

Special Topics in Logic and Security 1

Dataflow Analysis

Paul Irofti and Ioana Leuştean

Master Year II, Sem. I, 2023-2024

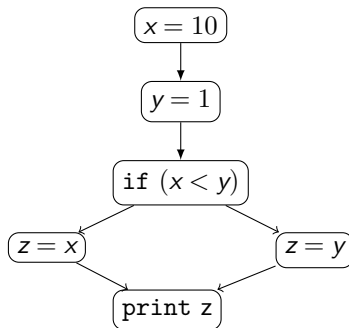
What is Static Analysis?

Static analysis involves the analysis of program properties without executing them.

Properties examples: type analysis, null pointers, unused assignments, code vulnerabilities (e.g. buffer overflow), etc.

Control Flow Graphs (CFG)

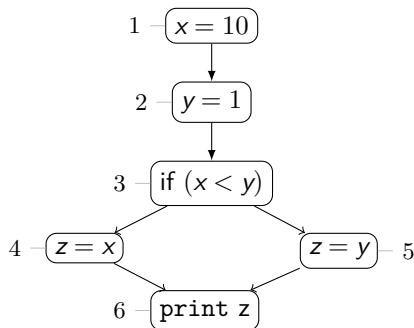
```
x = 10;  
y = 1;  
if (x < y) z = x;  
    else  z = y;  
print z
```



Control Flow Graphs (CFG)

```
x = 10;  
y = 1;  
if (x < y) z = x;  
    else  z=y;  
print z
```

We label each of the graph's nodes.



Example: determining live variables

https://cs420.epfl.ch/c/06_dataflow-analysis.html

A variable is **active** (**live**) at a given point in the program's execution if it is possible for the current variable value to be read before it is overwritten.

For a node n from the CFG we will denote with v_n the set of variables that are **live** before the execution of n .

With that

$$v_n = \left(\bigcup \{v_i \mid i \text{ successor of } n\} \setminus \text{Written}(n) \right) \cup \text{Read}(n)$$

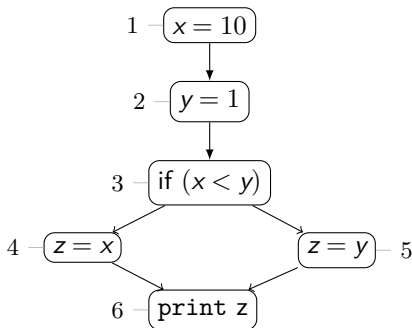
where

- $\text{Written}(n)$ is the set of variables that are redefined in n ,
- $\text{Read}(n)$ is the set of variables that are read in n .

Example: determining live variables

Taking into account the above identified condition, in each node we obtain a set of constraints:

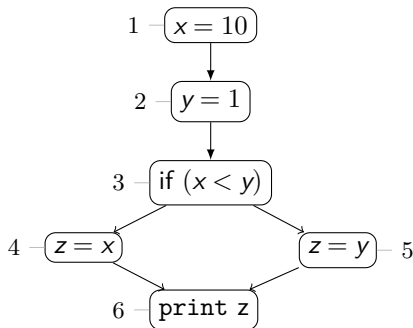
$$\begin{aligned}v_1 &= v_2 \setminus \{x\} \\v_2 &= v_3 \setminus \{y\} \\v_3 &= v_4 \cup v_5 \cup \{x, y\} \\v_4 &= (v_6 \setminus \{z\}) \cup \{x\} \\v_5 &= (v_6 \setminus \{z\}) \cup \{y\} \\v_6 &= \{z\}\end{aligned}$$



Example: determining live variables

Solving the system we obtain the solution:

$v_1 = \emptyset$
 $v_2 = \{x\}$
 $v_3 = \{x, y\}$
 $v_4 = \{x\}$
 $v_5 = \{y\}$
 $v_6 = \{z\}$



- In the problem analysis from above, we associated a set of variables to each of the node from the control flow graph that represent the solution of a system of equations from $\mathcal{P}(Var)$, where Var is the set of variables from the entire program.
- In general, the analysis of problems from this class leads to a set of equations of lattice L . Is there a generic method for solving such a system of equations?

