Conclusion:

Systems of inequations can be solved through fixpoint iteration, i.e., by repeated evaluation of right-hand sides.

Caveat: Naive fixpoint iteration is rather inefficient.

Example:

```
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>(1, x &gt; 1, e - 1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Exp</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
</tr>
<tr>
<td>3</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1, x &gt; 1)</td>
<td>(1, x &gt; 1)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
</tr>
<tr>
<td>5</td>
<td>Exp</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1, x &gt; 1, e - 1)</td>
</tr>
</tbody>
</table>
```

Idea: Round Robin Iteration

Instead of accessing the values of the last iteration, always use the current values of unknowns.

Conclusion:

Systems of inequations can be solved through fixpoint iteration, i.e., by repeated evaluation of right-hand sides.

Caveat: Naive fixpoint iteration is rather inefficient.

Example:

```
<table>
<thead>
<tr>
<th></th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>(1, x &gt; 1, e - 1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Exp</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
</tr>
<tr>
<td>3</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1, x &gt; 1)</td>
<td>(1, x &gt; 1)</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
<td>(1)</td>
</tr>
<tr>
<td>5</td>
<td>Exp</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1, x &gt; 1, e - 1)</td>
<td>(1, x &gt; 1, e - 1)</td>
</tr>
</tbody>
</table>
```
Idea: Round Robin Iteration

Instead of accessing the values of the last iteration, always use the current values of unknowns.

Example:

```
1 0
0 1
1 2
2 3
3 4
4 5
5
```

Example:

```
1 0
0 ∅
1 {1}
2 {1, x > 1}
3 {1, x > 1}
4 {1}
5 {1, x > 1}
```

The code for Round Robin Iteration in Java looks as follows:

```java
for (i = 1; i ≤ n; i++) xi = ⊥;
    do {
        if (finished = true;
            for (i = 1; i ≤ n; i++)
                if (xi ⊨ new)
                    finished = false;
                    x1 = x1 ⊨ new;
            }
        while (!finished);
    }
```
The code for Round Robin Iteration in Java looks as follows:

```java
for (i = 1; i ≤ n; i++) x_i = ⊥;

do {
    finished = true;
    for (i = 1; i ≤ n; i++) {
        new = f(x_1, ..., x_n);
        if (||x_i = new)) {
            finished = false;
            x_i = x_i ⊥ new;
        }
    }
} while (!finished);
```

Correctness:

Assume $y_i^{(d)}$ is the $i$-th component of $F^d_{⊥}$.
Assume $x_i^{(d)}$ is the value of $x_i$ after the $d$-th RR-iteration.

One proves:

1. $y_i^{(d)} ⊑ x_i^{(d)}$ :-)

Correctness:

Assume $y_i^{(d)}$ is the $i$-th component of $F^d_{⊥}$.
Assume $x_i^{(d)}$ is the value of $x_i$ after the $i$-th RR-iteration.

One proves:

1. $y_i^{(d)} ⊑ x_i^{(d)}$ :-)
2. $x_i^{(d)} ⊑ z_i$ for every solution $(z_1, ..., z_n)$ :-)

---

139

140

141

142
Correctness:

Assume \( y_i^{(d)} \) is the \( i \)-th component of \( F^d \downarrow \).
Assume \( x_i^{(d)} \) is the value of \( x_i \) after the \( i \)-th RR-iteration.

One proves:
1. \( y_i^{(0)} \subseteq x_i^{(d)} \implies \)
2. \( x_i^{(d)} \subseteq z_i \) for every solution \( (z_1, \ldots, z_n) \implies \)
3. If RR-iteration terminates after \( d \) rounds, then \( (x_1^{(d)}, \ldots, x_n^{(d)}) \) is a solution \( \implies \)

Caveat:

The efficiency of RR-iteration depends on the ordering of the unknowns !!!

Good:
\[
\rightarrow \quad u \text{ before } v, \quad \text{if } u \rightarrow^* v;
\rightarrow \quad \text{entry condition before loop body } \implies
\]
Caveat:

The efficiency of RR-iteration depends on the ordering of the unknowns !!!

**Good:**

→ u before v, if $u \to^* v$;

→ entry condition before loop body :-)

**Bad:**

e.g., post-order DFS of the CFG, starting at start :-)
Final Question:

Why is a (or the least) solution of the constraint system useful???

