

Exercises

This exam contains four exercises:

- Static Semantics (25%)
 - Dynamic Semantics (25%)
 - CCS and Equivalences (30%)
 - Structural Induction (20%)
-

Grading

Be sure to *motivate*, *explain*, and *argue* for the reasoning behind your solutions! Be concise and to the point (not necessarily "the more explanation the better"); include only issues relevant to the problem at hand (irrelevant issues may subtract points). All this is what wins you the points (i.e. there are no/very few points for unmotivated solutions "*out-of-the-blue*", correct or not).

The actual grading is done relative to the *course objectives*; i.e., that you demonstrate the ability:

- to *describe* formally the meaning of a wide range of programming constructs;
- to *explain* fundamental concepts, techniques, and results regarding formal semantics of programming languages;
- to *implement* semantic descriptions in familiar programming languages;
- to *analyze* the meaning of a wide range of programming constructs;
- to *compare* semantic descriptions;
- to *reason about* semantic descriptions;
- to *prove* consequences of semantic descriptions; and
- to *apply* all above skills to concrete programs (to understand and prove properties of programs).

...wherever relevant.

Damage Control

If you run out of time or cannot solve an exercise, *identify* and *restrict* your solution to a suitable subset (make this clear in what you hand in). This will, of course, reduce your grade (but at least you will get some points instead of none for no solution).

Write Clearly!

Write clearly; "*we cannot grade, what we cannot read*".

Good Luck!

1. Static semantics (25%)

Consider the following language **FRAC** the syntax of which is given by the syntactic category E in the following context-free grammar:

```
 $E$  :  $n$ 
      :  $v$ 
      :  $E + E$ 
      :  $E / E$ 
      :  $E \text{ xor } E$ 
      :  $\text{floor} ( E )$ 
      :  $\text{let } v=E \text{ in } E$ 
```

The symbol $n \in \mathcal{N}_{10}$ ranges over base-ten literals (where $\mathcal{N}_{10} = \{0, 1, \dots, 9\}^+$). The symbol $v \in VAR$ ranges over a finite set of variables (where $VAR = \{x, y, z\}$). Division should always produce fractional values. The operator “xor” is the bit-wise exclusive or on the binary representation of numbers (only defined on integers). The **floor** operation produces the integer value of a fractional (i.e., rounding the number down). The **let** construction should be able to bind both integers and fractionals. Let EXP be the set of all expressions obtainable from the syntactic category E , and let $TYPE$ be the set $\{\text{int}, \text{frac}\}$ (containing the two types **int** for integers and **frac** for fractionals).

- a) Give the *arity* and *signature* of a type relation, “ \vdash_{type} ”, relating a type environment α , an expression E , and a type τ :

$$\alpha \vdash_{type} E : \tau.$$

- b) Specify a reasonable type checking of the **FRAC** language (i.e., define the relation “ \vdash_{type} ”). *Motivate* and *argue* your choices (e.g., addition can be made to work on arguments of mixed types) and *point out* and *explain* how these choices are reflected in your semantic specification.

Here is a rule for addition (that you may wish to include in your solution):

$$[\text{ADD}_1] \quad \frac{\alpha \vdash_{type} E_0 : \text{int} \quad \alpha \vdash_{type} E_1 : \text{int}}{\alpha \vdash_{type} E_0 + E_1 : \text{int}}$$

2. Dynamic semantics (25%)

Consider once again the following language **FRAC** (as introduced in exercise 1):

```
 $E$  :  $n$   
    :  $v$   
    :  $E + E$   
    :  $E / E$   
    :  $E \text{ xor } E$   
    :  $\text{floor} ( E )$   
    :  $\text{let } v=E \text{ in } E$ 
```

- a) Give the *arity* and *signature* of a big-step semantics evaluation relation, \rightarrow , relating an environment ρ , an initial expression E , and a resulting value r :

$$\rho \vdash E \rightarrow r$$

- b) Specify a reasonable *big-step* dynamic semantics for the **FRAC** language (i.e., define the relation “ \rightarrow ”). *Motivate* and *argue* your choices (e.g., addition can be made to work on arguments of mixed types) and *point out* and *explain* how these choices are reflected in your semantic specification. Note that your dynamic semantic rules should be compatible with your static semantics (from exercise 1).

Here you may use the following mathematic functions:

- $[\cdot]_{10} : \mathcal{N}_{10} \rightarrow \mathbb{Z}$ which extracts semantic number from syntactic base-ten literals;
- $\otimes : \mathbb{Z} \times \mathbb{Z} \rightarrow \mathbb{Z}$ which is bit-wise exclusive or on \mathbb{Z} ;
- $+$: $\mathbb{Q} \times \mathbb{Q} \rightarrow \mathbb{Q}$ which is addition on \mathbb{Q} ;
- $/$: $\mathbb{Q} \times (\mathbb{Q} \setminus \{0\}) \rightarrow \mathbb{Q}$ which is division on \mathbb{Q} ;
- $\lfloor \cdot \rfloor : \mathbb{Q} \rightarrow \mathbb{Z}$ which is the floor operation (rounding down).

