Australian National University

COMP3610/6361
Principles of Programming Languages

Peter Höfner

Sep 27, 2023

Section 18

Weakest Preconditions

Generating Preconditions

$$\{\ ?\ \}\ c\ \{Q\}$$

- many possible preconditions
- some are more useful than others

Australian National University

Weakest Liberal Preconditions

Intuition: the weakest liberal precondition for c and Q is the *weakest* assertion P such that $\{P\}\ c\ \{Q\}$ is valid

Definition (Weakest Liberal Precondition)

P is a *weakest liberal precondition* of c and Q (wlp(c, Q)) if

$$\forall s, I.\ s \models_I P \iff \mathcal{C}[\![c]\!](s) \text{ is undefined } \vee\ \mathcal{C}[\![c]\!](s) \models_I Q$$

Weakest Preconditions

$$
\begin{aligned}
\text{wlp}(\textbf{skip}, P) &= P \\
\text{wlp}(l := a, P) &= P[a/l] \\
\text{wlp}((c_1 \ ; c_2), P) &= \text{wlp}(c_1, \text{wlp}(c_2, P)) \\
\text{wlp}(\textbf{if } b \textbf{ then } c_1 \textbf{ else } c_2, P) &= (b \Longrightarrow \text{wlp}(c_1, P)) \land \\
& (\neg b \Longrightarrow \text{wlp}(c_2, P)) \\
\text{wlp}(\textbf{while } b \textbf{ do } c, P) &= (b \Longrightarrow \text{wlp}(c, \text{wlp}(\textbf{while } b \textbf{ do } c, P))) \ \land \\
& (\neg b \Longrightarrow P) \\
&= \bigwedge_i F_i(P)
\end{aligned}
$$

where

$$
\begin{aligned}
F_0(P) &= \texttt{true} \\
F_{i+1}(P) &= (\neg b \Longrightarrow P) \land (b \Longrightarrow \text{wlp}(c, F_i(P)))
\end{aligned}
$$

(Greatest fixed point)

Properties of Weakest Preconditions

Lemma (Correctness of wlp)

$\forall c \in \textit{com}, Q \in \textit{assn}.$
$\models \{\textit{wlp}(c, Q)\}\ c\ \{Q\}$ *and*
$\forall R \in \textit{assn}.\ \models \{R\}\ c\ \{Q\}$ *implies* $(R \Longrightarrow \textit{wlp}(c, Q))$

Lemma (Provability of wlp)

$\forall c \in \textit{com}, Q \in \textit{assn}.\ \vdash \{\textit{wlp}(c, Q)\}\ c\ \{Q\}$

Soundness and Completeness

Theorem (Relative Completeness)

$P, Q \in$ *assn*, $c \in$ *com*. $\models \{P\}\ c\ \{Q\}$ *implies* $\vdash \{P\}\ c\ \{Q\}$.

Proof Sketch.

- let $\{P\}\ c\ \{Q\}$ be a valid partial correctness specification
- by the first lemma we have $\models P \implies \text{wlp}(c, Q)$
- by the second lemma we have $\vdash \{\text{wlp}(c, Q)\}\ c\ \{Q\}$
- hence $\vdash \{P\}\ c\ \{Q\}$, using the Rule (cons)

$\square$

Total Correctness

Definition (Weakest Precondition)

P is a *weakest precondition* of c and Q (wp(c, Q)) if

$$\forall s, I.\ s \models_I P \iff \mathcal{C}[\![c]\!](s) \models_I Q$$

all rules are the same, except the one for while. This requires a fresh ghost variable that guarantees termination

Lemma (Correctness of wp)

$\forall c \in$ *com*$, Q \in$ *assn*.
$\models [$*wp*$(c, Q)]\ c\ [Q]$ *and*
$\forall R \in$ *assn*. $\models [R]\ c\ [Q]$ *implies* $(R \Longrightarrow$ *wp*$(c, Q))$
(for appropriate definition of $\models$*)*

Strongest Postcondition

$$\{P\}\ c\ \{\ ?\ \}$$

- wlp motivates backwards reasoning
- this seems unintuitive and unnatural
- however, often it is known what a program is supposed to do
- sometimes forward reasoning is useful, e.g. reverse engineering

Strongest Postcondition

$$
\begin{aligned}
\mathsf{sp}(\textbf{skip}, P) &= P \\
\mathsf{sp}(l := a, P) &= \exists v.\ (l = a[v/l] \wedge P[v/l]) \\
\mathsf{sp}((c_1 \ ; c_2), P) &= \mathsf{sp}(c_2, \mathsf{sp}(c_1, P)) \\
\mathsf{sp}(\textbf{if}\ b\ \textbf{then}\ c_1\ \textbf{else}\ c_2, P) &= (\mathsf{sp}(c_1, b \wedge P)) \vee (\mathsf{sp}(c_2, \neg b \wedge P)) \\
\mathsf{sp}(\textbf{while}\ b\ \textbf{do}\ c, P) &= \mathsf{sp}(\textbf{while}\ b\ \textbf{do}\ c, \mathsf{sp}(c, P \wedge b)) \vee (\neg b \wedge P)
\end{aligned}
$$

where

$$
\begin{aligned}
F_0(P) &= \texttt{false} \\
F_{i+1}(P) &= (\neg b \wedge P) \vee (\mathsf{sp}(c, F_i(P \wedge b)))
\end{aligned}
$$

(Least fixed point)