\[
[x = M[e];]\# A = (A \cup \{e\}) \setminus \text{Expr}_x
\]
\[
[M[e_1] = e_2;]\# A = A \cup \{e_1, e_2\}
\]

By that, every path can be analyzed :-) 
A given program may admit several paths :-(
For any given input, another path may be chosen :-((

\[\longrightarrow\] We require the set:

\[
\mathcal{A}[v] = \bigcap \{[[\pi]]\# \emptyset \mid \pi : \text{start} \rightarrow^* v\}
\]
Concretely:

→ We consider all paths $\pi$ which reach $v$.
→ For every path $\pi$, we determine the set of expressions which are available along $\pi$.
→ Initially at program start, nothing is available :-)
→ We compute the intersection $\implies$ safe information
Concretely:

→ We consider all paths $\pi$ which reach $v$.
→ For every path $\pi$, we determine the set of expressions which are available along $\pi$.
→ Initially at program start, nothing is available :-) 
→ We compute the intersection $\implies$ safe information

How do we exploit this information ???
Transformation 1.1:

We provide novel registers $T_e$ as storage for the $e$:

$$x = e;$$

$$x = T_e;$$
Transformation 1.1:

We provide novel registers $T_e$ as storage for the $e$:
... analogously for $R = M[e]$; and $M[e_1] = e_2$.

Transformation 1.2:

If $e$ is available at program point $u$, then $e$ need not be re-evaluated:

We replace the assignment with $Nop$  :-)
Example:

\[ x = y + 3; \]
\[ x = 7; \]
\[ z = y + 3; \]
Example:

\[ x = y + 3; \]
\[ x = 7; \]
\[ z = y + 3; \]

\[ T = y + 3; \]
\[ x = T; \]
\[ x = 7; \]

\[ T = y + 3; \]
\[ z = T; \]
Example:

\[
\begin{align*}
x &= y + 3; \\
x &= 7; \\
z &= y + 3;
\end{align*}
\]
Example:

\[
\begin{align*}
x &= y + 3; \\
x &= 7; \\
z &= y + 3; \\
T &= y + 3;
\end{align*}
\]
Correctness: (Idea)

Transformation 1.1 preserves the semantics and $A[u]$ for all program points $u$ :-)

Assume $\pi : start \rightarrow^* u$ is the path taken by a computation. If $e \in A[u]$, then also $e \in [[\pi]]^\# \emptyset$.

Therefore, $\pi$ can be decomposed into:

![Diagram of path decomposition](image)

with the following properties:
• The expression $e$ is evaluated at the edge $k$;
• The expression $e$ is not removed from the set of available expressions at any edge in $\pi_2$, i.e., no variable of $e$ receives a new value. :-)

53
• The expression $e$ is evaluated at the edge $k$;
• The expression $e$ is not removed from the set of available expressions at any edge in $\pi_2$, i.e., no variable of $e$ receives a new value $\quad :-)$

$\implies$

The register $T_e$ contains the value of $e$ whenever $u$ is reached $\quad :-))$
Warning:

Transformation 1.1 is only meaningful for assignments $x = e$; where:

$\rightarrow e \notin Vars$;

$\rightarrow$ the evaluation of $e$ is non-trivial \( :-} \)
Warning:

Transformation 1.1 is only meaningful for assignments \( x = e \); where:

\[ x \notin Vars(e) ; \]

\[ e \notin Vars ; \]

\[ \text{the evaluation of } e \text{ is non-trivial} \quad :- \}

Which leaves us with the following question ...
Question:

How do we compute $A[u]$ for every program point $u$?
Question:

How can we compute $A[u]$ for every program point $u$?

