Lecture 4

Compute Relation for this Program

r = 0; ... assume (...) ...

while (x > 0) {

r = r + 3;

x = x - 1

}

$r = \{((r,x),(r',x')) \mid \ldots ? \}$

As the program state use the pair of integer variables (r,x)

1) compute guarded command language for this program (express 'while' and 'if' using 'assume').

Compute Relation for this Program

r, x, r', x'

```
r = 0;
while (x > 0) {
  r = r + 3;
  x = x - 1
}
```

```
r = 0;
( assume(x>0);
  r = r + 3;
  x = x – 1 )* ;
assume(x <= 0)
```

$$\left[\!\left[\begin{array}{l} \text{assume(x>0);} \\ \text{r = r + 3;} \\ \text{x = x – 1} \end{array} \right]\!\right] = [\![B]\!] = \{ \ldots \mid x > 0 \wedge r' = r + 3 \wedge x' = x - 1 \}$$

$$[\![B^*]\!] = [\![B]\!]^* = \bigcup_{k \geq 0} [\![B]\!]^k$$

$$\left[\!\left[\begin{array}{l} \text{r = 0;} \\ \text{B*;} \\ \text{assume(x <= 0)} \end{array} \right]\!\right] =$$

2) compute meaning of program pieces, from smaller to bigger

B

$$r' = r + 3 \ \wedge$$

$$x' = x - 1 \ \wedge$$

$$x > 0$$

B^k for $k > 0$

$r' = r + 3k \;\land$

$x' = x - k \;\land$

$x > 0 \;\land\; x - 1 > 0 \;\land\; \ldots \;\land\; x - (k-1) > 0$

i.e.

$r' = r + 3k \;\land$

$x' = x - k \;\land$

$x - (k - 1) > 0$

B^k for $k >= 0$

$$
\begin{aligned}
&(k > 0 \land \\
&\ r' = r + 3k \land \\
&\ x' = x - k \land \\
&\ x - (k - 1) > 0) \\
&\qquad\qquad\qquad\lor \\
&(k = 0 \land r' = r \land x' = x)
\end{aligned}
$$

B^*

$$(s,s') \in B^* \quad \Leftrightarrow \quad \exists\, k >= 0.\ (s,s') \in B^k$$

$$B^* = \{((r,x),(r',x')) \mid \exists\, k.\ k >= 0 \wedge$$
$$((k > 0 \wedge r' = r + 3k \ \wedge x' = x - k \ \ \wedge x - k >= 0) \ \ \vee$$
$$(k = 0 \wedge r' = r \wedge x' = x))\} =$$

$\mathbf{k=x\text{-}x'}$

$$\{((r,x),(r',x')) \mid$$
$$(\exists\, k.k > 0 \wedge r' = r + 3k \wedge x' = x - \mathbf{k} \wedge x - k >= 0) \vee$$
$$(\exists\, k.\ k = 0 \wedge r' = r \wedge x' = x)\} =$$

$$\{((r,x),(r',x')) \mid (r' = r + 3(x\text{-}x') \wedge x' >= 0 \wedge x\text{-}x' > 0) \ \ \vee$$
$$(r' = r \wedge x' = x)\}$$

Back to the Entire Program

r = 0;
B*;
assume(x <= 0)

```
r = 0;
while (x > 0) {
  r = r + 3;
  x = x - 1
}
```

$\{((r,x),(r',x')) \mid x' = x \wedge r' = 0\} \circ$
$\{((r,x),(r',x')) \mid (r'= r + 3(x-x') \wedge x' >= 0 \wedge x-x' > 0) \ \vee \ (r'= r \wedge x' = x)\} \circ$
$\{((r,x),(r',x')) \mid r' = r \wedge x' = x \wedge x <= 0)\} =$

$\{((r,x),(r',x')) \mid (r'= 3(x-x') \wedge x' >= 0 \wedge x-x' > 0) \ \vee \ (r = 0 \wedge r'= 0 \wedge x' = x)\} \circ$
$\{((r,x),(r',x')) \mid r' = r \wedge x' = x \wedge x <= 0)\} =$

$\{((r,x),(r',x')) \mid (r'= 3(x-x') \wedge x' >= 0 \wedge x-x' > 0) \vee$
$(r = 0 \wedge r'= 0 \wedge x' = x \wedge x <= 0)\}$

ask Princess if you can drop this

The above is the final relation for the program.

