Application:

Assume that we have computed the value of $x + y$ at program point $u$:

$$\begin{array}{c}
\circ \quad x+y \\
\downarrow \quad \pi \\
\circ \quad \pi \\
\end{array}$$

We perform a computation along path $\pi$ and reach $v$ where we evaluate again $x + y$ ...

Idea:

If $x$ and $y$ have not been modified in $\pi$, then evaluation of $x + y$ at $v$ must return the same value as evaluation at $u$

We can check this property at every edge in $\pi$
Idea:

If $x$ and $y$ have not been modified in $\pi$, then evaluation of $x + y$ at $v$ must return the same value as evaluation at $u$.

We can check this property at every edge in $\pi$.

More generally:

Assume that the values of the expressions $A = \{e_1, \ldots, e_r\}$ are available at $u$. 
Idea:

If $x$ and $y$ have not been modified in $\pi$, then evaluation of $x + y$ at $v$ must return the same value as evaluation at $u$.

We can check this property at every edge in $\pi$.

More generally:

Assume that the values of the expressions $A = \{e_1, \ldots, e_r\}$ are available at $u$.

Every edge $k$ transforms this set into a set $[k]^# A$ of expressions whose values are available after execution of $k$. 
This transformations can be composed to the effect of a path
\[ \pi = k_1 \ldots k_r: \]
\[ [\pi]^\# = [k_r]^\# \circ \ldots \circ [k_1]^\# \]
... which transformations can be composed to the effect of a path \( \pi = k_1 \ldots k_r \):

\[
[\pi]^\# = [k_r]^\# \circ \ldots \circ [k_1]^\#
\]

The effect \( [k]^\# \) of an edge \( k = (u, lab, v) \) only depends on the label \( lab \), i.e., \( [k]^\# = [lab]^\# \)
... which transformations can be composed to the effect of a path \( \pi = k_1 \ldots k_r \):

\[
[\pi]^\# = [k_r]^\# \circ \ldots \circ [k_1]^\#
\]

\( k = (u, \text{lab}, v) \) only depends on the label \text{lab}, i.e., \( [k]^\# = [\text{lab}]^\# \)

where:

\[
\begin{align*}
[;]^\# A &= A \\
[\text{Pos}(e)]^\# A &= [\text{Neg}(e)]^\# A &= A \cup \{e\} \\
[x = e;]^\# A &= (A \cup \{e\}) \setminus \text{Expr}_x 
\end{align*}
\]

\( \text{Expr}_x \) all expressions which contain \( x \)
\[ [x = M[e];] \# A \quad = \quad (A \cup \{e\}) \backslash \text{Expr}_x \]

\[ [M[e_1] = e_2;] \# A \quad = \quad A \cup \{e_1, e_2\} \]
\[[x = M[e];]\] \# 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.

Depending on the input, different paths may be chosen.
\[ [x = M[e];] \triangledown A \; = \; (A \cup \{e\}) \setminus \text{Expr}_x \]
\[ [M[e_1] = e_2;] \triangledown A \; = \; A \cup \{e_1, e_2\} \]

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

\[ \Rightarrow \quad \text{For each node } v, \text{ we need the set:} \]
\[ A[v] \; = \; \bigcap \{[\pi] \triangledown \emptyset \mid \pi : \text{start} \to^* v\} \]
Concretely:

→ The analysis considers all paths $\pi$ that reach $v$.

→ For every path $\pi$, the analysis determines the set of expressions that are available along $\pi$.

→ Initially at program start, nothing is available

→ An expression is available at $v$ if it is available along all paths to $v$. Therefore, the analysis computes the intersection of the availability sets as 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 safe information

How do we exploit this information???
Transformation UT (unique temporaries):

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

```
x = e;
```

```
T_e = e;
```

```
x = T_e;
```
Transformation UT:

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

Transformation AEE (available expression elimination):

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

We replace the assignment with \textit{Nop}
Example:

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

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

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

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

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

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

Transformation UT 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 $e \in [\pi]^2 \emptyset$.