POSET

Definition (partial order)

Let A be a set. A binary relation $R \subseteq A \times A$ is a **partial order** if:

- reflexive: $(x, x) \in R$ for any $x \in A$
- antisymmetric: $(x, y) \in R$ and $(y, x) \in R \implies x = y, \forall x, y \in A$
- transitive: $(x, y) \in R$ and $(y, z) \in R \implies (x, z) \in R, \forall x, y, z \in A$

Definition (POSET)

A partially ordered set (**poset**) is a pair (A, R) , where A is a set and R is a partial order on A .

Key poset elements

Definition (infimum and supremum)

Let A be a poset and $X \subseteq A$. The element $a \in A$ is called:

- lower bound for X if $a \leq x \forall x \in X$
- upper bound for X if $x \leq a \quad \forall x \in X$
- **infimum** for X if a is the greatest lower bound; we denote $a = \inf X$
- **supremum** for X if a is the lowest upper bound; we denote $a = \sup X$

Definition (top and bottom)

Let (A, \leq) be a poset. An element $e \in A$ is called:

- minimal element if $a \leq e \implies a = e$ (more than one)
- the least (**bottom**, minimum) element if $e \leq a \quad \forall a \in A$ (unique)
- maximal element if $e \leq a \implies a = e$ (more than one)
- the greatest (**top**, maximum) element if $a \leq e \quad \forall a \in A$ (unique)

Complete Partial Order (CPO)

Definition (chain)

A partial order is called total if any two elements are comparable. A totally ordered set (or a **chain**) is a pair (A, \leq) where A is a set and \leq is a total order on A .

Definition (CPO)

A complete partial order (CPO) is a poset (C, \leq, \perp) such that C has a bottom element \perp and $\sup X$ exists for any chain $X \subseteq C$.

Lattice

Definition

A poset (L, \leq) is a **lattice** if $\exists \sup\{x_1, x_2\}, \inf\{x_1, x_2\} \quad \forall x_1, x_2 \in L$.

The infimum and the supremum become operations on L :

$$\vee : L \times L \rightarrow L, \quad x_1 \vee x_2 := \sup\{x_1, x_2\}$$

$$\wedge : L \times L \rightarrow L, \quad x_1 \wedge x_2 := \inf\{x_1, x_2\}$$

Proposition

The following identities hold:

- *associativity:* $(x \vee y) \vee z = x \vee (y \vee z), \quad (x \wedge y) \wedge z = x \wedge (y \wedge z),$
- *commutativity:* $x \vee y = y \vee x, \quad x \wedge y = y \wedge x,$
- *absorption:* $x \vee (x \wedge y) = x, \quad x \wedge (x \vee y) = x.$

Definition (complete)

A lattice (L, \leq, \vee, \wedge) is complete if $\inf X$ and $\sup X$ exist for any $X \subseteq L$.

Monotony

Definition (monotone)

Given two posets (A, \leq_A) and (B, \leq_B) , we say that a function $f: A \rightarrow B$ is monotone (increasing) if $a_1 \leq_A a_2 \implies f(a_1) \leq_B f(a_2), \forall a_1, a_2 \in A$.

Definition (continuous)

Given two CPOs (A, \leq_A) and (B, \leq_B) we say that a function $f: A \rightarrow B$ is continuous if $f(\sup\{a_n \mid n \in \mathbb{N}\}) = \sup\{f(a_n) \mid n \in \mathbb{N}\}$ for any chain $\{a_n \mid n \in \mathbb{N}\}$ from A .

Remark

Continuity implies monotony.

Fixed Point

Definition (fixed point)

An element $a \in A$ is a fixed point of $f: A \rightarrow A$ if $f(a) = a$.

Theorem (Fixed point theorem for complete lattices (Knaster-Tarski))

If (L, \leq) is a complete lattice and $F: L \rightarrow L$ is a monotone (increasing) function, then $a = \inf\{x \in L \mid F(x) \leq x\}$ is the least fixed point of F .

Theorem (Fixed point theorem for CPOs (Kleene))

If (C, \leq, \perp) is a CPO and $F: C \rightarrow C$ is a continuous function, then $a = \sup\{F^n(\perp) \mid n \in \mathbb{N}\}$ is the least fixed point of F .

Note that a exists since the sequence $F^0(\perp) = \perp \leq F(\perp) \leq F^2(\perp) \leq \dots \leq F^n(\perp)$ is a chain.

Kleene Proof

Theorem (Fixed point theorem for CPOs (Kleene))

If (C, \leq, \perp) is a CPO and $F: C \rightarrow C$ is a continuous function, then $a = \sup\{F^n(\perp) \mid n \in \mathbb{N}\}$ is the least fixed point of F .

