DATA FLOW ANALYSIS

Claus Brabrand
(( brabrand@itu.dk )))

Associate Professor, Ph.D.
((( Programming, Logic, and Semantics )))

IT University of Copenhagen

[ HOW TO ANALYZE LANGUAGES AUTOMATICALLY ]
### Agenda

- **Relations:**
  - Crossproducts, powersets, and relations

- **Lattices:**
  - Partial-Orders, least-upper-bound, and lattices

- **Monotone Functions:**
  - Monotone Functions and Transfer Functions

- **Fixed Points:**
  - Fixed Points and Solving Recursive Equations

- **Putting it all together…:**
  - Example revisited
Quick recap:

- A control-flow graph
- A lattice
- Transfer functions

We (only) need 3 things:

- $\lambda E . E \[x \mapsto 1 \]$: int $x = 1$
- $\lambda E . E \[y \mapsto 3 \]$: int $y = 3$
- $\lambda E . E \[x \mapsto E(x) \oplus 2 \]$: $x = x + 2$
- $\lambda E . E \[x \mapsto E(y), y \mapsto E(x) \]$: $x \leftrightarrow y$
- print($x, y$)

$[\bot, \bot] \in ENV_L$

$[1, \bot]$

$[1, 3]$

$[1, 3]$

$[3, 3]$

$[3, 1]$

$\top$ ~ “could be anything”

$\bot$ ~ “we haven’t analyzed yet”
Example: Least upper bound

Analyses use ‘⊔’ to **combine information** (at confluence points):
Lattice
A **Lattice** is a **Partial Order**

...where \( \cup S \) exists for all subsets \( S \subseteq L \)

**Examples** (of lattices):

**Non-Examples** (of non-lattices):

*We must be able to combine information*

**Simplification**: if we restrict to **finite height lattices** (which are the ones used in practice); then \( \cup \) only has to exist for each *pair* of elements (i.e., not for all subsets of elements)
Power-Lattices

Powerset Lattices (as Hasse Diagrams):

- \( \mathcal{P}(\{x, y\}) = \{ \emptyset, \{x\}, \{y\}, \{x, y\} \} \)

- ...ordered under '\(\subseteq\)' = '\(\subset\)' (i.e., subset inclusion):

\[
\begin{align*}
\mathcal{P}(\{x, y\}) & \quad \mathcal{P}(\{a, b, c\}) & \quad \mathcal{P}(\{x+1, y\times2\}) \\
\end{align*}
\]
Agenda

- Relations:
  - Crossproducts, powersets, and relations

- Lattices:
  - Partial-Orders, least-upper-bound, and lattices

- Monotone Functions:
  - Monotone Functions and Transfer Functions

- Fixed Points:
  - Fixed Points and Solving Recursive Equations

- Putting it all together…:
  - Example revisited
**Constant** Transfer functions:

- Analysing value-of-’b’
  
  - $b = \text{true}$
  
  - $f() = \text{true}$

- Analysing sign-of-’x’
  
  - $x = 7$
  
  - $f() = +$
Unary Transfer functions:

- Analysing value-of-’b’
  - Transfer function: $b = \neg b$

- Analysing sign-of-’x’
  - Transfer function: $x = -x$

Exercise: what’s this transfer function?
Transfer Functions

**Binary** Transfer functions:

- Analysing value-of-'b'
  - Transfer function

- Analysing sign-of-'x'
  - Transfer function

**Exercise:**

- What's this transfer function?
Environments

Runtime: \( \text{Var} \rightarrow \text{Val} \)

Analysis: \( \text{Var} \rightarrow \mathcal{L} \)  (i.e., abstract values)
We need Transfer Functions on Environments!

- Say lattice \( L \) analyses "constantness" of one single value:

- We need environments for analyzing:
  - (tracking both \( v \) and \( w \)):

- i.e. we need lattice \( L \times L' \):
  - Note:

\[
L \times L \approx \{x,y\} \rightarrow L \quad \text{and} \quad L_{\text{VAR}} \approx \text{VAR} \rightarrow L
\]
Crossproduct Lattice

*Environment Lattice:*

- \( L \times L \):

\[
\begin{array}{c}
T \\
is\text{constant}
\end{array}
\times
\begin{array}{c}
T \\
is\text{constant}
\end{array}
= \begin{array}{c}
T \\
is\text{constant}
\end{array}
\]

