Now, \((**)\) is proved by case distinction on the edge labels \(lab\).

Let \(s = (\rho, \mu) \Delta D\). In particular, \(\bot \neq D : \text{Vars} \rightarrow \mathbb{Z}^+\)

Case \(x = e_1\):

\[\begin{align*}
\rho_1 &= \rho \oplus \{x \mapsto [e_1]^T\mu\} \\
D_1 &= D \oplus \{x \mapsto [e_1]^T D\}
\end{align*}\]

\[\implies (\rho_1, \mu_1) \Delta D_1\]

Case \(x = M[e_2]\):

\[\begin{align*}
\rho_1 &= \rho \\
\mu_1 &= \mu \oplus \{[e_1]^T \rho \mapsto [e_2]^T \rho\}
\end{align*}\]

\[D_1 = D\]

\[\implies (\rho_1, \mu_1) \Delta D_1\]
Case $\text{Neg}(e)$:

$(q_l, \mu_1) = s$

where:

\[
0 = [e] \rho \Delta [e] D
\]

\[
0 \subseteq [e] D
\]

\[
\perp \neq D_1 = D
\]

\[
(r_l, \mu_1) \Delta D_1
\]

To prove \((*)\), we show for every edge \(k\):

\[
\Delta D
\]

\[
[k] \Delta
\]

\[
[k]^2
\]

\[
D
\]

\[
D_1
\]

Then \((*)\) follows by induction.

Case $\text{Pos}(e)$:

$(r_l, \mu_1) = s$

where:

\[
0 \neq [e] \rho
\]

\[
\Delta [e] D
\]

\[
0 \neq [e]^2 D
\]

\[
\perp \neq D_1 = D
\]

\[
(r_l, \mu_1) \Delta D_1
\]

We show:

\((*)\)

If $s \Delta D$ and $[\pi] s$ is defined, then:

\[
([\pi] s) \Delta ([\pi]^2 D)
\]
We conclude: The assertion $(*)$ is true.

The MOP-Solution:

$$D^*[v] = \bigcup\{[[\pi]]^2 D_\pi \mid \pi : \text{start} \rightarrow^* v\}$$

where $D_\pi x = \top \quad (x \in \text{Vars})$.

By $(*)$, we have for all initial states $s$ and all program executions $\pi$ which reach $v$:

$$([\pi]_s) \Delta \ (D^*[v])$$

In order to approximate the MOP, we use our constraint system ...

---

We conclude: The assertion $(*)$ is true.

The MOP-Solution:

$$D^*[v] = \bigcup\{[[\pi]]^2 D_\pi \mid \pi : \text{start} \rightarrow^* v\}$$

where $D_\pi x = \top \quad (x \in \text{Vars})$.

By $(*)$, we have for all initial states $s$ and all program executions $\pi$ which reach $v$:

$$([\pi]_s) \Delta \ (D^*[v])$$

---

Example

```
x = 10;
y = 1;
```

```latex
\begin{tikzpicture}
  \node (1) at (0,0) {0};
  \node (2) at (1,0) {1};
  \node (3) at (2,0) {2};
  \node (4) at (2,-1) {3};
  \node (5) at (2,-2) {4};
  \node (6) at (3,-1) {5};
  \node (7) at (3,0) {6};
  \draw [->] (1) -- node [above] {$x = 10;$} (2);
  \draw [->] (2) -- node [above] {$y = 1;$} (3);
  \draw [->] (3) -- node [above] {Pos($x > 1$)} (4);
  \draw [->] (4) -- node [above] {Neg($x > 1$)} (5);
  \draw [->] (5) -- node [above] {Pos($x > 1$)} (6);
  \draw [->] (6) -- node [above] {Neg($x > 1$)} (7);
\end{tikzpicture}
```
Example

```
0
x = 10;

1
y = 1;

2
Neg(x > 1)

3
Pos(x > 1)

4
y = x + y;

5
M[R] = y;

6
y = x - 1;

7
```

Example

```
0
x = 10;

1
y = 1;

2
Neg(x > 1)

3
Pos(x > 1)

4
y = x + y;

5
M[R] = y;

6
y = x - 1;

7
```

Example

```
0
x = 10;

1
y = 1;

2
Neg(x > 1)

3
Pos(x > 1)

4
y = x + y;

5
M[R] = y;

6
y = x - 1;

7
```

Example

```
0
x = 10;

1
y = 1;

2
Neg(x > 1)

3
Pos(x > 1)

4
y = x + y;

5
M[R] = y;

6
y = x - 1;

7
```

Conclusion

Although we compute with concrete values, we fail to compute everything ...

The fixpoint iteration, at least, is guaranteed to terminate:

For n program points and m variables, we maximally need: 
$n \cdot (m + 1)$ rounds.