Kleene Proof

Theorem (Fixed point theorem for CPOs (Kleene))

If (C, \leq, \perp) is a CPO and $F: C \rightarrow C$ is a continuous function, then $a = \sup\{F^n(\perp) | n \in \mathbb{N}\}$ is the least fixed point of F .

i) a is a fixed point

Kleene Proof

Theorem (Fixed point theorem for CPOs (Kleene))

If (C, \leq, \perp) is a CPO and $F: C \rightarrow C$ is a continuous function, then $a = \sup\{F^n(\perp) \mid n \in \mathbb{N}\}$ is the least fixed point of F .

i) a is a fixed point

$$\begin{aligned} F(a) &= F(\sup\{F^n(\perp) \mid n \in \mathbb{N}\}) \\ &= \sup\{F(F^n(\perp)) \mid n \in \mathbb{N}\} && \text{(continuity)} \\ &= \sup\{F^{n+1}(\perp) \mid n \in \mathbb{N}\} \\ &= \sup\{F^n(\perp) \mid n \in \mathbb{N}\} = a \end{aligned}$$

Kleene Proof

Theorem (Fixed point theorem for CPOs (Kleene))

If (C, \leq, \perp) is a CPO and $F : C \rightarrow C$ is a continuous function, then $a = \sup\{F^n(\perp) \mid n \in \mathbb{N}\}$ is the least fixed point of F .

i) a is a fixed point

$$\begin{aligned} F(a) &= F(\sup\{F^n(\perp) \mid n \in \mathbb{N}\}) \\ &= \sup\{F(F^n(\perp)) \mid n \in \mathbb{N}\} && \text{(continuity)} \\ &= \sup\{F^{n+1}(\perp) \mid n \in \mathbb{N}\} \\ &= \sup\{F^n(\perp) \mid n \in \mathbb{N}\} = a \end{aligned}$$

ii) a is the least fixed point

Kleene Proof

Theorem (Fixed point theorem for CPOs (Kleene))

If (C, \leq, \perp) is a CPO and $F : C \rightarrow C$ is a continuous function, then $a = \sup\{F^n(\perp) \mid n \in \mathbb{N}\}$ is the least fixed point of F .

i) a is a fixed point

$$\begin{aligned} F(a) &= F(\sup\{F^n(\perp) \mid n \in \mathbb{N}\}) \\ &= \sup\{F(F^n(\perp)) \mid n \in \mathbb{N}\} && \text{(continuity)} \\ &= \sup\{F^{n+1}(\perp) \mid n \in \mathbb{N}\} \\ &= \sup\{F^n(\perp) \mid n \in \mathbb{N}\} = a \end{aligned}$$

ii) a is the least fixed point

① Assume b is another fixed point, i.e. $F(b) = b$.

Kleene Proof

Theorem (Fixed point theorem for CPOs (Kleene))

If (C, \leq, \perp) is a CPO and $F : C \rightarrow C$ is a continuous function, then $a = \sup\{F^n(\perp) \mid n \in \mathbb{N}\}$ is the least fixed point of F .

i) a is a fixed point

$$\begin{aligned} F(a) &= F(\sup\{F^n(\perp) \mid n \in \mathbb{N}\}) \\ &= \sup\{F(F^n(\perp)) \mid n \in \mathbb{N}\} && \text{(continuity)} \\ &= \sup\{F^{n+1}(\perp) \mid n \in \mathbb{N}\} \\ &= \sup\{F^n(\perp) \mid n \in \mathbb{N}\} = a \end{aligned}$$

ii) a is the least fixed point

- 1 Assume b is another fixed point, i.e. $F(b) = b$.
- 2 We prove by induction on $n \geq 1$ that $F^n(\perp) \leq b$.

Kleene Proof

Theorem (Fixed point theorem for CPOs (Kleene))

If (C, \leq, \perp) is a CPO and $F: C \rightarrow C$ is a continuous function, then $a = \sup\{F^n(\perp) \mid n \in \mathbb{N}\}$ is the least fixed point of F .

i) a is a fixed point

$$\begin{aligned} F(a) &= F(\sup\{F^n(\perp) \mid n \in \mathbb{N}\}) \\ &= \sup\{F(F^n(\perp)) \mid n \in \mathbb{N}\} && \text{(continuity)} \\ &= \sup\{F^{n+1}(\perp) \mid n \in \mathbb{N}\} \\ &= \sup\{F^n(\perp) \mid n \in \mathbb{N}\} = a \end{aligned}$$

ii) a is the least fixed point

- 1 Assume b is another fixed point, i.e. $F(b) = b$.
- 2 We prove by induction on $n \geq 1$ that $F^n(\perp) \leq b$.
- 3 For $n = 0$, $F^0(\perp) = \perp \leq b$ because \perp is the first element.