We collect all restrictions to the values of $A[u]$ into a system of constraints:

$$A[start] \subseteq \emptyset$$

$$A[v] \subseteq \lbrack k \rbrack^\# (A[u])$$

$k = (u, _, v)$ edge
Wanted:

- a maximally large solution (??)
- an algorithm which computes this :-)

Example:
Wanted:

- a maximally large solution
- an algorithm which computes this

Example:

\[
\begin{align*}
\text{Pos}(x > 1) & \quad \text{Neg}(x > 1) \\
0 & \quad 1 \\
y = 1; & \quad y = x \times y; \\
\end{align*}
\]

\[
\begin{align*}
\mathcal{A}[0] & \subseteq \emptyset \\
x = x - 1; & \\
3 & \\
4 & \\
5 & \\
2 & \\
1 & \\
0 &
\end{align*}
\]
Wanted:

- a maximally large solution (??)
- an algorithm which computes this :-) 

Example:

\[ y = 1; \]
\[ x = x - 1; \]
\[ y = x \times y; \]
\[ \mathcal{A}[0] \subseteq \emptyset \]
\[ \mathcal{A}[1] \subseteq (\mathcal{A}[0] \cup \{1\}) \setminus \text{Expr}_y \]
\[ \mathcal{A}[1] \subseteq \mathcal{A}[4] \]
Wanted:

- a maximally large solution (??)
- an algorithm which computes this :-)

Example:

\[ \begin{align*}
A[0] & \subseteq \emptyset \\
A[1] & \subseteq (A[0] \cup \{1\}) \setminus Expr_y \\
\end{align*} \]
Wanted:

- a maximally large solution (??)
- an algorithm which computes this :-)

Example:

\[\begin{align*}
A[0] & \subseteq \emptyset \\
A[1] & \subseteq (A[0] \cup \{1\}) \setminus \text{Expr}_y \\
A[3] & \subseteq (A[2] \cup \{x \times y\}) \setminus \text{Expr}_y
\end{align*}\]
Wanted:

- a maximally large solution (??)
- an algorithm which computes this :-) 

Example:

\[ y = 1; \]
\[ x = x - 1; \]
\[ y = x \times y; \]
\[ x = x - 1; \]

\[ A[0] \subseteq \emptyset \]
\[ A[1] \subseteq (A[0] \cup \{1\}) \setminus Expr_y \]
\[ A[2] \subseteq A[1] \cup \{x > 1\} \]
\[ A[3] \subseteq (A[2] \cup \{x \times y\}) \setminus Expr_y \]
Wanted:

- a maximally large solution (??)
- an algorithm which computes this :-)

Example:

\[
\begin{align*}
\mathcal{A}[0] & \subseteq \emptyset \\
\mathcal{A}[1] & \subseteq (\mathcal{A}[0] \cup \{1\}) \setminus \text{Expr}_y \\
\mathcal{A}[2] & \subseteq \mathcal{A}[1] \cup \{x > 1\} \\
\mathcal{A}[3] & \subseteq (\mathcal{A}[2] \cup \{x \ast y\}) \setminus \text{Expr}_y \\
\mathcal{A}[4] & \subseteq (\mathcal{A}[3] \cup \{x - 1\}) \setminus \text{Expr}_x \\
\mathcal{A}[5] & \subseteq \mathcal{A}[1] \cup \{x > 1\}
\end{align*}
\]
Wanted:

- a maximally large solution (??)
- an algorithm which computes this :-) 

Example:

Solution:

\[ A[0] = \emptyset \]

\[ A[1] = \{1\} \]

\[ A[2] = \{1, x > 1\} \]

\[ A[3] = \{1, x > 1\} \]

\[ A[4] = \{1\} \]

\[ A[5] = \{1, x > 1\} \]
Observation:

- The possible values for $A[u]$ form a complete lattice:

$$
\mathbb{D} = 2^{Expr} \quad \text{with} \quad B_1 \sqsubseteq B_2 \quad \text{iff} \quad B_1 \sqsupseteq B_2
$$
Observation:

