\eta_1, \eta_2, r \quad \models \quad \varphi \Rightarrow \psi$$

iff

$$\forall r'.\ \text{if } (\eta_1, \eta_2, r * r' \models \varphi),\ \text{then } (\eta_1, \eta_2, r * r' \models \psi)$$

Semantics of Implications

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

$$\eta_1, \eta_2, r \quad \models \quad \varphi \Rightarrow \psi$$

iff

$$\forall r'.\ \text{if } (\eta_1, \eta_2, r * r' \models \varphi),\ \text{then } (\eta_1, \eta_2, r * r' \models \psi)$$

Kripke Semantics of Intuit. Logic.

- Worlds: Relations r on Heaps
- Accessibility Relation:

$$r \sqsubseteq r' \iff \exists r_0.\, r * r_0 = r'.$$

Relational Semantics of Specifications

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

Relational Semantics of Specifications

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

$$\eta_1, \eta_2, \mathsf{eq}_{[\![\mathsf{emp}]\!]} \quad \models \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

Relational Semantics of Specifications

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

$$\eta_1, \eta_2, \mathsf{eq}_{[\![\mathsf{emp}]\!]} \quad \models \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

$$\forall r.\, \text{if } (\eta_1, \eta_2, r \models \{l \mapsto -\}k\{\mathsf{emp}\}),\ \text{then } (\eta_1, \eta_2, r \models \{P\}C\{Q\}).$$

Relational Semantics of Specifications

$$k : \mathsf{com} \quad \vdash \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

$$\eta_1, \eta_2, \mathsf{eq}_{[\![\mathsf{emp}]\!]} \quad \models \quad \{l \mapsto -\}k\{\mathsf{emp}\} \Rightarrow \{P\}C\{Q\}$$

$$\forall r.\ \text{if } (\eta_1, \eta_2, r \models \{l \mapsto -\}k\{\mathsf{emp}\}),\ \text{then } (\eta_1, \eta_2, r \models \{P\}C\{Q\}).$$

$$\forall r.\ \text{if } \eta_1(k)[\mathsf{eq}_a * r \to \mathsf{eq}_b * r]\eta_2(k), \text{then } [\![C]\!]_{\eta_1}[\mathsf{eq}_p * r \to \mathsf{eq}_q * r][\![C]\!]_{\eta_2}$$
$$(\text{where } a = [\![l \mapsto -]\!],\ b = [\![\mathsf{emp}]\!],\ p = [\![P]\!],\ \text{and } q = [\![Q]\!])$$

Semantics of Triples

$$[\![\mathsf{com}]\!] \stackrel{\text{def}}{=} \mathsf{Heaps} \rightarrow_{\text{local}} (\mathsf{Heaps} \cup \{\mathsf{fault}\})$$

$$\eta_1, \eta_2, r \models \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \rightarrow \mathsf{eq}_{[\![Q]\!]} * r \right] [\![C]\!]_{\eta_2}$$

Semantics of Triples

$$[\![\mathsf{com}]\!] \stackrel{\text{def}}{=} \mathsf{Heaps} \rightarrow_{\cancel{\text{local}}} (\mathsf{Heaps} \cup \{\mathsf{fault}\})$$

$$\eta_1, \eta_2, r \models \{P\}C\{Q\}$$

iff

$$[\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r \rightarrow \mathsf{eq}_{[\![Q]\!]} * r \right] [\![C]\!]_{\eta_2}$$

Semantics of Triples

$$[\![\mathsf{com}]\!] \stackrel{\text{def}}{=} \mathsf{Heaps} \rightarrow_{\cancel{\text{local}}} (\mathsf{Heaps} \cup \{\mathsf{fault}\})$$

$$\eta_1, \eta_2, r \models \{P\}C\{Q\}$$

iff

$$\forall r'.\ [\![C]\!]_{\eta_1} \left[\mathsf{eq}_{[\![P]\!]} * r * r' \rightarrow \mathsf{eq}_{[\![Q]\!]} * r * r'\right] [\![C]\!]_{\eta_2}$$

Semantics of Triples

$$[\![\text{com}]\!] \stackrel{\text{def}}{=} \text{Heaps} \rightarrow_{\text{local}} (\text{Heaps} \cup \{\text{fault}\})$$

$$\eta_1, \eta_2, r \models \{P\}C\{Q\}$$

iff

$$\forall r'.\ [\![C]\!]_{\eta_1} \left[\text{eq}_{[\![P]\!]} * r * r' \rightarrow \text{eq}_{[\![Q]\!]} * r * r' \right] [\![C]\!]_{\eta_2}$$

[Lemma] Supp. C does not call fns. If $\models_{\text{new}} \{P\}C\{Q\}$, $[\![C]\!]$ satisfies the locality cond. restricted to $[\![P{*}\text{true}]\!]$.

[Frame Property]

$$h \in [\![P{*}\text{true}]\!] \wedge [\![C]\!](h) = h_1 \Longrightarrow [\![C]\!](h \bullet h_9) = h_1 \bullet h_9$$

[Frame Property when set quantification is used]

$$h \in [\![P{*}\mathsf{true}]\!] \wedge [\![C]\!](h) = h_1 \Longrightarrow \exists h_2 \in [\![Q]\!].\ [\![C]\!](h \bullet h_9) = h_2 \bullet h_9$$

$$[\![\mathsf{com}]\!] \stackrel{\text{def}}{=} \mathsf{Heaps} \rightarrow_{\text{~~local~~}} (\mathsf{Heaps} \cup \{\mathsf{fault}\})$$

$$\eta, q \models \{P\}C\{Q\}$$

iff

$$\forall q_0.\ [\![C]\!]_\eta([\![P]\!] * q_0 * q) \subseteq [\![C]\!]_\eta([\![Q]\!] * q_0 * q)$$

[Lemma] Supp. C does not call fns. If $\models_{\text{new}} \{P\}C\{Q\}$, $[\![C]\!]$ satisfies the locality cond. restricted to $[\![P{*}\mathsf{true}]\!]$.

[Frame Property]

$$h \in [\![P{*}\mathsf{true}]\!] \wedge [\![C]\!](h) = h_1 \Longrightarrow [\![C]\!](h \bullet h_9) = h_1 \bullet h_9$$

Semantics in the Paper

Uses FM-domain theory and biorthogonality to deal with memory allocation.

Theory of Data Abstraction for Pointer Programs

- Focus on good clients.
- Provability in sep. logic is a good candidate for deciding good clients.

Relational Kripke Reading of Logic

- Can reveal that proofs in a logic says more than what are intended.