Kleene Proof

Theorem (Fixed point theorem for CPOs (Kleene))

If (C, \leq, \perp) is a CPO and $F : C \rightarrow C$ is a continuous function, then $a = \sup\{F^n(\perp) \mid n \in \mathbb{N}\}$ is the least fixed point of F .

i) a is a fixed point

$$\begin{aligned} F(a) &= F(\sup\{F^n(\perp) \mid n \in \mathbb{N}\}) \\ &= \sup\{F(F^n(\perp)) \mid n \in \mathbb{N}\} && \text{(continuity)} \\ &= \sup\{F^{n+1}(\perp) \mid n \in \mathbb{N}\} \\ &= \sup\{F^n(\perp) \mid n \in \mathbb{N}\} = a \end{aligned}$$

ii) a is the least fixed point

- ① Assume b is another fixed point, i.e. $F(b) = b$.
- ② We prove by induction on $n \geq 1$ that $F^n(\perp) \leq b$.
- ③ For $n = 0$, $F^0(\perp) = \perp \leq b$ because \perp is the first element.
- ④ If $F^n(\perp) \leq b$, then $F^{n+1}(\perp) \leq F(b)$ because F is increasing. Since $F(b) = b$ we get $F^{n+1}(\perp) \leq b$.

Knaster-Tarski Proof

Theorem

If (L, \leq) is a complete lattice and $F : L \rightarrow L$ is a monotone (increasing) function, then $a = \inf\{x \in L \mid F(x) \leq x\}$ is the least fixed point of F .

Knaster-Tarski Proof

Theorem

If (L, \leq) is a complete lattice and $F : L \rightarrow L$ is a monotone (increasing) function, then $a = \inf\{x \in L \mid F(x) \leq x\}$ is the least fixed point of F .

i) a is a fixed point

Knaster-Tarski Proof

Theorem

If (L, \leq) is a complete lattice and $F : L \rightarrow L$ is a monotone (increasing) function, then $a = \inf\{x \in L \mid F(x) \leq x\}$ is the least fixed point of F .

i) a is a fixed point

① Let $X := \{x \in L \mid F(x) \leq x\} \implies a \leq x, \quad \forall x \in X \implies F(a) \leq F(x) \leq x$

Knaster-Tarski Proof

Theorem

If (L, \leq) is a complete lattice and $F : L \rightarrow L$ is a monotone (increasing) function, then $a = \inf\{x \in L \mid F(x) \leq x\}$ is the least fixed point of F .

i) a is a fixed point

- 1 Let $X := \{x \in L \mid F(x) \leq x\} \implies a \leq x, \forall x \in X \implies F(a) \leq F(x) \leq x$
- 2 Consequently $F(a)$ is a lower bound for $X \implies F(a) \leq a$

Knaster-Tarski Proof

Theorem

If (L, \leq) is a complete lattice and $F : L \rightarrow L$ is a monotone (increasing) function, then $a = \inf\{x \in L \mid F(x) \leq x\}$ is the least fixed point of F .

i) a is a fixed point

- 1 Let $X := \{x \in L \mid F(x) \leq x\} \implies a \leq x, \forall x \in X \implies F(a) \leq F(x) \leq x$
- 2 Consequently $F(a)$ is a lower bound for $X \implies F(a) \leq a$
- 3 It follows that $F(a) \in X \implies a \leq F(a)$

Knaster-Tarski Proof

Theorem

If (L, \leq) is a complete lattice and $F : L \rightarrow L$ is a monotone (increasing) function, then $a = \inf\{x \in L \mid F(x) \leq x\}$ is the least fixed point of F .

i) a is a fixed point

- ① Let $X := \{x \in L \mid F(x) \leq x\} \implies a \leq x, \forall x \in X \implies F(a) \leq F(x) \leq x$
- ② Consequently $F(a)$ is a lower bound for $X \implies F(a) \leq a$
- ③ It follows that $F(a) \in X \implies a \leq F(a)$
- ④ We proved that $F(a) \leq a$ and $a \leq F(a)$, so $F(a) = a$.