- The possible values for $A[u]$ form a complete lattice:

$$\mathbb{D} = 2^{Expr} \quad \text{with} \quad B_1 \sqsubseteq B_2 \quad \text{iff} \quad B_1 \supseteq B_2$$

- The functions $\lfloor k \rfloor^\#: \mathbb{D} \to \mathbb{D}$ are monotonic, i.e.,

$$\lfloor k \rfloor^\#(B_1) \sqsubseteq \lfloor k \rfloor^\#(B_2) \quad \text{iff} \quad B_1 \sqsubseteq B_2$$
Background 2: Complete Lattices

A set \( \mathbb{D} \) together with a relation \( \subseteq \subseteq \mathbb{D} \times \mathbb{D} \) is a partial order if for all \( a, b, c \in \mathbb{D} \),

\[
\begin{align*}
   a & \subseteq a & \text{reflexivity} \\
   a \subseteq b \land b \subseteq a & \implies a = b & \text{anti–symmetry} \\
   a \subseteq b \land b \subseteq c & \implies a \subseteq c & \text{transitivity}
\end{align*}
\]

Examples: 

1. \( \mathbb{D} = 2^{\{a, b, c\}} \) with the relation “\( \subseteq \)”: 

![Diagram](image-url)
2. \( \mathbb{Z} \) with the relation “=”:

\[ \cdots -2 -1 0 1 2 \cdots \]

3. \( \mathbb{Z} \) with the relation “\( \leq \)”:

\[ \cdots -1 0 1 2 \cdots \]

4. \( \mathbb{Z}_\perp = \mathbb{Z} \cup \{\perp\} \) with the ordering:

\[ \cdots -2 -1 0 1 2 \cdots \]
$d \in \mathbb{D}$ is called upper bound for $X \subseteq \mathbb{D}$ if

$$x \sqsubseteq d \quad \text{for all } x \in X$$

Caveat:

• has no upper bound!
• has the upper bounds
\( d \in \mathbb{D} \) is called upper bound for \( X \subseteq \mathbb{D} \) if

\[
x \sqsubseteq d \quad \text{for all } x \in X
\]

\( d \) is called least upper bound (lub) if

1. \( d \) is an upper bound and
2. \( d \sqsubseteq y \) for every upper bound \( y \) of \( X \).
$d \in \mathbb{D}$ is called upper bound for $X \subseteq \mathbb{D}$ if

$$x \sqsubseteq d \quad \text{for all } x \in X$$

$d$ is called least upper bound (lub) if

1. $d$ is an upper bound and
2. $d \sqsubseteq y$ for every upper bound $y$ of $X$.

Caveat:

- $\{0, 2, 4, \ldots\} \subseteq \mathbb{Z}$ has no upper bound!
- $\{0, 2, 4\} \subseteq \mathbb{Z}$ has the upper bounds $4, 5, 6, \ldots$
A complete lattice (cl) \( \mathbb{D} \) is a partial ordering where every subset \( X \subseteq \mathbb{D} \) has a least upper bound \( \bigcup X \in \mathbb{D} \).

**Note:**

Every complete lattice has

\(\rightarrow\) a least element \( \bot = \bigcup \emptyset \in \mathbb{D} \);

\(\rightarrow\) a greatest element \( \top = \bigcup \mathbb{D} \in \mathbb{D} \).
Examples:

1. $\mathbb{D} = 2\{a,b,c\}$ is a cl :-)

2. $\mathbb{D} = \mathbb{Z}$ with “=” is not.

3. $\mathbb{D} = \mathbb{Z}$ with “≤” is neither.