Therefore, $\pi$ can be decomposed into:

```
\begin{xy}
  0 *+[F]{\text{start}}; u_1 =< \pi_1 \\
  u_1 =< k \cong u_2 \\
  u_2 =< \pi_2 \\
  \end{xy}
```

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

---

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

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

$\rightarrow e \not\in Vars$;

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

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

$\rightarrow x \notin Vars(e)$;

$\rightarrow e \notin Vars$;

$\rightarrow$ the evaluation of $e$ is non-trivial

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 constraints on the values of $A[u]$ into a system of constraints:

- $A[\text{start}] \subseteq \emptyset$
- $A[v] \subseteq [k]^\circ (A[u])$

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

Why $\subseteq$?
Wanted:

- a greatest solution (??)
- an algorithm that computes this solution

Example:

```
0
  y = 1;

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

3
  x = x - 1;

4

2
  y = x * y;
```
Wanted:

- a greatest solution
- an algorithm that computes this solution

Example:
Wanted:

- a greatest solution (??)
- an algorithm that computes this solution

Example:

\[ \mathcal{A}[0] \subseteq \emptyset \]
\[ \mathcal{A}[1] \subseteq (\mathcal{A}[0] \cup \{1\}) \setminus \text{Expr}_y \]
\[ \mathcal{A}[1] \subseteq \mathcal{A}[4] \]
Wanted:

- a greatest solution (??)
- an algorithm that computes this solution

Example:

\[
\begin{align*}
A[0] & \subseteq \emptyset \\
A[1] & \subseteq (A[0] \cup \{1\}) \backslash \text{Expr}_y \\
\end{align*}
\]
Wanted:

- a **greatest** solution (??)
- an algorithm that computes this solution

Example:

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

- a greatest solution (??)
- an algorithm that computes this solution

Example:

\[
\begin{align*}
&\text{Neg}(x > 1) \\
&A[0] \subseteq \emptyset \\
&A[1] \subseteq (A[0] \cup \{1\}) \backslash Expr_y \\
\end{align*}
\]
Wanted:

- a greatest solution (??)
- an algorithm that computes this solution

Example:

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

- a greatest solution (??)
- an algorithm that computes this solution

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:

  \[ \mathcal{D} = 2^{\text{Expr}} \]  
  with  \[ B_1 \subseteq B_2 \]  
  iff  \[ B_1 \supseteq B_2 \]

  By convention:  \[ a \leq 5 \]  
  iff  \[ a \] is not less precise than 6
Observation:

- The possible values for $A[u]$ form a complete lattice:
  \[ \mathbb{D} = 2^{Expr} \quad \text{with} \quad B_1 \subseteq B_2 \quad \text{iff} \quad B_1 \supseteq B_2 \]

- The functions $[k]^\#: \mathbb{D} \rightarrow \mathbb{D}$ are monotonic, i.e.,
  \[ [k]^\#(B_1) \subseteq [k]^\#(B_2) \quad \text{iff} \quad B_1 \subseteq B_2 \]
Background 2: Complete Lattices

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

- $a \sqsubseteq a$ \hspace{1cm} \text{reflexivity}
- $a \sqsubseteq b \land b \sqsubseteq a \implies a = b$ \hspace{1cm} \text{anti-symmetry}
- $a \sqsubseteq b \land b \sqsubseteq c \implies a \sqsubseteq c$ \hspace{1cm} \text{transitivity}

Examples:

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

3. \( \mathbb{Z}_\bot = \mathbb{Z} \cup \{ \bot \} \) with the ordering:

*Flat lattice*
$d \in \mathbb{D}$ is called upper bound for $X \subseteq \mathbb{D}$ if

$$x \subseteq d \quad \text{for all } x \in X$$
\[ d \in \mathcal{D} \text{ is called upper bound for } X \subseteq \mathcal{D} \text{ if} \]

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

\[ d \text{ is called least upper bound (lub) if} \]

1. \( d \) is an upper bound and

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

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

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

1. $d$ is an upper bound and
2. $d \subseteq y$ 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 partially ordered set $\mathbb{D}$ is a **complete lattice (cl)** if 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 $\leq$ is not a cl.
3. $\mathbb{D} = \mathbb{Z}_\bot$ is also not
4. With an extra element $\top$, we obtain the flat lattice $\mathbb{Z}_\top = \mathbb{Z} \cup \{\bot, \top\}$.

