Load and Store Semantics

• Step relation: $c / st, m \rightarrow c' / st', m'$

\[
\begin{align*}
  e / st \downarrow \ell & \quad m(\ell) = v \\
  x := *e / st, m & \rightarrow \text{SKIP} / st(x \mapsto v), m
\end{align*}
\]

\[
\begin{align*}
  e_1 / st \downarrow \ell & \quad e_2 / st \downarrow v \\
  *e_1 := e_2 / st, m & \rightarrow \text{SKIP} / st, m(\ell \mapsto v)
\end{align*}
\]
Hoare Logic Rules

\[[P] \text{ skip } [P]\] \quad \{ [x \mapsto e]P \} \ x := e \ [P] \quad \{P\} \ c_1 \ {Q} \quad \{Q\} \ c_2 \ {R}\]

\[[P \land e = \text{true}] c_1 \ {Q}\] \quad \{P \land e = \text{false}\} \ c_2 \ {Q}\]

\[[P \} \text{ if } e \text{ then } c_1 \text{ else } c_2 \ {Q}\]

\[P_1 \Rightarrow P_2 \quad \{P_2\} \ c \ {Q_2} \quad Q_2 \Rightarrow Q_1\] \quad \{P \land e = \text{true}\} \ c \ {P}\]

\[\{P\} \text{ while } e \text{ do } c \ {P \land e = \text{false}}\]
Hoare Logic with Memory

• Assertions $P, Q$ were predicates on states (in Coq, state $\rightarrow$ Prop)
• Now predicates on states and memory (state $\times$ mem $\rightarrow$ Prop)

• $\{X = 5\}(st, m)$ still means $st \ X = 5$

• $\{X \mapsto 5\}(st, m)$ means $\exists \ell, st \ X = \ell \land m(\ell) = 5$
Hoare Logic with Memory

• \{X = 5\}(st, m) still means st \ X = 5
• \{X \mapsto 5\}(st, m) means \exists \ell, st \ X = \ell \land m(\ell) = 5

\{ X = 5 \land Y = 5 \land Z = 5 \} X := 3 \{ X = 3 \land Y = 5 \land Z = 5 \}

\{ X \mapsto 5 \land Y \mapsto 5 \land Z \mapsto 5 \} \ast X := 3 \{ X \mapsto 3 \land Y \mapsto 5 \land Z \mapsto 5 \}

Not if \ Y or Z hold the same address as \ X! (aliasing)
Hoare Logic with Memory

- \{X = 5\}(st, m) still means \(st X = 5\)
- \{X \mapsto 5\}(st, m) means \(\exists \ell, st X = \ell \land m(\ell) = 5\)

\[
\{ X = 5 \land Y = 5 \land Z = 5 \} \ X := 3 \ \{ X = 3 \land Y = 5 \land Z = 5 \}
\]

\[
\{ X \mapsto 5 \land Y \mapsto 5 \land Z \mapsto 5 \land X \neq Y \land X \neq Z \} \\
* X := 3 \\
\{ X \mapsto 3 \land Y \mapsto 5 \land Z \mapsto 5 \}
\]

Not if \(Y\) or \(Z\) hold the same address as \(X\)! (aliasing)
Hoare Logic with Memory

• \{X = 5\}(st, m) still means st X = 5
• \{X \mapsto 5\}(st, m) means \exists \ell, st X = \ell \land m(\ell) = 5

\{ X = 5 \land Y = 5 \land Z = 5 \} X := 3 \{ X = 3 \land Y = 5 \land Z = 5 \}

\{ X \mapsto 5 \ast Y \mapsto 5 \ast Z \mapsto 5 \}
\ast X := 3
\{ X \mapsto 3 \ast Y \mapsto 5 \ast Z \mapsto 5 \}

P \ast Q \text{ means “} P \text{ and } Q, \text{ but not at the same address(es)”}
Separation Logic

- \( \{X = 5\}(st, m) \) still means \( st\ X = 5 \)
- \( \{X \mapsto 5\}(st, m) \) means \( \exists \ell, st\ X = \ell \land m(\ell) = 5 \)
- \( \{P \ast Q\}(st, m) \) means \( \exists m_1 \ m_2, \{P\}(st, m_1) \land \{Q\}(st, m_2) \land m_1 \perp m_2 \land m_1 \cup m_2 = m \)

\( m_1, m_2 \) are partial memories, sometimes called sub-heaps

\( m_1 \perp m_2 \) means that \( m_1 \) and \( m_2 \) have no locations in common
Separation Logic