4. $\mathbb{D} = \mathbb{Z}_\bot$ is also not :-(

5. With an extra element $\top$, we obtain the flat lattice

\[ \mathbb{Z}_\top = \mathbb{Z} \cup \{\bot, \top\} : \]

\[
\begin{array}{cccccc}
\top & & & & & \\
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \\
\vcenter{\hbox{\vdots}} & -2 & -1 & 0 & 1 & 2 & \vcenter{\hbox{\vdots}} \\
\downarrow & \downarrow & \downarrow & \downarrow & \downarrow & \\
\bot & & & & & \\
\end{array}
\]
We have:

**Theorem:**

If \( \mathbb{D} \) is a complete lattice, then every subset \( X \subseteq \mathbb{D} \) has a greatest lower bound \( \bigwedge X \).
We have:

**Theorem:**

If $D$ is a complete lattice, then every subset $X \subseteq D$ has a greatest lower bound $\bigwedge X$.

**Proof:**

Construct $U = \{ u \in D \mid \forall x \in X : u \sqsubseteq x \}$.  
// the set of all lower bounds of $X$  :-)


We have:

**Theorem:**
If \( \mathbb{D} \) is a complete lattice, then every subset \( X \subseteq \mathbb{D} \) has a greatest lower bound \( \bigcap X \).

**Proof:**
Construct \( U = \{ u \in \mathbb{D} | \forall x \in X : u \sqsubseteq x \} \).
// the set of all lower bounds of \( X \)  :-)

Set: \( g := \bigcup U \)
Claim: \( g = \bigcap X \)
(1) **$g$ is a lower bound of $X$:**

Assume $x \in X$. Then:

- $u \subseteq x$ for all $u \in U$
- $\implies x$ is an upper bound of $U$
- $\implies g \subseteq x$  :-)

---

$(2)$

$g$ is the greatest lower bound of $X$:

Assume $u$ is a lower bound of $X$. Then:

- $u \subseteq x$ for all $u \in U$
- $\implies x$ is an upper bound of $U$
- $\implies g \subseteq x$  :-)

---
(1) \( g \) is a **lower bound** of \( X \):

Assume \( x \in X \). Then:

\[
\begin{align*}
u \subseteq x \text{ for all } u \in U \\
\implies x \text{ is an upper bound of } U \\
\implies g \subseteq x \quad :-(
\end{align*}
\]

(2) \( g \) is the **greatest lower bound** of \( X \):

Assume \( u \) is a lower bound of \( X \). Then:

\[
\begin{align*}
u \in U \\
\implies u \subseteq g \quad :-))
\end{align*}
\]
We are looking for solutions for systems of constraints of the form:

\[ x_i \sqsubseteq f_i(x_1, \ldots, x_n) \]  

(\star)
We are looking for solutions for systems of constraints of the form:

\[ x_i \supseteq f_i(x_1, \ldots, x_n) \quad (\ast) \]

where:

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
<th>Here</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_i )</td>
<td>unknown</td>
<td>( A[u] )</td>
</tr>
<tr>
<td>( \mathbb{D} )</td>
<td>values</td>
<td>( 2^{Expr} )</td>
</tr>
<tr>
<td>( \subseteq \subseteq \mathbb{D} \times \mathbb{D} )</td>
<td>ordering relation</td>
<td>( \supseteq )</td>
</tr>
<tr>
<td>( f_i : \mathbb{D}^n \rightarrow \mathbb{D} )</td>
<td>constraint</td>
<td>...</td>
</tr>
</tbody>
</table>
We are looking for solutions for systems of constraints of the form:

\[ x_i \sqsupseteq f_i(x_1, \ldots, x_n) \quad (*) \]

where:

<table>
<thead>
<tr>
<th>( x_i )</th>
<th>unknown</th>
<th>here: ( \mathcal{A}[u] )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathcal{D} )</td>
<td>values</td>
<td>here: ( 2^{Expr} )</td>
</tr>
<tr>
<td>( \sqsubseteq \sqsubseteq \mathcal{D} \times \mathcal{D} )</td>
<td>ordering relation</td>
<td>here: ( \supseteq )</td>
</tr>
<tr>
<td>( f_i: \mathcal{D}^n \rightarrow \mathcal{D} )</td>
<td>constraint</td>
<td>here: ...</td>
</tr>
</tbody>
</table>