For a complete lattice $\mathbb{D}$, consider systems:

\begin{align*}
\mathcal{I}[\text{start}] &\ni d_0 \\
\mathcal{I}[v] &\ni [k]^T(\mathcal{I}[u]) \quad k = (u, \ldots, v) \quad \text{edge}
\end{align*}

where $d_0 \in \mathbb{D}$ and all $[k]^T : \mathbb{D} \to \mathbb{D}$ are monotonic ...

$\implies$ Monotonic Analysis Framework

---

Final Question:

Why is a (or the least) solution of the constraint system useful???

For a complete lattice $\mathbb{D}$, consider systems:

\begin{align*}
\mathcal{I}[\text{start}] &\ni d_0 \\
\mathcal{I}[v] &\ni [k]^T(\mathcal{I}[u]) \quad k = (u, \ldots, v) \quad \text{edge}
\end{align*}

where $d_0 \in \mathbb{D}$ and all $[k]^T : \mathbb{D} \to \mathbb{D}$ are monotonic ...

$\implies$ Monotonic Analysis Framework

---

Wanted: $\ MOP \quad \text{(Merge Over all Paths)}$

\[ \mathcal{I}^*[v] = \bigcup \{ [\pi]^T d_0 \mid \pi : \text{start} \rightarrow^* v \} \]
... end of background on: Complete Lattices

Final Question:

Why is a (or the least) solution of the constraint system useful? ???

For a complete lattice $\mathcal{D}$, consider systems:

$$\mathcal{I}[\text{start}] \sqsubseteq d_0$$

$$\mathcal{I}[v] \sqsubseteq [k]^v(\mathcal{I}[u]) \quad k = (n, \ldots, v) \quad \text{edge}$$

where $d_0 \in \mathcal{D}$ and all $[k]^v : \mathcal{D} \to \mathcal{D}$ are monotonic ...

$\implies$ Monotonic Analysis Framework

Wanted: MOP (Merge Over all Paths)

$$\mathcal{I}^*[v] = \bigsqcup ([\pi]^v d_0 \mid \pi : \text{start} \to^* v)$$

Theorem

Kam, Ullman 1975

Assume $\mathcal{I}$ is a solution of the constraint system. Then:

$$\mathcal{I}[v] \sqsubseteq \mathcal{I}^*[v] \quad \text{for every} \quad v$$

Jeffrey D. Ullman, Stanford
Wanted: MOP  (Merge Over all Paths)

$$\mathcal{I}^*[v] = \bigsqcup \left\{ [\pi]^2 d_0 \mid \pi : \text{start} \rightarrow^* v \right\}$$

**Theorem**  Kam, Ullman 1975

Assume $\mathcal{I}$ is a solution of the constraint system. Then:

$$\mathcal{I}[v] \supseteq \mathcal{I}^*[v] \quad \text{for every } v$$

In particular: $\mathcal{I}[v] \supseteq [\pi]^2 d_0$  for every $\pi : \text{start} \rightarrow^* v$

---

**Proof:** Induction on the length of $\pi$.

**Foundation:** $\pi = \emptyset$  (empty path)

Then:

$$[\pi]^2 d_0 = [\emptyset]^2 d_0 = d_0 \sqsubseteq \mathcal{I}[\text{start}]$$

---

Wanted: MOP  (Merge Over all Paths)

$$\mathcal{I}^*[v] = \bigsqcup \left\{ [\pi]^2 d_0 \mid \pi : \text{start} \rightarrow^* v \right\}$$

**Theorem**  Kam, Ullman 1975

Assume $\mathcal{I}$ is a solution of the constraint system. Then:

$$\mathcal{I}[v] \supseteq \mathcal{I}^*[v] \quad \text{for every } v$$

---

**Proof:** Induction on the length of $\pi$. 

**Foundation:** $\pi = \emptyset$  (empty path)

Then:

$$[\pi]^2 d_0 = [\emptyset]^2 d_0 = d_0 \sqsubseteq \mathcal{I}[\text{start}]$$
Proof: Induction on the length of $\pi$.

Foundation: $\pi = \epsilon$ (empty path)

Then:

$$[\pi]d_0 = [k]d_0 = d_0 \subseteq \mathcal{I}_{\text{start}}$$

Step: $\pi = \pi'k$ for $k = (u, \ldots, v)$ edge.

