Auxiliary Variables

\[
\begin{cases}
  y = 0 \\
  z = x + 2
\end{cases}
\]

\[
\begin{cases}
  x = 0 \\
  x = z + 2 \quad \text{if } x = 0
\end{cases}
\]

\[
\begin{cases}
  y = 0 \\
  z = x + 2
\end{cases}
\]

Not ifree \quad \text{not preserved by } x = x + 2

\[
\begin{cases}
  y = 0 \\
  z = x + 2
\end{cases}
\]

\[
\begin{cases}
  x = 0 \\
  x = x + 2 \quad \text{if } x = 0
\end{cases}
\]

If free? So interferes w/ q.?

\[
\begin{cases}
  y = 0 \\
  z = x + 2
\end{cases}
\]

\[
\begin{cases}
  y = 0 \\
  z = x + 2
\end{cases}
\]

\[
\begin{cases}
  y = 0 \\
  z = x + 2
\end{cases}
\]

\[
\begin{cases}
  y = 0 \\
  z = x + 2
\end{cases}
\]

\[
\begin{cases}
  y = 0 \\
  z = x + 2
\end{cases}
\]

\[
\begin{cases}
  y = 0 \\
  z = x + 2
\end{cases}
\]
\[ \begin{align*}
&\begin{cases}
  y = 0 \\
  x = x_0 + 2 \\
  x = x_0 \land (x = 0 \lor x = 2)
\end{cases} \\
&x_i = x + 2
\end{align*} \]

\[ \begin{align*}
&\begin{cases}
  y = 0 \\
  x = x_0 + 2 \\
  x = 2
\end{cases} \\
&\begin{cases}
  x = 2 \\
  x = x_0 + 4 \\
  y = x_0 \land (x = 0 \lor x = 2)
\end{cases} \\
&\text{Not true}
\end{align*} \]

Problem is we need to know if the other thread has run or not.

\[ \begin{align*}
&\begin{cases}
  T \land \text{inc} = F \\
  \begin{cases}
    x = 0 \\
    (x_i = x + 2; \text{inc} = T)
  \end{cases}
\end{cases} \\
&\begin{cases}
  T \\
  x_i = 0 \\
  x = 0 \lor (\text{inc} \land x = 2)
\end{cases} \\
&\begin{cases}
  \text{inc} \\
  (x_i = x_0 + 2; \text{inc} = T)
\end{cases}
\end{align*} \]

Adding flag

\text{inc} lets us

\text{inc} doesn't need to include info about inc iff \( x = x_0 + 2 \lor x = 0 \lor x = 2 \)

show correctness

but adds computation to the pgm.

\text{inc} — not a logical variable (appears in pgm)

but also not much of a pgm var

(not relevant for value of x)
inc is an auxiliary variable used in program but only for proof purposes not for doing desired calculation.

Forward:
\( \{ p \_ x = x_{0} \} x := e \{ \exists l \_ x \} x = e[\_ x_{0} / x]_{3} \)

Logical:
\( \{ x_{0} := x \} \{ p \_ x = x_{0} \} x := e \) etc.

Would make \( x_{0} \) auxiliary — program not used computationally to calc \( x \).

delta \( x \) to

Example 1 uses a var to show termination of a loop, but it doesn't affect computation of \( x, f(x) \).

Auxiliary var usage:
- add to pgm + pf to get correctness
- show that removing them doesn't affect computation of non-auxiliary vars
- remove them from pgm

How to show this?
Take a page vars, split them into 2 parts

\( V-A \), \( V-A \)

Required vars \( \rightarrow \) auxiliary

\( V-A \) supposed to contain all vars whose values we're interested in or need

\[ \text{if } x > 0 \text{ then important } = 17 \text{ f} \]

we want this?

\[ \text{Need } x \text{ too} \]

Computations of vars in \( V-A \) must depend only on vars in \( V-A \) (not in \( A \))

If we want \( V \) and can ignore actual computations of these

Some vars appear in tests (if/while)

Those can't be in \( A \)

Primary vars \( \text{Needed (indirectly)} \)

Required vars \( \rightarrow \) \( x \text{ or} \)

\( \text{Need directly or indirectly} \)
Easy marking technique

\[ (v) := r + (w) \]

Any var can use aux vars ~ required vars
\[ r' := r \]

~ req var := ~ req var ~ makes this

Disallowed:
\[ (v) \text{ calc required using auxiliary if/while } \]
\[ r := (a) a, \text{ test uses aux.} \]

\[ \begin{align*}
  \{ \mathcal{P} \}(x_0) &= \xi \mid \pi \mathcal{x} = \pi_0 \mid \xi = e + \mathcal{P}[x_0/x] \\
  \text{we want } x_0 \uparrow \text{ not used in p,q} \end{align*} \]

Auxiliary labeling \((x_0)\) is consistent

\[ \begin{cases}
  \xi = \xi, \quad \pi \mathcal{x} = \pi_0, \quad \xi \mathcal{f}(x) \mid \mathcal{d} x = x(x_0) \mid \mathcal{d} x = f(x_0) - x_0 \\
  \text{Want } x_0 \text{ and } dx \text{ auxiliary } \xi \text{ on rhs but } dx \text{ on lhs. } \xi \\
  \text{x on left, only } x \text{ on rhs ... labeling } x_0 \text{ and } dx \text{ consistent} \end{cases} \]
Can expand labeling by taking asgts of form \( y = u + v \) and labeling \( x \):

\[
(x)_i = u - (v)_i
\]

Repeat

Then aux labeling is acceptable if no aux vars appear in tests.

\[
x_i = y_i; \quad y_s = v + w; \quad \text{if } w > 0 \text{ then } x_s = x + 1; \quad w_s = w - 1
\]

Is \((x)\) consistent?

\[
(x)_i = y_i; \quad y_s = v + w; \quad \text{if } w > 0 \text{ then } (x)_s = (x) + 1; \quad w_s = w - 1
\]

Yes

Is \((y)\) consistent? No

\[
x_s = (y)_s; \quad (y)_i = v + w; \quad \text{if } w > 0 \text{ then } x_s = x + 1; \quad w_s = w - 1
\]

But \((x), (y)\) is consistent

\[
y_s = (v) + w
\]

Similarly, if \( u \cdot \text{aux vars} \), \((y)_s = (u) + w\)

So need \((y)\)

So need \((x)\)

\((x), (y), (v)\) consistent

Can't use \((w)\) because \( w \) appears in \( \text{if } w > 0 \).
The labelings \( \{ x, 3 \}, \{ x, y, 3 \}, \{ x, y, v \} \) again.

Vars are ok, no other combination is.

Candidate code w/ any labeling:

\[
\begin{align*}
(x) &:= y; \\
(y) &:= v+w; \text{ if } w > 0 \text{ then } (y) := (x) + 1; \\
&\quad \text{w} := w - 1
\end{align*}
\]

Replace w/ skip, optimize:

\[
y := v \text{ if } w > 0 \text{ then } w := w - 1 \text{ f}.
\]