Correctness as Relation Inclusion

program $\rightarrow$ relation p
specification $\rightarrow$ relation s

program meets specification:

$$p \subseteq s$$

example: $p = \{((r,x),(r',x')).\ r'=2x \wedge x'=0\ \}$
$s = \{((r,x),(r',x')).\ x > 0 \rightarrow r' > x'\ \}$

then the above program p meets the specification s

because implication holds:

$$r'=2x \wedge x'=0 \rightarrow (\ x > 0 \rightarrow r' > x'\ )$$

$\cap$ $\wedge$
$\cup$ $\vee$
$\rightarrow$ $\subseteq$
$\leftrightarrow$ $=$

Checking Contracts (require/ensure)

```
require(x > 0)
r = 0;
while (x > 0) {
  r = r + 3;
  x = x - 1
}
ensure(r == 3*(x-old(x)))
```

$\{((r,x),(r',x')) \mid (r' = 3(x'-x) \wedge x' >= 0 \wedge x-x' > 0) \vee$
$(r = 0 \wedge r' = 0 \wedge x' = x \wedge x <= 0)\}$

Try to prove the validity of:

$pre(x) \wedge program(x,x') \rightarrow post(x,x')$
$program(x,x') \rightarrow (pre(x) \rightarrow post(x,x'))$

$((r' = 3(x-x') \wedge x' >= 0 \wedge x-x' > 0) \vee (r = 0 \wedge r' = 0 \wedge x' = x \wedge x <= 0)) \rightarrow$
$(x > 0 \rightarrow r' == 3(x' - x))$

The program assertion holds if and only if the formula is valid.

Translating **ensure:** **x** becomes **x'** whereas **old(x)** becomes **x**

Checking Assertions

```
r = 0;
while (x > 0) {
  r = r + 3;
  x = x - 1
}
assert(r >= x)
```

{((r,x),(r',x')) | (r'= 3(x'-x) /\ x' >= 0 /\ x-x' > 0) \/
(r = 0 /\ r'= 0 /\ x' = x /\ x <= 0)}

Try to prove the validity of:

((r'= 3(x-x') /\ x' >= 0 /\ x-x' > 0) \/ (r = 0 /\ r'= 0 /\ x' = x /\ x <= 0)) →
r' >= x'

The program assertion holds if and only if the formula is valid.

Recall the Simple Language

x = T ✓
if (F) c1 else c2 → (assume(F) ;° c1) ∪ (assume(¬F) ; c2)
c1 ;° c2 ✓
while (F) c1

Δ_F ∘ ∪ $\Delta_{\neg F}$

ordinary control structures

terms like in P.A.

c ::= x=T | (if (F) c else c) | c ; c | (while (F) c)
T ::= K | V | (T + T) | (T - T) | (K * T) | (T / K) | (T % K)
F ::= (T==T) | (T < T) | (T > T) | (~F) | (F && F) | (F || F)
V ::= x | y | z | ...
K ::= 0 | 1 | 2 | ...