Then:

$$[\pi']d_0 \subseteq \mathcal{I}[u] \quad \text{by I.H. for } \pi$$

$$\Rightarrow [\pi]'d_0 = [k]^4([\pi]'d_0)$$

$$\subseteq [k]^4(\mathcal{I}[u]) \quad \text{since } [k]^4 \text{ monotonic}$$

$$\subseteq \mathcal{I}[v] \quad \text{since } \mathcal{I} \text{ solution :-)}$$

Disappointment:

Are solutions of the constraint system just upper bounds ???

Answer:

In general: yes :-(
The function \( f : \mathbb{D}_1 \rightarrow \mathbb{D}_2 \) is called
- distributive, if \( f(\bigsqcup X) = \bigsqcup \{fx \mid x \in X\} \) for all \( \emptyset \neq X \subseteq \mathbb{D} \);
- strict, if \( f \perp = \perp \);
- totally distributive, if \( f \) is distributive and strict.

\[ \bigsqcup \emptyset = \perp \]

The function \( f : \mathbb{D}_1 \rightarrow \mathbb{D}_2 \) is called
- distributive, if \( f(\bigsqcup X) = \bigsqcup \{fx \mid x \in X\} \) for all \( \emptyset \neq X \subseteq \mathbb{D} \);
- strict, if \( f \perp = \perp \);
- totally distributive, if \( f \) is distributive and strict.

Examples:
- \( fx = x \cap a \cup b \) for \( a, b \subseteq U \).

\[ \mathcal{D} = \begin{array}{c}
\mathcal{U} \\
\subseteq \\
\end{array} \]

\[ a \cup b = \perp \]

- \( D_1 = D_2 = \mathbb{N} \cup \{\infty\} \)
- \( \text{inc } x = x + 1 \)

The function \( f : \mathbb{D}_1 \rightarrow \mathbb{D}_2 \) is called
- distributive, if \( f(\bigsqcup X) = \bigsqcup \{fx \mid x \in X\} \) for all \( \emptyset \neq X \subseteq \mathbb{D} \);
- strict, if \( f \perp = \perp \);
- totally distributive, if \( f \) is distributive and strict.

Examples:
- \( fx = x \cap a \cup b \) for \( a, b \subseteq U \).

**Strictness:** \( f \emptyset = a \cap \emptyset \cup b = b = \emptyset \) whenever \( b = \emptyset \) \( :-( \)

**Distributivity:**
\[
f(x_1 \cup x_2) = f(x_1 \cap (a \cup b)) \cup (a \cap x_2) = f(x_1 \cup f x_2) = f x_1 \cup f x_2 \quad :-( \)
• \( \mathcal{D}_1 = \mathcal{D}_2 = \mathbb{N} \cup \{\infty\} \), \quad \text{inc} \ x = x + 1

  \text{Strictness:} \quad f \perp = \text{inc} \ 0 = 1 \quad \not= \quad \perp \quad \therefore

  \text{Distributivity:} \quad f \left( \bigcup X \right) = \bigcup \{x + 1 \mid x \in X\} \quad \text{for} \quad \emptyset \neq X

  \therefore

• \( \mathcal{D}_1 = \mathcal{D}_2 = \mathbb{N} \cup \{\infty\} \), \quad \text{inc} \ x = x + 1

  \text{Strictness:} \quad f \perp = \text{inc} \ 0 = 1 \quad \not= \quad \perp \quad \therefore

  \text{Distributivity:} \quad f \left( \bigcup X \right) = \bigcup \{x + 1 \mid x \in X\} \quad \text{for} \quad \emptyset \neq X

  \therefore

• \( \mathcal{D}_1 = (\mathbb{N} \cup \{\infty\})^2 \), \quad \mathcal{D}_2 = \mathbb{N} \cup \{\infty\}, \quad f(x_1, x_2) = x_1 + x_2

  \text{Strictness:} \quad f \perp = 0 + 0 = 0 \quad \therefore

• \( \mathcal{D}_1 = \mathcal{D}_2 = \mathbb{N} \cup \{\infty\} \), \quad \text{inc} \ x = x + 1

  \text{Strictness:} \quad f \perp = \text{inc} \ 0 = 1 \quad \not= \quad \perp \quad \therefore

