Homework SOS
Master Science Informatique 2017–2018

This homework is handed out on 12th December 2017 and is due 12th January 2018. Answers should be sent by mail to `Delphine.Demange@irisa.fr` *and* `Thomas.Jensen@irisa.fr`*. The deadline is strict. You can prepare the work in groups of one or two people (send one copy per group). Questions which are marked with * are more open and will be evaluated by taking into account the technical depth of your reflexion.*

We extend the While language with arrays and operations for manipulating these. The arrays can contain either integers or other arrays. The new syntax is given below.

Arithmetic expressions

$$\begin{array}{rcl} a & ::= & n \mid x \mid a_1 \ o \ a_2 \quad o \in \{+, -, \times\} \\ & & \mid x.\texttt{length} \mid x[a] \end{array}$$

Boolean expressions (unchanged)

$$\begin{array}{rcl} b & := & \texttt{true} \mid \texttt{false} \mid a_1 \ o \ a_2 \quad o \in \{=, \leq\} \\ & & \mid \texttt{not} \ b \mid b_1 \ \texttt{and} \ b_2 \mid b_1 \ \texttt{or} \ b_2 \end{array}$$

Commands

$$\begin{array}{rcll} S & ::= & \texttt{x} := a \mid \texttt{skip} \mid S_1;\ S_2 \mid \texttt{if} \ b \ \texttt{then} \ S_1 \ \texttt{else} \ S_2 \mid \texttt{while} \ b \ \texttt{do} \ S & \\ & & \texttt{x} := \texttt{new} \ \mathit{arr\_dcl}(a) & \text{(creation of array of length } a\text{)} \\ & & x[a_1] := a_2 & \text{(array update)} \\ \mathit{arr\_dcl} & ::= & \texttt{array[int]} \mid \texttt{array}[\mathit{arr\_dcl}] & \end{array}$$

$n \in \mathit{Num},\ \texttt{x} \in \mathit{Var},\ a, a_1, a_2 \in \mathit{Aexp},\ b, b_1, b_2 \in \mathit{Bexp},\ S, S_1, S_2 \in \mathit{Comm}$

Example:

```
x := new array[int](10);
x[0] := 42;
y := x;
i := 1;
while (i ≤ x.length − 1) do (x[i] := y[i − 1]; i := i + 1)
```

Because of aliasing, such a program ends with both x and y equal to $[42, 42, 42, 42, 42, 42, 42, 42, 42, 42]$.

Question 1. Give a small-step semantics to this extended language:

- Define the new **State** domain.
- Define the new denotational semantics of expressions.
- Define the new structural operational semantics (small-step) of commands.

We expect to see a short but precise discussion of the choices that you make when defining the new domain and semantics. In particular, discuss the way arrays are initialized, and errors are modeled.

Question 2(*). Would your choices have an impact on a big-step semantics? A formal definition of a big-step semantics is not expected.

Question 3. We suppose that all instructions are given a unique label. Propose a data-flow analysis for determining the potential may-aliases at each program point. You may want to use the allocation site of an array as abstraction but other abstract domains for modelling memory allocation are also welcome. The allocation site of an array is the label of the `new` instruction that created it.

Question 4(*). Discuss which language features make the alias analysis lose precision. Describe briefly how other analyses could help improving your alias analysis (a formal definition of these other analyses is not expected).

Question 5. The goal is now to define a type system for the While language. We suppose that every variable is given one (and only one) type, thoughout the whole program. We write $\Gamma(x)$ for the type of a variable x.

- Give the definition of types you consider in your type system.
- Define what it means to be well-typed and which typing your system is intended to guarantee. The notion of being well-typed should at least enforce that a program does not try to add two arrays or to index an integer.
- To define the type judgment $\Gamma \vdash S$ ("command S is well-typed in the typing context Γ"), give the corresponding typing rules, and explain why your rules enforce the guarantees you previously choosed.