Knaster-Tarski Proof

Theorem

If (L, \leq) is a complete lattice and $F : L \rightarrow L$ is a monotone (increasing) function, then $a = \inf\{x \in L \mid F(x) \leq x\}$ is the least fixed point of F .

i) a is a fixed point

- 1 Let $X := \{x \in L \mid F(x) \leq x\} \implies a \leq x, \forall x \in X \implies F(a) \leq F(x) \leq x$
- 2 Consequently $F(a)$ is a lower bound for $X \implies F(a) \leq a$
- 3 It follows that $F(a) \in X \implies a \leq F(a)$
- 4 We proved that $F(a) \leq a$ and $a \leq F(a)$, so $F(a) = a$.

ii) a is the least fixed point

Knaster-Tarski Proof

Theorem

If (L, \leq) is a complete lattice and $F : L \rightarrow L$ is a monotone (increasing) function, then $a = \inf\{x \in L \mid F(x) \leq x\}$ is the least fixed point of F .

i) a is a fixed point

- 1 Let $X := \{x \in L \mid F(x) \leq x\} \implies a \leq x, \forall x \in X \implies F(a) \leq F(x) \leq x$
- 2 Consequently $F(a)$ is a lower bound for $X \implies F(a) \leq a$
- 3 It follows that $F(a) \in X \implies a \leq F(a)$
- 4 We proved that $F(a) \leq a$ and $a \leq F(a)$, so $F(a) = a$.

ii) a is the least fixed point

- 1 assume b is another fixed point, i.e. $F(b) = b$

Knaster-Tarski Proof

Theorem

If (L, \leq) is a complete lattice and $F : L \rightarrow L$ is a monotone (increasing) function, then $a = \inf\{x \in L \mid F(x) \leq x\}$ is the least fixed point of F .

i) a is a fixed point

- 1 Let $X := \{x \in L \mid F(x) \leq x\} \implies a \leq x, \forall x \in X \implies F(a) \leq F(x) \leq x$
- 2 Consequently $F(a)$ is a lower bound for $X \implies F(a) \leq a$
- 3 It follows that $F(a) \in X \implies a \leq F(a)$
- 4 We proved that $F(a) \leq a$ and $a \leq F(a)$, so $F(a) = a$.

ii) a is the least fixed point

- 1 assume b is another fixed point, i.e. $F(b) = b$
- 2 then $b \in X$, so $a \leq b$.

The Product Lattice

Definition

We define the pointwise order relationship:

$$(x_1, \dots, x_n) \leq (y_1, \dots, y_n) \iff x_i \leq y_i, \forall i \in \{1, \dots, n\}$$

Theorem

If L_1, L_2, \dots, L_n are lattices then so is the product

$$L_1 \times L_2 \times \dots \times L_n = \{(x_1, x_2, \dots, x_n) \mid x_i \in L_i\}$$

where the lattice order \leq is defined pointwise.

Remark

Let L be a lattice and $n \geq 1$, note that (L^n, \leq) is a lattice. More so, if L is a complete lattice then L^n is also a complete lattice.

Equation Systems in Complete Lattices

Let L be a complete lattice and $n \geq 1$ and $F_1, \dots, F_n : L^n \rightarrow L$ a set of monotone functions. We want to solve the equation system:

$$\begin{aligned}x_1 &= F_1(x_1, \dots, x_n) \\x_2 &= F_2(x_1, \dots, x_n) \\&\dots \\x_n &= F_n(x_1, \dots, x_n)\end{aligned}$$

with $x_1, \dots, x_n \in L$.

Definition

Denote the function $F : L^n \rightarrow L^n$ to be

$$F(x_1, \dots, x_n) = (F_1(x_1, \dots, x_n), F_2(x_1, \dots, x_n), \dots, F_n(x_1, \dots, x_n))$$

and observe that the system can be rewritten as

$$F(x_1, \dots, x_n) = (x_1, \dots, x_n)$$

The equation system can be solved using the [Fixed Point Theorem](#)!