  \text{Distributivity:} \quad f \left( \bigcup X \right) = \bigcup \{x + 1 \mid x \in X\} \quad \text{for} \quad \emptyset \neq X

  \therefore

• \( \mathcal{D}_1 = \mathcal{D}_2 = \mathbb{N} \cup \{\infty\} \), \quad \text{inc} \ x = x + 1

  \text{Strictness:} \quad f \perp = \text{inc} \ 0 = 1 \quad \not= \quad \perp \quad \therefore

  \text{Distributivity:} \quad f \left( \bigcup X \right) = \bigcup \{x + 1 \mid x \in X\} \quad \text{for} \quad \emptyset \neq X

  \therefore

• \( \mathcal{D}_1 = (\mathbb{N} \cup \{\infty\})^2 \), \quad \mathcal{D}_2 = \mathbb{N} \cup \{\infty\}, \quad f(x_1, x_2) = x_1 + x_2

  \text{Strictness:} \quad f \perp = 0 + 0 = 0 \quad \therefore

  \text{Distributivity:}

  f \left( (1, 4) \cup (4, 1) \right) = f(4, 4) = 8

  \neq 5 = f(1, 4) \cup f(4, 1) \quad \therefore
Remark:

If \( f : D_1 \to D_2 \) is distributive, then also monotonic \( \blacksquare \)

Obviously: \( a \sqsubseteq b \iff a \sqcup b = b \).

From that follows:

\[
\begin{align*}
fb &= f(a \sqcup b) \\
    &= f(a) \sqcup fb \\
\implies fa &\sqsubseteq fb 
\end{align*}
\]
**Assumption:** All $v$ are reachable from $start$.

Then:

**Theorem**

If all effects of edges $[k]^2$ are distributive, then: $I^*[v] = I[v]$ for all $v$.

---


Has developed the operating system CP/M and GUIs for PCs.

---

**Assumption:** All $v$ are reachable from $start$.

Then:

**Theorem**

If all effects of edges $[k]^2$ are distributive, then: $I^*[v] = I[v]$ for all $v$.

**Proof:**

It suffices to prove that $I^*$ is a solution :-(

For this, we show that $I^*$ satisfies all constraints :-(.)
(1) We prove for $\text{start}:

\[ \mathcal{I}[\text{start}] = \bigcup \{ \{x\}^2 d_0 \mid \pi : \text{start} \rightarrow^* \text{start} \} \]
\[ \ni \{x\}^2 d_0 \]
\[ \ni d_0 \quad :) \]

(2) For every $k = (u, \ldots, v)$ we prove:

\[ \mathcal{I}^*[v] = \bigcup \{ \{x\}^2 d_0 \mid \pi : \text{start} \rightarrow^* v \} \]
\[ \ni \{\{x\}^2 d_0 \mid \pi' : \text{start} \rightarrow^* u \} \]
\[ = \bigcup\bigcup \{\{x\}^2 d_0 \mid \pi' : \text{start} \rightarrow^* u \} \]
\[ = \{k\}^2 \bigcup \{\{x\}^2 d_0 \mid \pi' : \text{start} \rightarrow^* u \} \]
\[ = \{k\}^2 \bigcup \{\{x\}^2 d_0 \mid \pi' : \text{start} \rightarrow^* u \} \]
since $\{\pi' \mid \pi' : \text{start} \rightarrow^* u \}$ is non-empty :)
Caveat:

- **Reachability** of all program points cannot be abandoned! Consider:

  $\begin{array}{ccc} 0 \xrightarrow{} 1 \xrightarrow{} 2 \end{array}$

  where $\mathbb{D} = \mathbb{N} \cup \{ \infty \}$

  Then:

  $I[2] = \text{inc} \ 0 = 1$

  $I^+[2] = \bigcup \emptyset = 0$

- **Unreachable** program points can always be thrown away $\therefore$

Summary and Application:

$\rightarrow$ The effects of edges of the analysis of availability of expressions are distributive:

$$ (a \cup (x_1 \cap x_2)) \setminus b = ((a \cup x_1) \cap (a \cup x_2)) \setminus b = ((a \cup x_1) \setminus b) \cap ((a \cup x_2) \setminus b) $$

$$ X \setminus b = X \setminus \overline{b} $$