Here is a rule for addition (that you may wish to include in your solution):

$$[\text{ADD}_1] \quad \frac{\rho \vdash E_0 \rightarrow z_0 \quad \rho \vdash E_1 \rightarrow z_1 \quad z_0 \in \mathbb{Z}, z_1 \in \mathbb{Z}, z = z_0 + z_1}{\rho \vdash E_0 + E_1 \rightarrow z}$$

- c) Discuss *static* and *dynamic* errors in the language **FRAC** and in your specification of **FRAC**.
-

3. CCS and Equivalences (30%)

- a) Give the transition diagram including all derivations (inference trees) for process P defined below using the usual CCS small-step semantics (in the note [CCS, p. 29]):

$$P := p.0 + q.Q \quad Q := q.Q$$

Consider the following subset of CCS (without *tau*, *parallel composition*, *renaming*, and *restriction*):

$$P : 0 \mid a.P \mid P+P \mid K$$

Let $PROC$ be the set of all processes obtainable from the syntactic category P . Let $\Rightarrow_{\subseteq} PROC \times Act^*$ be defined below. We will write " $P \xrightarrow{\omega} \cdot$ " as a short-hand for " $(P, \omega) \in \Rightarrow$ ".

$$\begin{array}{l}
 \text{[NIL]} \quad \frac{}{0 \xrightarrow{\epsilon} \cdot} \\
 \text{[ACT]} \quad \frac{P \xrightarrow{\omega} \cdot}{a.P \xrightarrow{a\omega} \cdot} \\
 \text{[SUM}_1\text{]} \quad \frac{P \xrightarrow{\omega} \cdot}{P + P' \xrightarrow{\omega} \cdot} \\
 \text{[SUM}_2\text{]} \quad \frac{P' \xrightarrow{\omega} \cdot}{P + P' \xrightarrow{\omega} \cdot} \\
 \text{[CON]} \quad \frac{P \xrightarrow{\omega} \cdot}{K \xrightarrow{\omega} \cdot} \quad K =_{\text{def}} P
 \end{array}$$

- b) Give all derivations (inference trees) for process P using this big-step semantics.
- c) Discuss essential differences between the big-step semantics given above and the *small-step semantics* (appropriate subset of the definition in the note [CCS, p. 29]) in relation to the process P (defined earlier).
- d) Let *small-step trace equivalence* on processes be denoted by \approx_{tr} . The above big-step semantics now induces a notion of big-step trace equivalence on processes (i.e., the relation $\approx_{BS\text{-}tr} \subseteq PROC \times PROC$). How are the two relations \approx_{tr} and $\approx_{BS\text{-}tr}$ related?:

- 1) $\approx_{tr} = \approx_{BS\text{-}tr}$
- 2) $\approx_{tr} \subset \approx_{BS\text{-}tr}$
- 3) $\approx_{tr} \supset \approx_{BS\text{-}tr}$
- 4) $(\approx_{tr} \not\subset \approx_{BS\text{-}tr}) \wedge (\approx_{tr} \not\supset \approx_{BS\text{-}tr})$

Explain your answer.

Hint: look at processes: $R := p.0$ and $S := p.0 + q.0 + q.Q$ in relation to P.

4. Structural Induction (20%)

Consider the language of bit sequences BSQ :

$$B \quad : \quad 0 \quad | \quad 1 \quad | \quad B \ 0 \quad | \quad B \ 1$$

Let the *size* of a bit sequence be defined compositionally by the function $|\cdot| : \text{BSQ} \rightarrow \mathbb{N}$:

$$\begin{aligned} |0| &:= 1 \\ |1| &:= 1 \\ |B \ 0| &:= |B| + 1 \\ |B \ 1| &:= |B| + 1 \end{aligned}$$

Let the semantics of a bit sequence evaluation be defined by the following big-step evaluation relation:

$$\rightarrow_B \subseteq \text{BSQ} \times \mathbb{N}$$

$$\begin{aligned} [\text{ZERO}] & \frac{}{0 \rightarrow_B 0} \\ [\text{ONE}] & \frac{}{1 \rightarrow_B 1} \\ [\text{B-ZERO}] & \frac{B \rightarrow_B n}{B \ 0 \rightarrow_B 2n} \\ [\text{B-ONE}] & \frac{B \rightarrow_B n}{B \ 1 \rightarrow_B 2n + 1} \end{aligned}$$

Let the property $P(B)$ be defined by:

$$P(B) \quad := \quad \forall n \in \mathbb{N} : \langle (B \rightarrow_B n) \Rightarrow (n \leq 2^{|B|} - 1) \rangle$$

- Give the proof structure for a structural induction proof that: $\forall B \in \text{BSQ} : P(B)$.
- Now prove using structural induction that: $\forall B \in \text{BSQ} : P(B)$.