Caveat

The effects of edge are not distributive !!!
Example

\begin{center}
\begin{tabular}{c|c|c|c}
  & 1 & 2 & 3 \\
\hline
x & y & x & y \\
0 & T & T & T \\
1 & 10 & 10 & T \\
2 & 10 & 1 & T \\
3 & 10 & 1 & T \\
4 & 10 & 10 & T \\
5 & 9 & 10 & T \\
6 & \perp & \perp & T \\
7 & \perp & \perp & T \\
\end{tabular}
\end{center}

Conclusion

Although we compute with concrete values, we fail to compute everything ...

The fixpoint iteration, at least, is guaranteed to terminate:

For \( n \) program points and \( m \) variables, we maximally need: \( n \cdot (m + 1) \) rounds.

Caveat

The effects of edge are not distributive !!!

Counter Example: \( f = [x = x + y]^t \)

Let \( D_1 = \{x \mapsto 2, y \mapsto 3\} \) \( D_2 = \{x \mapsto 3, y \mapsto 2\} \)

\begin{align*}
\text{Dann } f D_1 \cup f D_2 &= \{x \mapsto 5, y \mapsto 3\} \cup \{x \mapsto 5, y \mapsto 2\} \\
&= \{x \mapsto 5, y \mapsto T\} \\
&\neq \{x \mapsto T, y \mapsto T\} \\
&= f \{x \mapsto T, y \mapsto T\} \\
&= f (D_1 \cup D_2)
\end{align*}
We conclude:

The least solution $D$ of the constraint system in general yields only an **upper approximation** of the MOP, i.e.,

$$D^*[v] \subseteq D[v]$$

As an upper approximation, $D[v]$ nonetheless describes the result of every program execution $\pi$ which reaches $v$:

$$([\pi](\rho, \mu)) \Delta (D[v])$$

whenever $[\pi](\rho, \mu)$ is defined.

---

Transformation 4:

**Removal of Dead Code**

$$D[u] = \bot$$

**Extensions**

- Instead of complete right-hand sides, also subexpressions could be simplified:

  $$x + (3 \ast y) \xrightarrow{(x \rightarrow T, y \rightarrow 5)} x + 15$$

  ... and further simplifications be applied, e.g.:

  $$x \ast 0 \rightarrow 0$$
  $$x \ast 1 \rightarrow x$$
  $$x + 0 \rightarrow x$$
  $$x - 0 \rightarrow x$$
  ...
Extensions

- Instead of complete right-hand sides, also subexpressions could be simplified:

\[ x + (3 \cdot y) \xrightarrow{\{x \rightarrow y + 5\}} x + 15 \]

... and further simplifications be applied, e.g.:

- So far, the information of conditions has not yet be optimally exploited:

\[
\begin{align*}
  x = 0 & \implies 0 \\
  x = 1 & \implies x \\
  x + 0 & \implies x \\
  x - 0 & \implies x \\
  \ldots
\end{align*}
\]

Even if the value of \( x \) before the if statement is unknown, we at least know that \( x \) definitely has the value 7 — whenever the then-part is entered.

Therefore, we can define:

\[
[P = (x == e)] D = \begin{cases} 
  D & \text{if } [x == e] D = 1 \\
  \bot & \text{if } [x == e] D = 0 \\
  D_1 & \text{otherwise}
\end{cases}
\]

where

\[
D_1 = D \oplus \{ x \mapsto (D \land [e] D) \}
\]
The effect of an edge labeled \( \text{Neg}(x \neq e) \) is analogous.

Our Example

\[
\begin{array}{c}
\text{Neg}(x == 7) & \text{Pos}(x == 7) \\
\downarrow & \downarrow \\
1 & 2 \\
\text{y = x + 3;}
\end{array}
\]

The effect of an edge labeled \( \text{Neg}(x \neq e) \) is analogous.

Our Example

\[
\begin{array}{c}
\text{Neg}(x == 7) & \text{Pos}(x == 7) \\
\downarrow & \downarrow \\
1 & 2 \\
\text{y = x + 3;}
\end{array}
\]

1.5 Interval Analysis

Observation

- Programmers often use global constants for switching debugging code on/off.
  
  \[\Rightarrow\] Constant propagation is useful!

- In general, precise values of variables will be unknown — perhaps, however, a tight interval!!
Example

```plaintext
for (i = 0; i < 12; i++)
    if (i < 4)
        A1 = A + i;
    M[A1] = i;

// A start address of an array
// if the array-bound check
```

The inner check is superfluous.