Determining live variables

It results that, in order to solve the system:

$$\begin{aligned}v_1 &= v_2 \setminus \{x\} \\v_2 &= v_3 \setminus \{y\} \\v_3 &= v_4 \cup v_5 \cup \{x, y\} \\v_4 &= (v_6 \setminus \{z\}) \cup \{x\} \\v_5 &= (v_6 \setminus \{z\}) \cup \{y\} \\v_6 &= \{z\}\end{aligned}$$

we need to find the solution to the equation (where v_i is denoted as x_i):

$$\begin{aligned}F(x_1, x_2, x_3, x_4, x_5, x_6) = \\(x_2 \setminus \{x\}, x_3 \setminus \{y\}, x_4 \cup x_5 \cup \{x, y\}, (x_6 \setminus \{z\}) \cup \{x\}, (x_6 \setminus \{z\}) \cup \{y\}, \{z\})\end{aligned}$$

in the complete lattice $(\mathcal{P}(Var), \subseteq, \emptyset, Var)$.

Fixed Point Theorem Solution

$$F(x_1, x_2, x_3, x_4, x_5, x_6) = \\ (x_2 \setminus \{x\}, x_3 \setminus \{y\}, x_3 \setminus \{y\}, (x_6 \setminus \{z\}) \cup \{x\}, (x_6 \setminus \{z\}) \cup \{y\}, \{z\})$$

Determining the least fixed-point:

	x_1	x_2	x_3	x_4	x_5	x_6
\perp	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
$F(\perp)$	\emptyset	\emptyset	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$
$F^2(\perp)$	\emptyset	$\{x\}$	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$
$F^3(\perp)$	\emptyset	$\{x\}$	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$

where $\perp = (\emptyset, \dots, \emptyset)$

IMP and live variables

Let the set of the variables successor to the node n be:

$$Join(v_n) = \bigcup \{v_i \mid i \text{ successor of } n\}$$

Then, the general formulation for live variables in node k is:

$$v_n = Join(v_n) \setminus Written(n) \cup Read(n)$$

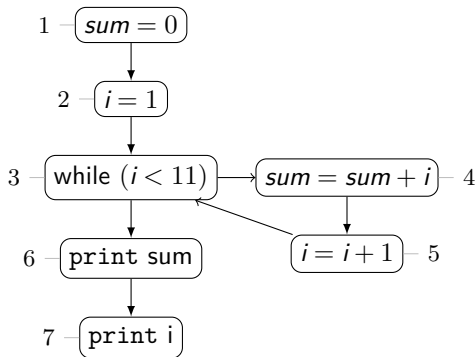
Particular formulations for IMP instructions:

- | | | |
|----------------------|-----------|---|
| • Expressions (E) | E | $x + 3, (x > 7)$ |
| • Instruction blocks | | |
| • Assignment | $x = E;$ | $v_n = Join(v_n) \setminus \{x\} \cup Var(E)$ |
| • Conditional | if (E) | $v_n = Join(v_n) \cup Var(E)$ |
| • Loops | while (E) | $v_n = Join(v_n) \cup Var(E)$ |
| • Output | print E; | $v_n = Join(v_n) \cup Var(E)$ |

where $Var(E)$ represents the set of variables present in expression E .

Control Flow Graphs (CFG)

```
sum = 0;  
i = 1;  
while (i < 11) {  
    sum = sum + i;  
    i = i + 1;  
}  
print sum  
print i
```



Control Flow Graphs (CFG)

$$v_1 = v_2 \setminus \{sum\}$$

$$v_2 = v_3 \setminus \{i\}$$

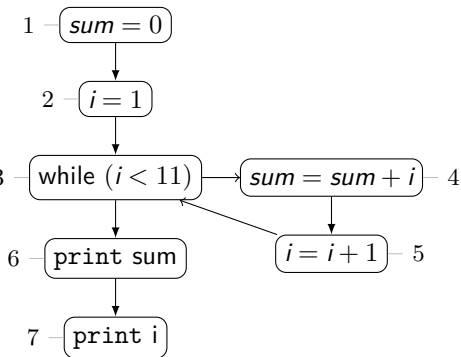
$$v_3 = v_4 \cup v_6 \cup \{i\}$$

$$v_4 = (v_5 \setminus \{sum\}) \cup \{i, sum\}$$

$$v_5 = (v_3 \setminus \{i\}) \cup \{i\}$$

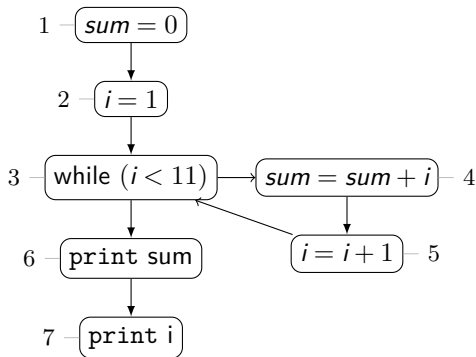
$$v_6 = v_7 \cup \{sum\}$$

$$v_7 = \{i\}$$



Control Flow Graphs (CFG)