\[ \begin{align*}
&\text{\ldots -2 -1 0 1 2 \ldots} \\
&\text{\ldots} \\
&\text{\ldots}
\end{align*} \]
Back to the system of constraints for Available Expressions!

\[ A[start] \subseteq \emptyset \]
\[ A[v] \subseteq \llbracket k \rrbracket^\partial (A[u]) \quad k = (u, \_, v) \text{ edge} \]

Combine all constraints for each variable \( v \) by applying the least-upper-bound operator \( \rightarrow \)

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

Correct because:

\[ x \supseteq d_1 \land \ldots \land x \supseteq d_k \quad \text{iff} \quad x \supseteq \bigcup\{d_1, \ldots, d_k\} \]
We derive a generic form of the systems of constraints:

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

relation to the running example:

| \( x_i \) | unknown | here: \( A[u] \) |
| \( D \) | values | here: \( 2^{Expr} \) |
| \( \subseteq \subseteq D \times D \) | ordering relation | here: \( \supseteq \) |
| \( f_i: D^n \rightarrow D \) | constraint | here: \( \ldots \) |
A mapping \( f : \mathbb{D}_1 \rightarrow \mathbb{D}_2 \) is called monotonic, if \( f(a) \sqsubseteq f(b) \) for all \( a \sqsubseteq b \).
A mapping \( f : \mathcal{D}_1 \rightarrow \mathcal{D}_2 \) is called monotonic, if \( f(a) \subseteq f(b) \) for all \( a \subseteq b \).

**Examples:**

(1) \( \mathcal{D}_1 = \mathcal{D}_2 = 2^U \) 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 \textit{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 \( f : \mathcal{D}_1 \rightarrow \mathcal{D}_2 \) is called **monotonic** if \( f(a) \subseteq f(b) \) for all \( a \subseteq b \).

**Examples:**

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

2. \( \mathcal{D}_1 = \mathcal{D}_2 = \mathbb{Z} \) (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.
Theorem:

If $f_1 : \mathcal{D}_1 \to \mathcal{D}_2$ and $f_2 : \mathcal{D}_2 \to \mathcal{D}_3$ are monotonic, then also $f_2 \circ f_1 : \mathcal{D}_1 \to \mathcal{D}_3$
Theorem:
If $f_1 : D_1 \to D_2$ and $f_2 : D_2 \to D_3$ are monotonic, then also $f_2 \circ f_1 : D_1 \to D_3$.

Theorem:
If $D_2$ is a complete lattice, then the set $[D_1 \to D_2]$ of monotonic functions $f : D_1 \to D_2$ is also a complete lattice where

\[
f \sqsubseteq g \quad \text{iff} \quad f x \sqsubseteq g x \quad \text{for all} \quad x \in D_1\]
Theorem:
If \( f_1 : \mathcal{D}_1 \rightarrow \mathcal{D}_2 \) and \( f_2 : \mathcal{D}_2 \rightarrow \mathcal{D}_3 \) are monotonic, then also \( f_2 \circ f_1 : \mathcal{D}_1 \rightarrow \mathcal{D}_3 \)

Theorem:
If \( \mathcal{D}_2 \) is a complete lattice, then the set \([\mathcal{D}_1 \rightarrow \mathcal{D}_2]\) of monotonic functions \( f : \mathcal{D}_1 \rightarrow \mathcal{D}_2 \) is also a complete lattice where
\[
 f \sqsubseteq g \text{ iff } f x \sqsubseteq g x \text{ for all } x \in \mathcal{D}_1
\]

In particular for \( F \subseteq [\mathcal{D}_1 \rightarrow \mathcal{D}_2] \),
\[
 \bigsqcup F = f \text{ mit } f x = \bigsqcup \{g x \mid g \in F\}
\]
A frequently occurring form of functions \( f_i x = a_i \cap x \cup b_i \).