Constraint for \( \mathcal{A}[v] \quad (v \neq \text{start}) \):

\[
\mathcal{A}[v] \subseteq \bigcap \{ [[k]]^\#(\mathcal{A}[u]) \mid k = (u, _, v) \text{ edge} \}
\]
We are looking for solutions for systems of constraints of the form:

\[ x_i \supseteq f_i(x_1, \ldots, x_n) \quad (\ast) \]

where:

<table>
<thead>
<tr>
<th>( x_i )</th>
<th>unknown here: ( A[u] )</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \mathbb{D} )</td>
<td>values here: ( 2^{Expr} )</td>
</tr>
<tr>
<td>( \subseteq \subseteq \mathbb{D} \times \mathbb{D} )</td>
<td>ordering relation here: ( \supseteq )</td>
</tr>
<tr>
<td>( f_i: \mathbb{D}^n \rightarrow \mathbb{D} )</td>
<td>constraint here: ...</td>
</tr>
</tbody>
</table>

Constraint for \( A[v] \quad (v \neq \text{start}) \):

\[ A[v] \subseteq \bigcap \{ [[k]]^\#(A[u]) \mid k = (u, _, v) \text{ edge} \} \]

Because:

\[ x \supseteq d_1 \land \ldots \land x \supseteq d_k \quad \text{iff} \quad x \supseteq \bigsqcup \{d_1, \ldots, d_k\} \quad :-) \]
A mapping \( f : \mathcal{D}_1 \rightarrow \mathcal{D}_2 \) is called \textbf{monotonic}, if \( f(a) \sqsubseteq f(b) \) for all \( a \sqsubseteq b \).
A mapping \( f : D_1 \rightarrow D_2 \) is called monotonic, if \( f(a) \subseteq f(b) \) for all \( a \subseteq b \).

**Examples:**

(1) \( D_1 = D_2 = 2^U \) for a set \( U \) and \( f(x) = (x \cap a) \cup b \).

Obviously, every such \( f \) is monotonic \(-:)\)
A mapping  \( f : \mathbb{D}_1 \rightarrow \mathbb{D}_2 \)  is called \textbf{monotonic}, is  \( f(a) \sqsubseteq f(b) \)  for all  \( a \sqsubseteq b \).


**Examples:**

(1)  \( \mathbb{D}_1 = \mathbb{D}_2 = 2^U \)  for a set  \( U \)  and  \( f x = (x \cap a) \cup b \).

   Obviously, every such  \( f \)  is monotonic  :-(

(2)  \( \mathbb{D}_1 = \mathbb{D}_2 = \mathbb{Z} \) (with the ordering “\( \leq \)”). Then:

   •  \( \text{inc } x = x + 1 \)  is monotonic.
   •  \( \text{dec } x = x - 1 \)  is monotonic.
A mapping $f : \mathbb{D}_1 \rightarrow \mathbb{D}_2$ is called monotonic, is $f(a) \sqsubseteq f(b)$ for all $a \sqsubseteq b$.

Examples:

(1) $\mathbb{D}_1 = \mathbb{D}_2 = 2^U$ for a set $U$ and $f x = (x \cap a) \cup b$. Obviously, every such $f$ is monotonic :-)

(2) $\mathbb{D}_1 = \mathbb{D}_2 = \mathbb{Z}$ (with the ordering “$\leq$”). Then:

- $\text{inc } x = x + 1$ is monotonic.
- $\text{dec } x = x - 1$ is monotonic.
- $\text{inv } x = -x$ is not monotonic :-)

91
Theorem:

If \( f_1 : \mathbb{D}_1 \rightarrow \mathbb{D}_2 \) and \( f_2 : \mathbb{D}_2 \rightarrow \mathbb{D}_3 \) are monotonic, then also \( f_2 \circ f_1 : \mathbb{D}_1 \rightarrow \mathbb{D}_3 \) :-)