$v_1 = \emptyset$
 $v_2 = \{sum\}$
 $v_3 = \{i, sum\}$
 $v_4 = \{i, sum\}$
 $v_5 = \{i, sum\}$
 $v_6 = \{i, sum\}$
 $v_7 = \{i\}$



Solving with the Fixed Point Theorem

$$F(x_1, x_2, x_3, x_4, x_5, x_6, x_7) = (x_2 \setminus \{\text{sum}\}, x_3 \setminus \{i\}, x_4 \cup x_6 \cup \{i\}, (x_5 \setminus \{\text{sum}\}) \cup \{i, \text{sum}\}, x_3 \setminus \{i\} \cup \{i\}, x_7 \cup \{\text{sum}\}, \{i\})$$

Determining the least fixed-point:

	x_1	x_2	x_3	x_4	x_5	x_6	x_7
\perp	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
$F(\perp)$	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	$\{\text{sum}\}$	$\{i\}$
$F^2(\perp)$	\emptyset	\emptyset	$\{\text{sum}, i\}$	$\{\text{sum}, i\}$	\emptyset	$\{\text{sum}\}$	$\{i\}$
$F^3(\perp)$	\emptyset	$\{\text{sum}\}$	$\{\text{sum}, i\}$	$\{\text{sum}, i\}$	$\{\text{sum}, i\}$	$\{\text{sum}\}$	$\{i\}$

where $\perp = (\emptyset, \dots, \emptyset)$

Exercise

Perform liveness analysis for the following program:

```
x = 42;
while (x > 1) {
    y = x / 2;
    if (y > 3)
        x = x - y;
    z = x - 4;
    if (z > 0)
        x = x / 2;
    z = z - 1;
}
print x
```

Algorithms for determining the *lfp*

In order to obtain the least-fixed point (*lfp*) we use the following chain until we converge:

$$\mathbf{F}^0(\perp) = \perp \leq \mathbf{F}(\perp) \leq \mathbf{F}^2(\perp) \leq \dots \leq \mathbf{F}^n(\perp) \leq \dots$$

Which can be formalized in pseudo-code as:

Algorithm 1: NaiveLFP

```
1  $\mathbf{x} = (\perp, \perp, \dots, \perp);$   
2 while  $\mathbf{x} \neq \mathbf{F}(\mathbf{x})$  do  
3    $\mathbf{x} = \mathbf{F}(\mathbf{x});$   
4 return  $\mathbf{x};$ 
```

Complexity: depends on the height of lattice L^n and the evaluation cost of $\mathbf{F}(\mathbf{x})$.

Convergence: requires n iterations.

Algorithms for determining the *lfp*

Algorithm 1 does not profit from the lattice structure:

- recomputes the inputs that were not changed in the former iteration
- does not take into consideration the inputs that were already computed in the current iteration

In the first example we had:

$$F(x_1, x_2, x_3, x_4, x_5, x_6) = \\ (x_2 \setminus \{x\}, x_3 \setminus \{y\}, x_4 \cup x_5 \cup \{x, y\}, (x_6 \setminus \{z\}) \cup \{x\}, (x_6 \setminus \{z\}) \cup \{y\}, \{z\})$$

and the fixed-point is computed in 6 iterations

	x_1	x_2	x_3	x_4	x_5	x_6
\perp	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
$F(\perp)$	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	$\{z\}$
$F^2(\perp)$	\emptyset	\emptyset	\emptyset	$\{x\}$	$\{y\}$	$\{z\}$
$F^3(\perp)$	\emptyset	\emptyset	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$
$F^4(\perp)$	\emptyset	$\{x\}$	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$
$F^5(\perp)$	\emptyset	$\{x\}$	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$

Comparisson of *lfp* algorithms

$$F(\mathbf{x}) = (x_2 \setminus \{x\}, x_3 \setminus \{y\}, x_4 \cup x_5 \cup \{x, y\}, (x_6 \setminus \{z\}) \cup \{x\}, (x_6 \setminus \{z\}) \cup \{y\}, \{z\})$$

