Technische Universität München
Institut für Informatik
Prof. Tobias Nipkow, Ph.D.
Fabian Huch

WS 2021/22
18.01.2022

Semantics of Programming Lectures

Exercise Sheet 11

Exercise 11.1 Complete Lattice over Lists

Show that lists of the same length – ordered point-wise – form a partial order if the element type is partially ordered. Partial orders are predefined as the type class *order*.

instantiation *list* :: (*order*) *order*

Define the infimum operation for a set of lists. The first parameter is the length of the result list.

definition *Inf_list* :: *"nat ⇒ ('a::complete_lattice) list set ⇒ 'a list"*

Show that your ordering and the infimum operation indeed form a complete lattice:

interpretation
Complete_Lattice "{xs. length xs = n}" "Inf_list n" **for** *n*

Exercise 11.2 Fixed Point Theory

Let $'a$ be a complete lattice with ordering $\leq$ and $f{::}'a \Rightarrow 'a$ be a monotonic function. Moreover, let x_0 be a post-fixpoint of f, i.e., $x_0 \leq f\ x_0$. Prove:

$$\bigsqcup\{f^i(x_0) \mid i{\in}\mathbb{N}\} \leq \bigsqcup\{f^{i+1}\ (x_0) \mid i{\in}\mathbb{N}\}$$

Hint: The least upper bound satisfies the following properties

$$x{\in}A \Longrightarrow x \leq \bigsqcup A \qquad \text{(upper)}$$

$$(\forall\, x{\in}A.\ x \leq u) \Longrightarrow \bigsqcup A \leq u \qquad \text{(least)}$$

General homework instructions The exercises on this sheet are similar to pen&paper exam questions. Solve them on pen&paper, then fill out your solution into template on the submission system.

Homework 11.1 Collecting Semantics

Submission until Sunday, Jan 23, 23:59pm.

Consider a version with two (informally given) extensions:

- the arithmetic expressions from last weeks sheet
- a Do-While construct, where the command is executed first and then the condition is checked for the next iteration.

Annotations and the step function stay the same for all other constructs. For the new construct, the annotations can be inferred from the following annotated command:

```
|  x := 2 {A0};
|  DO {A1}
|    IF x < 3 THEN {A2} x := x * x {A3}
|              ELSE {A4} x := y / x {A5}
|    {A6}
|  WHILE 0 < x
|  {A7}
```

Compute the collecting semantics: Show how the annotations change with each application of the step function, until you reach a fix-point. On paper, use the tabular notation:

	0	1	2	3	4	5	6	7	8	9	10
$A0$											
$A1$											
$A2$											
$A3$											
$A4$											
$A5$											
$A6$											
$A7$											

For brevity, abbreviate a state $<x := k,\ y := j>$ by k,j when you fill in the table. Entries that do not change can be left blank.

Write down all changed entries in your pen&paper table, as a (xy-)tuple per state and with the number of the header column as *col* index:

type_synonym *st* = *"int ∗ int"*
type_synonym *col* = *"nat ∗ (st set)"*
type_synonym *row* =*"col list"*

definition *table* :: *"row list"*

Homework 11.2 Counterexamples

Submission until Sunday, Jan 23, 23:59pm.

We know that least pre-fixpoints of monotone functions are also least fixpoints.

1. Show that leastness matters: find a (small!) partial order with a monotone function that has a pre-fixpoint that is not a fixpoint.
2. Show that the reverse implication does not hold: find a partial order with a monotone function function that has a least fixpoint that is not a least pre-fixpoint.

Do pen&paper-proofs, either as comment in your submission file or send it by e-mail.

Homework 11.3 Linter Survey (bonus)

Submission until Sunday, Jan 23, 23:59pm. This is a bonus exercise worth 2 points.

Fill out the survey about the linter tool at https://forms.gle/bqxEm4uwSvk1wGXA6.

The survey is anonymous, but will have a code at the end – fill it in here:

definition *code* :: *string*