Transformation 1.1:

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

$\text{Pos}(e) = x = e;\quad \text{Neg}(e) = x = T_e;$

$\text{Pos}(T_e) = x = e;\quad \text{Neg}(T_e) = x = T_e;$
... 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:

\[
\begin{align*}
T_e &= e; \\
\end{align*}
\]

We replace the assignment with \textit{Nop}  :-)

\( u \)
Example:

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

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

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

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

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 :-}
• 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 :-)

\[\implies\]

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

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

$\rightarrow e \not\in Vars$;

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

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

\[ \rightarrow x \not\in Vars(e); \]
\[ \rightarrow e \not\in Vars; \]
\[ \rightarrow \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:

\[
\begin{align*}
A[start] & \subseteq \emptyset \\
A[v] & \subseteq [k]^{\#} (A[u]) \quad k = (u, _, v) \quad \text{edge}
\end{align*}
\]
Wanted:

- a maximally **large** solution  
- an algorithm which computes this  

Example:

```
0

y = 1;

1

Neg(\(x > 1\))  Pos(\(x > 1\))

5

2

y = x * y;

3

x = x - 1;

4
```
Wanted:

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

Example:

$$\begin{align*}
0 & \quad y = 1; \\
1 & \quad \text{Neg}(x > 1) \quad \text{Pos}(x > 1) \\
2 & \quad y = x \times y; \\
3 & \quad x = x - 1; \\
4 & \\
5 & \quad A[0] \subseteq \emptyset
\end{align*}$$
Wanted:

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

Example:

\[ 1 \]
\[ y = 1; \]
\[ \text{Neg}(x > 1) \]
\[ \text{Pos}(x > 1) \]
\[ 2 \]
\[ y = x \times y; \]
\[ 3 \]
\[ x = x - 1; \]
\[ 4 \]
\[ 0 \]
\[ \text{A}[0] \subseteq \emptyset \]
\[ \text{A}[1] \subseteq (\text{A}[0] \cup \{1\}) \setminus \text{Expr}_y \]
\[ \text{A}[1] \subseteq \text{A}[4] \]
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*}

\begin{align*}
0 & \quad y = 1; \\
1 & \quad \text{Neg}(x > 1) \quad \text{Pos}(x > 1) \quad y = x \ast y; \\
2 & \quad x = x - 1; \\
3 & \\
4 & \\
5 & \text{Neg}(x > 1)
\end{align*}
Wanted:

- a maximally large solution  
- an algorithm which computes this

Example:

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

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

Example:

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

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

Example:
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:

$$\mathbb{D} = 2^{Expr} \quad \text{with} \quad B_1 \sqsubseteq B_2 \quad \iff \quad B_1 \supseteq B_2$$
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 $\lbrack k \rbrack^\#: \mathbb{D} \rightarrow \mathbb{D}$ are monotonic, i.e.,

$$\lbrack k \rbrack^\#(B_1) \sqsubseteq \lbrack k \rbrack^\#(B_2) \quad \text{iff} \quad B_1 \sqsubseteq B_2$$
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 $a, b, c \in \mathbb{D}$,

- reflexivity: $a \subseteq a$
- anti-symmetry: $a \subseteq b \land b \subseteq a \implies a = b$
- transitivity: $a \subseteq b \land b \subseteq c \implies a \subseteq c$

Examples:

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

```
      a, b, c
     /  \
    /    \
  a, b  a, c
   /  \
  /    \
a    b, c
      /
     /  \
    /    \
   a    b
      /  \
     /    \
   /      \
  a
```
2. \( \mathbb{Z} \) with the relation “=”:

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

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

![Diagram for Z with ≤]

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

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

$$x \sqsubseteq d \quad \text{for all } x \in X$$
$d \in \mathbb{D}$ 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 $X$. 
\( d \in \mathbb{D} \) is called \textbf{upper bound} for \( X \subseteq \mathbb{D} \) if
\[
x \sqsubseteq d \quad \text{for all} \quad x \in X
\]

\( d \) is called \textbf{least upper bound (lub)} if
1. \( d \) is an upper bound and
2. \( d \sqsubseteq y \) for every upper bound \( y \) of \( X \).

\textbf{Caveat:}

- \( \{0, 2, 4, \ldots\} \subseteq \mathbb{Z} \) has \textbf{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. $\mathcal{D} = 2^{\{a,b,c\}}$ is a cl  :-)

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

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

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

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

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

**Theorem:**

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

Theorem:

If $\mathbb{D}$ is a complete lattice, then every subset $X \subseteq \mathbb{D}$ has a greatest lower bound $\cap X$.

Proof:

Construct $U = \{u \in \mathbb{D} \mid \forall x \in X : u \sqsubseteq x\}$.

// 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 $\cap X$.

**Proof:**

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

Set:  

$g := \bigcup U$

Claim:  

$g = \cap X$
(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 $U$

$\implies g \subseteq x$  \[ :-) \]
(1)  \( g \) is a **lower bound** of \( X \):

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

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

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

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

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

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

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

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

where:

| \( x_i \)  | unknown here: \( A[u] \) |
| \( \mathbb{D} \)  | values here: \( 2^{Expr} \) |
| \( \sqsubseteq \subseteq \mathbb{D} \times \mathbb{D} \)  | ordering relation here: \( \supseteq \) |
| \( f_i: \mathbb{D}^n \rightarrow \mathbb{D} \)  | constraint here: \( \ldots \) |

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

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

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

(*)

where:

| \(x_i\)       | unknown            | here: \(A[u]\) |
|\(D\)           | values             | here: \(2^{Expr}\) |
|\(\sqsubseteq \sqsubseteq D \times D\) | ordering relation | here: \(\supseteq\) |
|\(f_i: D^n \to D\) | constraint         | here: \(\ldots\) |

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

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

Because:

\[
x \sqsupseteq d_1 \land \ldots \land x \sqsupseteq d_k \ iff \ x \sqsupseteq \bigcup \{d_1, \ldots, d_k\} \quad :-)
\]