Ideally, an efficient algorithm would reduce the number of iterations:

	x_1	x_2	x_3	x_4	x_5	x_6
\perp	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
$F(\perp)$	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	$\{z\}$
$F^2(\perp)$	\emptyset	\emptyset	\emptyset	$\{x\}$	$\{y\}$	$\{z\}$
$F^3(\perp)$	\emptyset	\emptyset	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$
$F^4(\perp)$	\emptyset	$\{x\}$	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$
$F^5(\perp)$	\emptyset	$\{x\}$	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$

versus

	x_1	x_2	x_3	x_4	x_5	x_6
\perp	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset
$F(\perp)$	\emptyset	\emptyset	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$
$F^2(\perp)$	\emptyset	$\{x\}$	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$
$F^3(\perp)$	\emptyset	$\{x\}$	$\{x, y\}$	$\{x\}$	$\{y\}$	$\{z\}$

Round Robin *lfp*

Gauss-Seidel-like component-wise updates:

- computes x_i based on $x_{j < i}$ from the current iteration
- *lfp* is attained even though the outer iteration may provide a different output than the outer iteration in Alg. 1

Algorithm 2: RoundRobinLFP

```
1  $(x_1, x_2, \dots, x_n) = (\perp, \perp, \dots, \perp);$   
2 while  $(x_1, x_2, \dots, x_n) \neq \mathbf{F}(x_1, x_2, \dots, x_n)$  do  
3   for  $i \in 1 \rightarrow n$  do  
4      $x_i = \mathbf{F}_i(x_1, x_2, \dots, x_n);$   
5 return  $(x_1, x_2, \dots, x_n);$ 
```

Complexity: depends on the height of lattice L^n and the cost of evaluating $\mathbf{F}_i(\mathbf{x})$.

Convergence: requires at most n iterations.

Stochastic component wise updates:

- the order of operations is not important in Alg. 2
- computes x_i based on the x_j 's from the current iteration; in no particular order
- Alg. 2 still recomputes the unchanged inputs from the former iteration
- the constraints need to be applied until convergence

Algorithm 3: ChaoticLFP

```
1  $(x_1, x_2, \dots, x_n) = (\perp, \perp, \dots, \perp);$   
2 while  $(x_1, x_2, \dots, x_n) \neq \mathbf{F}(x_1, x_2, \dots, x_n)$  do  
3    $i = \text{random}(1 \rightarrow n);$   
4    $x_i = \mathbf{F}_i(x_1, x_2, \dots, x_n);$   
5 return  $(x_1, x_2, \dots, x_n);$ 
```

Complexity: first, it depends on how i is chosen at each iteration, then it depends on the height of lattice L^n and then on the cost of evaluating $\mathbf{F}_i(\mathbf{x})$.

Convergence: the algorithm is not guaranteed to stop, but if it does the result is correct.

SimpleWorkList *lfp*

Define the map $dep : Nodes \rightarrow 2^{Nodes}$, where $dep(v)$ is the set of nodes whose information depends on the information of node v .

Algorithm 4: SimpleWorkListLFP

```
1  $(x_1, x_2, \dots, x_n) = (\perp, \perp, \dots, \perp);$ 
2  $W = \{v_1, v_2, \dots, v_n\};$ 
3 while  $W \neq \emptyset$  do
4    $v_i = W.removeNext();$ 
5    $y = F_i(x_1, x_2, \dots, x_n);$ 
6   if  $y \neq x_i$  then
7      $x_i = y;$ 
8     for  $v_j \in dep(v_i)$  do
9        $W.add(v_j);$ 
10 return  $(x_1, x_2, \dots, x_n);$ 
```

The functions *removeNext()* and *add()* choose a random node from W and add a node in W , respectively.

Properties

- computes only the particular constraints for the current node
- does not recompute the inputs unchanged during the former iteration
- computes x_i based on the x_j 's from the current iteration; in no particular order
- constraints are applied until convergence
- each iteration has the same effect as the Alg. 3 iteration

Complexity: depends on the number of nodes n , the lattice height h , and the cost of evaluating $F_i(\mathbf{x})$.

Convergence: a single iteration either decreases the work volume in W or climbs up the lattice; the algorithm stops in finite time because the lattice height is finite and the iterations stop when W is void.