Idea 1

Determine for every variable \( x \) an (as tight as possible) interval of possible values:

\[
I = \{ [l, u] \mid l \in \mathbb{Z} \cup \{-\infty\}, u \in \mathbb{Z} \cup \{+\infty\}, l \leq u \}
\]

Partial Ordering:

\[
[l_1, u_1] \subseteq [l_2, u_2] \quad \text{iff} \quad l_2 \leq l_1 \land u_1 \leq u_2
\]

Thus:

\[
[l_1, u_1] \cup [l_2, u_2] = [l_1 \cap l_2, u_1 \cup u_2]
\]

Thus:

\[
[l_1, u_1] \cap [l_2, u_2] = [l_1 \cap l_2, u_1 \cap u_2]
\]

\[
[l_1, u_1] \setminus [l_2, u_2] = [l_1 \setminus l_2, u_1 \setminus u_2]
\]

whenever \((l_1 \cup l_2) \leq (u_1 \cup u_2)\)
Caveat

\[ \rightarrow \quad I \text{ is not a complete lattice!} \]
\[ \rightarrow \quad I \text{ has infinite ascending chains, e.g.,} \]
\[ [0, 0] \subset [0, 1] \subset [-1, 1] \subset [-1, 2] \subset \ldots \]

Description Relation:
\[ z \Delta [l, u] \quad \text{iff} \quad l \leq z \leq u \]

Concretization:
\[ \gamma [l, u] = \{ z \in \mathbb{Z} \mid l \leq z \leq u \} \]

Example

\[ \gamma [0, 7] = \{0, \ldots, 7\} \]
\[ \gamma [0, \infty] = \{0, 1, 2, \ldots\} \]

Computing with intervals: Interval Arithmetic

Addition:
\[ [l_1, u_1] + [l_2, u_2] = [l_1 + l_2, u_1 + u_2] \quad \text{where} \]
\[ -\infty + _{-} = -\infty \]
\[ +\infty + _{-} = +\infty \]
\[ // -\infty + \infty \text{ cannot occur} \]

Caveat

\[ \rightarrow \quad I \text{ is not a complete lattice!} \]
\[ \rightarrow \quad I \text{ has infinite ascending chains, e.g.,} \]
\[ [0, 0] \subset [0, 1] \subset [-1, 1] \subset [-1, 2] \subset \ldots \]

Description Relation:
\[ z \Delta [l, u] \quad \text{iff} \quad l \leq z \leq u \]

Concretization:
\[ \gamma [l, u] = \{ z \in \mathbb{Z} \mid l \leq z \leq u \} \]
**Negation:**
\[-\bar{[l, u]} = [-u, -l]\]

**Multiplication:**
\[[l_1, u_1] \times [l_2, u_2] = [a, b]\]
where
\[a = \frac{l_1 l_2 l_2 u_2 + u_1 l_2 u_2 l_2 u_2}{l_1 l_2 l_2 u_2 l_2 u_2 l_2 u_2}\]
\[b = \frac{l_1 l_2 l_2 u_2 l_2 u_2 l_2 u_2}{l_1 l_2 l_2 u_2 l_2 u_2 l_2 u_2}\]

**Example**
\[[0, 2] \times [3, 4] = [0, 8]\]
\[[-1, 2] \times [3, 4] = [-4, 8]\]
\[[-1, 2] \times [-3, 4] = [-6, 8]\]
\[[-1, 2] \times [-4, -3] = [-8, 4]\]

**Division:**
\[[l_1, u_1] / [l_2, u_2] = [a, b]\]
- If 0 is not contained in the interval of the denominator, then:
  \[a = \frac{l_1 l_2 l_1 / u_2}{l_2 / u_2 l_1 / u_2}\]
  \[b = l_1 / l_2 l_1 / u_2 l_1 / u_2\]
- If: \(l_2 \leq 0 \leq u_2\), we define:
  \([a, b] = [-\infty, +\infty]\)

\[
\left\\{\begin{array}{l}
[2, 4] = \left(\left\\{\frac{2}{\bar{1}, 4}\right\\} / \left\\{\frac{2}{\bar{1}, 4}\right\}\right)
\end{array}\right\}
\]

**Equality:**
\[[l_1, u_1] = [l_2, u_2] = \begin{cases} 
[1, 1] & \text{if } l_1 = u_1 = l_2 = u_2 \\
[0, 0] & \text{if } u_1 < l_2 \lor u_2 < l_1 \\
[0, 1] & \text{otherwise} 
\end{cases}\]

**Example**
\[\{42, 42\} = \{42, 42\}\]
\[\{0, 4\} = \{0, 4\}\]
\[\{0, 1\} = \{0, 1\}\]
Less:

\[
[t_1, u_1] \leq [t_2, u_2] = \begin{cases} [1, 1] & \text{if } u_1 < t_2 \\ [0, 0] & \text{if } u_2 \leq t_1 \\ [0, 1] & \text{otherwise} \end{cases}
\]

Example:

\[
[1.2] \leq [9, 42] = [1, 1] \\
[0.7] \leq [0.7] = [0, 1] \\
[3, 4] \leq [1, 2] = [0, 0]
\]