Boolean terms like
P.A. formulas without quantifiers

Normal form for Loop-Free Programs

Lemma: Let P be a program without loops. Then for some natural number n,

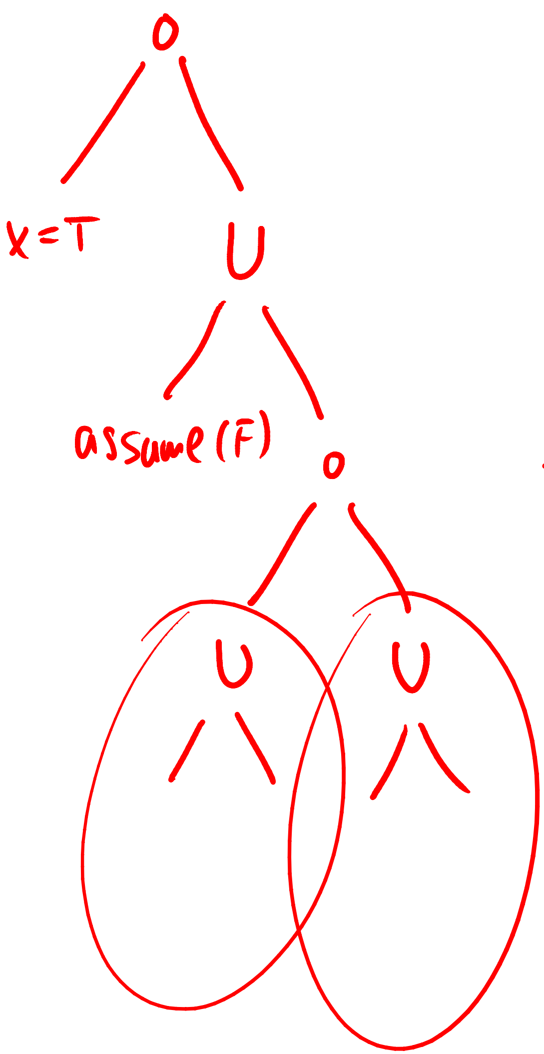
$$[\![P]\!] = \bigcup_{i=1}^{n} p_i$$

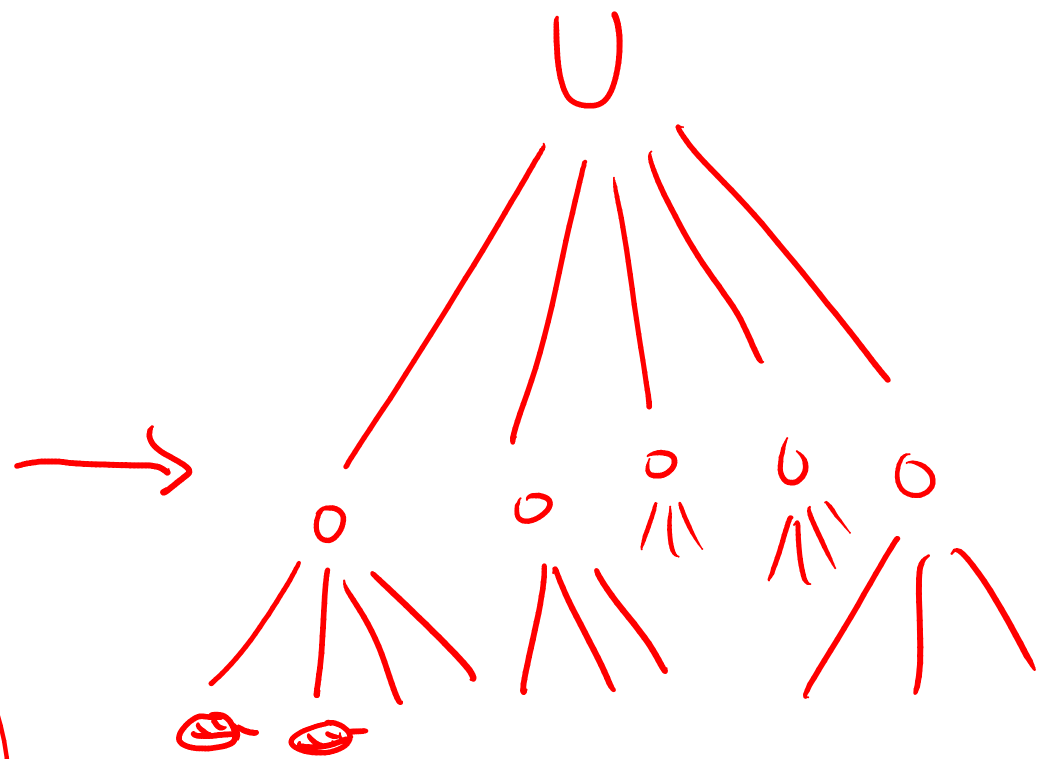
where each p_i is relation composition of

– relations for assignments

– Partial diagonal relations (assumes) $\Delta_{\text{"F"}}$

Prove this.





$$(r_1 \cup r_2) \circ (s_1 \cup s_2)$$
$$= (r_1 \circ s_1) \cup (r_1 \circ s_2) \cup (r_2 \circ s_1) \cup (r_2 \circ s_2)$$