CAP — Catch up course

Today: browse through several topics that will be useful in the second part of the CAP course

- 1. IMP and Hoare logic
- 2. Caml
- 3. Partial orders and fixpoints
- 4. Operational semantics for IMP and FUN more inference rules



A first example

everybody should be able to read the following program, and understand what each command does:

```
Q := 0;
R := X;
while R >= Y do (
   Q := Q+1;
   R := R-Y;
)
```

The grammar of IMP

```
an infinite set \mathcal V of variable identifiers X,Y,Z,\ldots arithmetical expressions a ::= X \mid a_1 + a_2 \mid a_1 * a_2 \mid -a \mid 1,2,3,\ldots programs p ::= X := a \mid p_1; p_2 \mid \text{if } a \geq 0 \text{ then } p_1 \text{ else } p_2 \mid \text{while } a \geq 0 \text{ do } p \mid \text{skip}
```

- ▶ skip: program that does nothing (maybe it is X := X)
- we could have boolean expressions, and programs of the form if b then p_1 else p_2 , while b do p

$$b ::= a \geq 0 \mid \neg b \mid b_1 \wedge b_2$$

Reasoning about the execution of IMP programs: Hoare logic

$$\{A\} p \{B\}$$

if the initial state satisfies assertion A, and if the execution of program p terminates, then the final state satisfies assertion B

The rules of Hoare logic, and how to read them

inference rules

Building a derivation in Hoare logic

on the board

Other examples

```
\{true\} while true do skip \{false\}
```

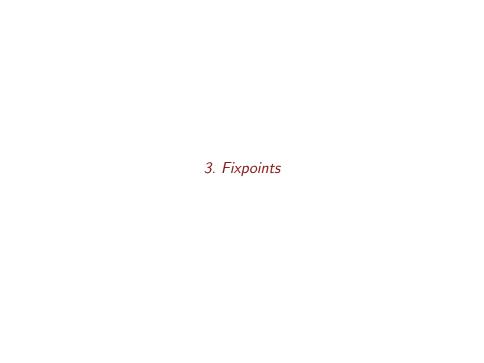
```
 \{X = N \land N > 0\} 
Y := 1;
while X>0 do (
Y := Y * X;
X := X - 1;
);
 \{Y = N!\}
```

2. A real functional programming language: OCaml

Demo

file camlcatchup.ml

(will be made available from the www page of the course)



Partial orders

- ▶ a partially ordered set (poset) is given by (S, \sqsubseteq) , where relation \sqsubseteq is reflexive, transitive, antisymmetric
 - examples: (\mathbb{N}, \leq) $(\mathcal{P}(S), \subseteq)$
 - ► Hasse diagrams

Partial orders

- ▶ a partially ordered set (poset) is given by (S, \sqsubseteq) , where relation \sqsubseteq is reflexive, transitive, antisymmetric
 - examples: (\mathbb{N}, \leq) $(\mathcal{P}(S), \subseteq)$
 - Hasse diagrams
- ▶ let (S, \sqsubseteq) be a poset, and consider $E \subseteq S$ (E is a subset of S)
 - ▶ $u \in S$ is an **upper bound** of E if $\forall x \in E, x \sqsubseteq u$
 - ▶ $u \in S$ is a **least upper bound (lub)** of E if all upper bounds of E are above u, that is, $\forall u' \in S, (\forall x \in S, s \sqsubseteq u') \Rightarrow u \sqsubseteq u'$ a least upper bound of E is written $\cup E$
 - symmetrically, lower bound and greatest lower bound (glb), written ∩E
- ► NB: glbs and lubs do not always exist. two remarkable such elements, when they exist, are
 - $ightharpoonup \bot = \cap S$, the least element of S, and
 - ightharpoonup $\top = \cup S$, the greatest element of S

Particular kinds of partial orders

- ▶ a **complete lattice** is a poset (S, \sqsubseteq) with "everything":
 - ▶ for any $E \subseteq S$, $\cap E$ and $\cup E$ exist
 - lacktriangle so in particular, ot and ot also exist

examples:

- $\blacktriangleright (\mathcal{P}(S),\subseteq,\emptyset,S,\cup,\cap)$
- $\qquad \qquad \bullet \ \, \big(\mathbb{Z} \cup \{-\infty, +\infty\}, \, \leq, \, -\infty, \infty, \max, \min\big)$

Particular kinds of partial orders

- ▶ a **complete lattice** is a poset (S, \sqsubseteq) with "everything":
 - ▶ for any $E \subseteq S$, $\cap E$ and $\cup E$ exist
 - lacktriangle so in particular, ot and ot also exist

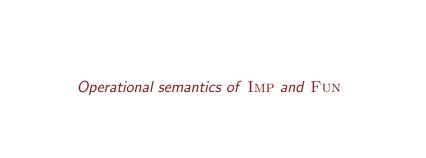
examples:

- $\blacktriangleright (\mathcal{P}(S),\subseteq,\emptyset,S,\cup,\cap)$
- $\qquad \qquad \bullet \quad \big(\mathbb{Z} \cup \{-\infty, +\infty\}, \, \leq, \, -\infty, \infty, \max, \min \big)$
- ▶ a complete partial order (cpo) is a poset (S, \sqsubseteq) such that for every *chain* $C \subseteq S$, $\cup C$ exists
 - ▶ a chain is some $\{x_1, x_2, ...\} \subseteq S$ such that $\forall i, x_i \sqsubseteq x_{i+1}$
 - ▶ a cpo has a least element, which is UØ

Two theorems about fixpoints

- ▶ Knaster-Tarski: the set of fixpoints of a monotone function $f: L \rightarrow L$, where L is a complete lattice, forms a complete lattice.
- ▶ Kleene: if f is a continuous function on a complete partial order, then $\bigcap \{\bot, f(\bot), f(f(\bot)), ...\}$ is the least fixpoint of f continuous: $f(\cup D) = \cup f(D)$ for D directed (i.e. $\forall x, y \in D$, x and y have an upper bound in D)

- these two theorems exist, and their proofs are rather elementary
- frequently used tools when reasoning mathematically about programs and their runs



The operational semantics of IMP programs

exists in several flavours, we define here the big step operational semantics

$$\sigma, p \Downarrow \sigma'$$

The execution of program p in initial state σ terminates and yields final state σ' .

what's a "state"?

- ▶ a memory state, an environment
- ▶ a map from variables to integers $\sigma: \mathcal{V} \to \mathbb{Z}$ given some program p, σ is a partial mapping from a *finite set* of variables to \mathbb{Z}
- ▶ σ , $p \Downarrow \sigma'$ is called a *judgment* it is defined by inference rules

on the board

A small functional programming language: Fun

grammar

an infinite set of Fun variables x,y,z,\ldots

 $ext{FUN programs}$ (are expressions) $e ::= e_1 \ e_2 \ ig| \ ext{fun} \ x o e \ ig| \ ext{let} \ x = e_1 \ ext{in} \ e_2 \ ig| \ x \ ig| \ 1, 2, 3, \dots$

big step operational semantics

$$e \Downarrow v$$

- \triangleright v is a value $v ::= \text{fun } x \rightarrow e \mid 1, 2, 3, \dots$
- no environment