The operations "\( \circ \)", "\( \sqcup \)" and "\( \sqcap \)" can be explicitly defined by:

\[
\begin{align*}
(f_2 \circ f_1) x &= a_1 \cap a_2 \cap x \cup a_2 \cap b_1 \cup b_2 \\
(f_1 \sqcup f_2) x &= (a_1 \cup a_2) \cap x \cup b_1 \cup b_2 \\
(f_1 \sqcap f_2) x &= (a_1 \cup b_1) \cap (a_2 \cup b_2) \cap x \cup b_1 \cap b_2
\end{align*}
\]
Wanted: least solution for:

\[ x_i \equiv f_i(x_1, \ldots, x_n), \quad i = 1, \ldots, n \quad (\ast) \]

where all \( f_i : \mathbb{D}^n \to \mathbb{D} \) are monotonic.
Wanted: minimally small solution for:

\[ x_i \equiv f_i(x_1, \ldots, x_n), \quad i = 1, \ldots, n \quad (\ast) \]

where all \( f_i : \mathbb{D}^n \to \mathbb{D} \) are monotonic.

Idea:

- Consider \( F : \mathbb{D}^n \to \mathbb{D}^n \) where

\[ F(x_1, \ldots, x_n) = (y_1, \ldots, y_n) \quad \text{with} \quad y_i = f_i(x_1, \ldots, x_n). \]
Wanted: minimally small solution for:

\[ x_i \equiv f_i(x_1, \ldots, x_n), \quad i = 1, \ldots, n \]  

\((*)\)

where all \( f_i : \mathbb{D}^n \to \mathbb{D} \) are monotonic.

Idea:

- Consider \( F : \mathbb{D}^n \to \mathbb{D}^n \) where

\[ F(x_1, \ldots, x_n) = (y_1, \ldots, y_n) \quad \text{with} \quad y_i = f_i(x_1, \ldots, x_n). \]

- If all \( f_i \) are monotonic, then also \( F \)
Wanted: minimally small solution for:

\[ x_i \equiv f_i(x_1, \ldots, x_n), \quad i = 1, \ldots, n \quad (\star) \]

where all \( f_i : \mathbb{D}^n \to \mathbb{D} \) are monotonic.

Idea:

- Consider \( F : \mathbb{D}^n \to \mathbb{D}^n \) where

\[ F(x_1, \ldots, x_n) = (y_1, \ldots, y_n) \text{ with } y_i = f_i(x_1, \ldots, x_n). \]

- If all \( f_i \) are monotonic, then also \( F \)

- We successively approximate a solution. We construct:

\[ \bot, \quad F \bot, \quad F^2 \bot, \quad F^3 \bot, \quad \ldots \]

Hope: We eventually reach a solution ... ????
Example: \[ \mathcal{D} = 2^{\{a, b, c\}}, \quad \subseteq = \subseteq \]

\[
\begin{align*}
x_1 & \supseteq \{a\} \cup x_3 \\
x_2 & \supseteq x_3 \cap \{a, b\} \\
x_3 & \supseteq x_1 \cup \{c\}
\end{align*}
\]
**Example:** \[ D = 2^{\{a,b,c\}}, \quad \subseteq = \subseteq \]

\[ x_1 \supseteq \{a\} \cup x_3 \]
\[ x_2 \supseteq x_3 \cap \{a, b\} \]
\[ x_3 \supseteq x_1 \cup \{c\} \]

**The Iteration:**

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>0</td>
<td>{a}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( x_2 )</td>
<td>0</td>
<td>{a}</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( x_3 )</td>
<td>0</td>
<td>{c}</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example: $\mathbb{D} = 2^{\{a,b,c\}}, \quad \subseteq = \subseteq$

\[
\begin{align*}
x_1 & \supseteq \{a\} \cup x_3 \\
x_2 & \supseteq x_3 \cap \{a, b\} \\
x_3 & \supseteq x_1 \cup \{c\}
\end{align*}
\]

The Iteration:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_1$</td>
<td>$\emptyset$</td>
<td>${a}$</td>
<td>${a, c}$</td>
<td>$\emptyset$</td>
<td>${a, c}$</td>
</tr>
<tr>
<td>$x_2$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
<td>$\emptyset$</td>
</tr>
<tr>
<td>$x_3$</td>
<td>$\emptyset$</td>
<td>${c}$</td>
<td>${a, c}$</td>
<td>${a, c}$</td>
<td>${a, c}$</td>
</tr>
</tbody>
</table>
Example: \( \mathbb{D} = 2^\{a, b, c\}, \quad \subseteq = \subseteq \)