**NB:** if \( L \) is a lattice, then \( L \times L \) is always a lattice
Exercise

- Calculate crossproduct lattice:
Transfer Functions on Environment Lattices

- **Constant** Transfer functions:

  1. For 'x' and 'y':
     - \( \lambda E \cdot E[\text{x = true}] \) (ENV transfer function)
   - (false, true)
   - (true, true)

  2. For 'n' and 'm':
     - \( \lambda E \cdot E[\text{n = 7}] \) (ENV transfer function)
   - (7, 0)
   - (0, 0)

Claus Brabrand, UFPE, Brazil
Transfer Functions on Environment Lattices

**Unary Transfer functions:**

For 'x'

For 'y'

For 'n'

For 'm'

**EXERCISE:**

(what’s the transfer func?)

\[
\lambda E . E[x = f_{\text{not}}(E(y))] \quad \text{ENV transfer function}
\]

\[
\lambda E . E[n = f_{-}(E(m))] \quad \text{ENV transfer function}
\]
Transfer Functions on Environment Lattices

**Binary** Transfer functions:

- Binary operations on boolean values:
  - For 'x' and 'y':
    - $x = x \land y$
  - For 'n' and 'm':
    - $n = n + m$

**ENV transfer function**

- For 'x' and 'y':
  - $\lambda E \cdot E[x = E(x) \land E(y)]$

- For 'n' and 'm':
  - $\lambda E \cdot E[x = E(n) \lor E(m)]$

Claus Brabrand, UFPE, Brazil

DATA-FLOW ANALYSIS

Monotonicity

Monotone Transfer Functions
Monotone Functions

- **Monotone** function \( f : L \rightarrow L \) (on a lattice \( L \)):
  \[
  \forall x, y \in L : x \subseteq y \implies f(x) \subseteq f(y)
  \]

- **Note:**
  - this is **not** saying that 'f' is ascending:
    \[
    \forall x \in L : x \subseteq f(x)
    \]

All the transfer functions you have seen were monotone! :-)

- **Examples:**
  \[
  f() = \text{true}, \quad \text{...on lattice:}
  \]

Claus Brabrand, UFPE, Brazil

DATA-FLOW ANALYSIS

More Examples:

- \( \lambda E . E[x = \text{true}] \)
- \( \lambda E . E[x = \text{not}(E(y))] \)
- \( \lambda E . E[x = E(x) \land E(y)] \)

Exercise:

Check monotonicity of 'f' below:

1) \( f(X) = \{a, b\} \)
2) \( f(X) = X \cup \{a\} \)
3) \( f(X) = \{a, b\} \setminus X \)
4) \( f(X) = X^c \)

Monotonicity:
\[ \forall x, y \in L : x \subseteq y \Rightarrow f(x) \subseteq f(y) \]

Monotone Func's (cont'd)

...on lattice:

\[ \lambda E . E[x = E(x) \land E(y)] \]

...on lattice:

\[ \lambda E . E[x = \text{not}(E(y))] \]

...on lattice:

\[ \lambda E . E[x = \text{true}] \]
Agenda

- Relations:
  - Crossproducts, powersets, and relations

- Lattices:
  - Partial-Orders, least-upper-bound, and lattices

- Monotone Functions:
  - Monotone Functions and Transfer Functions

- Fixed Points:
  - Fixed Points and Solving Recursive Equations

- Putting it all together…:
  - Example revisited
Fixed-Points

A fixed-point for a function $f : L \rightarrow L$

...is an element $\ell \in L$

such that: $\ell = f(\ell)$

Example:

Function: $f(X) = X \cup \{c\}$

...over $P\{a,b,c\}$

'f' has many fixed-points:

- $\{c\}$ \hspace{1cm} LEAST fixed point
- $\{a,c\}$
- ...
- $\{a,b,c\}$ In fact, anything that includes 'c'
Another Example:

Recursive equations:

\[
\begin{align*}
  a &= \bot \\
  b &= f_{x=1}(a) \\
  c &= b \cup d \\
  d &= f_{x=x+1}(c)
\end{align*}
\]
Fixed-Point Theorem!

If:

- ’f’ is a monotone function: $f : L \to L$
- …over a lattice $L$ (with finite height):

Then:

- ’f’ is guaranteed to have a unique least fixed-point
- …which is computable as:

$$\text{fix}(f) = \bigsqcup_{i \geq 0} f^i(\bot)$$

Intuition: $\bot \subseteq f(\bot) \subseteq f(f(\bot)) \subseteq \ldots$ until equality

Proof is quite simple [cf. Notes, p.13 top]
Now you can...