• \{X = 5\}(st, m) still means \(st X = 5\)
• \{X \mapsto 5\}(st, m) means \(\exists \ell, st X = \ell \land m(\ell) = 5\) and \(\ell\) is the only location in \(m\)
• \{P \star Q\}(st, m) means \(\exists m_1 \ m_2, \{P\}(st, m_1) \land \{Q\}(st, m_2) \land m_1 \bot m_2 \land m_1 \cup m_2 = m\)

\(m_1, m_2\) are partial memories, sometimes called sub-heaps

\(m_1 \bot m_2\) means that \(m_1\) and \(m_2\) have no locations in common

So \(\{X \mapsto 5 \star Y \mapsto 5\}\)(st, m) means “there are two locations in \(m\), whose addresses are in \(X\) and \(Y\), and both store the value 5”
Separation Logic: Frame Rule

\[
\frac{\{P\} \ c \ \{Q\}}{\{P \ast R\} \ c \ \{Q \ast R\}} \quad \text{when} \ \text{vars}(c) \cap \text{vars}(R) = \emptyset
\]

\{P\} \ c \ \{Q\} \text{ defines the “memory footprint” of } c
Separation Logic Rules

\[
\begin{align*}
\{ [x \mapsto e]P \} x := e \{ P \} \\
\{ e \mapsto v \} x := * e \{ e \mapsto v \land x = v \} \\
\{ e_1 \mapsto v \} \star e_1 := e_2 \{ e_1 \mapsto e_2 \}
\end{align*}
\]
Separation Logic Example

\{a[0] \mapsto v_0 * a[1] \mapsto v_1 * \cdots * a[N - 1] \mapsto v_{N-1} *

b[0] \mapsto u_0 * b[1] \mapsto u_1 * \cdots * b[N - 1] \mapsto u_{N-1}\} 

i = 0;
while(i < N){
    b[i] = a[i];
    i = i + 1;
}

\{a[0] \mapsto v_0 * a[1] \mapsto v_1 * \cdots * a[N - 1] \mapsto v_{N-1} *

b[0] \mapsto v_0 * b[1] \mapsto v_1 * \cdots * b[N - 1] \mapsto v_{N-1}\}
Separation Logic Example

\{a \mapsto v_0 \ast a + 1 \mapsto v_1 \ast \cdots \ast a + N - 1 \mapsto v_{N-1} \ast
\}
\{b \mapsto u_0 \ast b + 1 \mapsto u_1 \ast \cdots \ast b + N - 1 \mapsto u_{N-1}\}

i = 0;
while(i < N){
    *(b + i) = *(a + i);
    i = i + 1;
}
\{a \mapsto v_0 \ast a + 1 \mapsto v_1 \ast \cdots \ast a + N - 1 \mapsto v_{N-1} \ast
\}
\{b \mapsto v_0 \ast b + 1 \mapsto v_1 \ast \cdots \ast b + N - 1 \mapsto v_{N-1}\}
Separation Logic Example

\{a \mapsto v_0 \ast a + 1 \mapsto v_1 \ast \cdots \ast a + N - 1 \mapsto v_{N-1} \ast \\
\text{ } b \mapsto u_0 \ast b + 1 \mapsto u_1 \ast \cdots \ast b + N - 1 \mapsto u_{N-1}\}\}

i = 0;
while(i < N){
    t = *(a + i);
    *(b + i) = t;
    i = i + 1;
}

\{a \mapsto v_0 \ast a + 1 \mapsto v_1 \ast \cdots \ast a + N - 1 \mapsto v_{N-1} \ast \\
\text{ } b \mapsto v_0 \ast b + 1 \mapsto v_1 \ast \cdots \ast b + N - 1 \mapsto v_{N-1}\\}
Separation Logic Example

\{a \mapsto v_0 \ast a + 1 \mapsto v_1 \ast \cdots \ast a + N - 1 \mapsto v_{N-1} \ast
b \mapsto u_0 \ast b + 1 \mapsto u_1 \ast \cdots \ast b + N - 1 \mapsto u_{N-1}\}

i = 0;
while(i < N){
    \{a \mapsto v_0 \ast a + 1 \mapsto v_1 \ast \cdots \ast a + N \mapsto v_N \ast
    b \mapsto v_0 \ast b + 1 \mapsto v_1 \ast \cdots \ast b + i - 1 \mapsto v_{i-1} \ast b + i \mapsto u_i \ast \cdots \ast b + N - 1 \mapsto u_{N-1}\}
    \text{t} = \ast(a + i);
    \ast(b + i) = \text{t};
    i = i + 1;
}
\{a \mapsto v_0 \ast a + 1 \mapsto v_1 \ast \cdots \ast a + N - 1 \mapsto v_{N-1} \ast
b \mapsto v_0 \ast b + 1 \mapsto v_1 \ast \cdots \ast b + N - 1 \mapsto v_{N-1}\}