- \( x_1 \supseteq \{a\} \cup x_3 \)
- \( x_2 \supseteq x_3 \cap \{a, b\} \)
- \( x_3 \supseteq x_1 \cup \{c\} \)

The Iteration:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( x_1 )</td>
<td>\emptyset</td>
<td>{a}</td>
<td>{a, c}</td>
<td>{a, c}</td>
<td>{a, c}</td>
</tr>
<tr>
<td>( x_2 )</td>
<td>\emptyset</td>
<td>\emptyset</td>
<td>\emptyset</td>
<td>\emptyset</td>
<td>\emptyset</td>
</tr>
<tr>
<td>( x_3 )</td>
<td>\emptyset</td>
<td>{c}</td>
<td>{a, c}</td>
<td>{a, c}</td>
<td>{a, c}</td>
</tr>
</tbody>
</table>
Example: \[ \mathbb{D} = 2^{\{a,b,c\}}, \quad \subseteq = \subseteq \]

\[
x_1 \supseteq \{a\} \cup x_3
\]
\[
x_2 \supseteq x_3 \cap \{a, b\}
\]
\[
x_3 \supseteq x_1 \cup \{c\}
\]

The Iteration:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_1)</td>
<td>(\emptyset)</td>
<td>({a})</td>
<td>({a, c})</td>
<td>({a, c})</td>
<td></td>
</tr>
<tr>
<td>(x_2)</td>
<td>(\emptyset)</td>
<td>(\emptyset)</td>
<td>(\emptyset)</td>
<td>({a})</td>
<td></td>
</tr>
<tr>
<td>(x_3)</td>
<td>(\emptyset)</td>
<td>({c})</td>
<td>({a, c})</td>
<td>({a, c})</td>
<td></td>
</tr>
</tbody>
</table>
Example: \( \mathbb{D} = 2^{\{a,b,c\}}, \quad \mathbb{C} = \subseteq \)

\[
\begin{align*}
x_1 & \supseteq \{a\} \cup x_3 \\
x_2 & \supseteq x_3 \cap \{a, b\} \\
x_3 & \supseteq x_1 \cup \{c\}
\end{align*}
\]

The Iteration:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>(x_1)</td>
<td>(\emptyset)</td>
<td>({a})</td>
<td>({a, c})</td>
<td>({a, c})</td>
<td>ditto</td>
</tr>
<tr>
<td>(x_2)</td>
<td>(\emptyset)</td>
<td>(\emptyset)</td>
<td>(\emptyset)</td>
<td>({a})</td>
<td></td>
</tr>
<tr>
<td>(x_3)</td>
<td>(\emptyset)</td>
<td>({c})</td>
<td>({a, c})</td>
<td>({a, c})</td>
<td></td>
</tr>
</tbody>
</table>

119
Theorem

- \( \bot, F \bot, F^2 \bot, \ldots \) form an ascending chain:

\[
\bot \subseteq F \bot \subseteq F^2 \bot \subseteq \ldots
\]

- If \( F^k \bot = F^{k+1} \bot \), a solution is obtained which is the least one.
- If all ascending chains are finite, such a \( k \) always exists.
Theorem

- \( \bot, F \bot, F^2 \bot, \ldots \) form an ascending chain:

\[
\bot \subseteq F \bot \subseteq F^2 \bot \subseteq \ldots
\]

- If \( F^k \bot = F^{k+1} \bot \), a solution is obtained which is the least one
- If all ascending chains are finite, such a \( k \) always exists.

Proof

The first claim follows by induction:

**Foundation:** \( F^0 \bot = \bot \subseteq F^1 \bot \)
Step: Assume \( F^{i-1} \bot \subseteq F^i \bot \). Then

\[
F^i \bot = F(F^{i-1} \bot) \subseteq F(F^i \bot) = F^{i+1} \bot
\]

since \( F \) monotonic
Step: Assume $F^{i-1} \perp \subseteq F^i \perp$. Then

$$F^i \perp = F (F^{i-1} \perp) \subseteq F (F^i \perp) = F^{i+1} \perp$$

since $F$ monotonic

Conclusion:

If $\mathbb{D}$ is finite, a solution can be found which is definitely the least

Question:

What, if $\mathbb{D}$ is not finite ???