- Solve *ANY* recursive equations
  (involving monotone functions over lattices):

\[
\begin{align*}
  x &= f(x, y, z) \\
  y &= g(x, y, z) \\
  z &= h(x, y, z)
\end{align*}
\]

- Which means that you can solve *any* recursive data-flow analysis equation! :-}
## Agenda

- **Relations:**
  - Crossproducts and relations

- **Lattices:**
  - Partial-Orders, least-upper-bound, and lattices

- **Monotone Functions:**
  - Monotone Functions and Transfer Functions

- **Fixed Points:**
  - Fixed Points and Solving Recursive Equations

- **Putting it all together…:**
  - Example revisited
All you need is…:

We (only) need 3 things:
- A control-flow graph
- A lattice
- Transfer functions

Given program:
```plaintext
int x = 1;
int y = 3;
if (...) {
    x = x+2;
} else {
    x <-> y;
}
print(x,y);
```

```
λE . E[x₁→ 1]
```

```
λE . E[y₁→ 3]
```

```
λE . E[x₁→ E(x) ⊕ 2]
```

```
x = x+2;
```

```
x <-> y;
```

```
λE . E[x₁→ E(y), y₁→ E(x)]
```

```
print(x,y);
```
Solve Equations :-) 

- One **big** lattice:
  - E.g., \((L^{\text{VAR}})^{\text{PP}}\)

- 1 **big** abstract value vector:
  - \([[[\bot, \bot], [\bot, \bot], \ldots, [\bot, \bot]]] \in (L^{\text{VAR}})^{\text{PP}}\)

- 1 **big** transfer function:
  - \(F : (L^{\text{VAR}})^{\text{PP}} \rightarrow (L^{\text{VAR}})^{\text{PP}}\)

- Compute fixed-point (simply):
  - Start with bottom value vector (\(\bot(L^{\text{VAR}})^{\text{PP}}\))
  - Iterate transfer function ‘\(F\)’ (until nothing changes)
  - Done; print out (or use) solution…! :-)}
- Agenda

- Relations:
  - Crossproducts, powersets, and relations

- Lattices:
  - Partial-Orders, least-upper-bound, and lattices

- Monotone Functions:
  - Monotone Functions and Transfer Functions

- Fixed Points:
  - Fixed Points and Solving Recursive Equations

- Putting it all together…:
  - Example revisited
The Entire Process :-) 

Program:

```plaintext
x = 0;
do {
    x = x+1;
} while (...);
output x;
```

1. Control-flow graph:

```
x = 0;
a = ...
b = ...
c = ...
d = ...
e = ...
output x;
```

2. Transfer functions:

```plaintext
f_{x=0}(l) = \emptyset
f_{x=x+1}(l) = l \oplus_{L} +
```

3. Recursive equations:

```plaintext
a = ⊥
b = f_{x=0}(a)
c = b \cup d
d = f_{x=x+1}(c)
e = d
output x;
```

4. one "big" transfer function:

```plaintext
T((a, b, c, d, e)) = (⊥, f_{x=0}(a), b \cup d, f_{x=x+1}(c), d)
```

5. Solve rec. equations...

```
T^0 ⊥ T^1 ⊥ T^2 ⊥ T^3 ⊥ T^4 ⊥ T^5
```

...over a "big" power-lattice:

```
|VAR|*|PP| = 1*5 = 5
```

solution

ANOTHER FIXED POINT

LEAST FIXED POINT
Exercise:

Repeat this process for program (of two vars):

```
x = 1;
y = 0;
while (v>w) {
  x <-> y;
}
y = y+1;
```

i.e., determine…:

1) Control-flow graph
2) Transfer functions
3) Recursive equations
4) One "big" transfer function
5) Solve recursive equations :-)

...using lattice:
Now, please: 3’ recap

- Please spend 3' on thinking about and writing down the main ideas and points from the lecture – now!

![Diagram: The Value of Rehearsal Following a Lecture](image)

**Figure 2.5. The Value of Rehearsal Following a Lecture.**

- Immediately
- After 1 day
- After 1 week
- After 2 weeks
- After 3 weeks

*Source: Adapted from Bassey (1968).*