Name: T-Radha Krishna Entry: 2012CS10258 Gp: 02

12.5

Indian Institute of Technology Delhi
Department of Computer Science and Engineering

CSL302
March 23, 2014

Programming Languages
60 minutes

Minor II
Maximum Marks: 60

Open notes. Write your name, entry number and group at the top of each sheet in the blanks provided. Answer all questions in the space provided, in blue or black ink (no pencils, no red pens). Budget your time according to the marks. Do rough work on separate sheets.

Q1. (8 marks) Unification For each of the following pairs of terms, provide the *most general unifier* (in completely simplified form) if it exists, and otherwise indicate why unification fails:

1. $g(h(a,Y),X)$ and $g(X,h(b,Y))$.

$[(X \mapsto h(a,Y)), (X \mapsto h(b,Y))]$

2. $g(h(X,a),Y)$ and $g(Z,h(b,X))$.

$[(Z \mapsto h(X,a)), (Y \mapsto h(b,X))]$

3. $g(h(a,X),X)$ and $g(Y,h(Y,b))$. $\sigma_1 T_1 = X$ $\sigma_2 T_2 = h(h(a,X),b)$

~~$[(Y \mapsto h(a,X)), (X \mapsto h(Y,b))]$~~

unification fails as X is present in h(h(a,X),b)

4. $g(h(a,X),h(X,b))$ and $g(h(a,b),h(Z,X))$.

Unification fails as variable (X) and constant (b) contradicts.

/8

Q2. (8 marks) Denotational Semantics. Consider the abstract syntax of expressions:

Abstract Syntax

$$e \in Exp \quad ::= \quad \ldots \mid \text{if } e_1 \text{ then } e_2 \text{ else } e_3 \text{ fi} \mid \text{let } x \stackrel{def}{=} e_1 \text{ in } e_2 \text{ end} \mid (e_1, e_2)$$

Provide the denotational semantics for *only the new expressions* of this language:

Semantics

Domain(s): $\mathbf{V} = \mathbb{1} \cup \mathbb{2} \cup \mathbb{N} \cup$ ___

Auxiliary concepts: $\rho \in ValAssgn = \mathcal{X} \to \mathbf{V}$

Semantic function(s):

$value[\![\_]\!]\_ : Exp \to ValAssgn \to \mathbf{V}$

0

Name: Entry: Gp

Q3. (4 marks) **Lookup: Prolog programming.**
Write a Prolog program that given a representation of type assumptions as a list of variable-type pairs of the form $[p(x_1, \tau_1), \ldots, p(x_n, \tau_n)]$ for some $n \geq 0$, and a variable Y, returns the type τ, if Y is in the domain of the type assumption.

```
lookup( [], Y, Tau) :- false

lookup ( [ p(X1,Tau1) | Rest ], X1, Tau1) :- True

lookup ( [ p(X1,Tau1) | Rest ], Y, Tau) :- lookup(Rest, Y, Tau)
```

4/4

Q4. (12 marks) **Type Checking Expressions: Prolog programming.**
Consider the following typing rules. Code the rules in Prolog by defining a relation has_type(*Gamma*, *E*, *Tau*), which expresses that under Type Assumptions *Gamma*, expression *E* has type *Tau*. (You only need to write the clauses of the relation for the three kinds of expressions for which the rules are given below.)

$$\frac{}{\Gamma \vdash x : \Gamma(x)} \quad (x \in dom(\Gamma))$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \qquad \Gamma \vdash e_2 : \tau_2}{\Gamma \vdash (e_1, e_2) : \tau_1 \times \tau_2}$$

$$\frac{\Gamma \vdash e_1 : \text{boolean} \qquad \Gamma \vdash e_2 : \tau \qquad \Gamma \vdash e_3 : \tau}{\Gamma \vdash \text{if } e_1 \text{ then } e_2 \text{ else } e_3 \text{ fi} : \tau}$$

$$\frac{\Gamma \vdash e_1 : \tau_1 \qquad \Gamma[x : \tau_1] \vdash e_2 : \tau_2}{\Gamma \vdash \text{let } x \stackrel{def}{=} e_1 \text{ in } e_2 \text{ end} : \tau_2}$$

4.5/12

1.) has_type (Gamma, pair(E1,E2), (τ1 X τ2)) :- tuple(τ1, τ2)
2) has_type (Gamma, E1, τ1), has_type (Gamma, E2, τ2)
pair (E1, E2) :- (E1, E2)

2

has_type (Gamma, if_else (E1, E2, E3), τ) :-
has_type (Gamma, E1, boolean),
has_type (Gamma, E2, τ1), has_type (Gamma, E3, τ2)
has_type (Gamma, E
if_else (True, E2, E3) :- E2
if_else (False, E2, E3) :- E3

2.5

) has_type (Gamma, E1, τ1), has_type (Ga

Entry: Gp:

Q5. (12 marks) **Small-step/Reduction Semantics in Contextual Form**. Provide *small-step* operational semantics (i.e., *reduction* semantics) for the conditional expressions if e_1 then e_2 else e_3 fi, for pairing (e_1, e_2) and projection $\mathbf{proj}_i^2\ e$ in the Contextual form assuming a strategy where we evaluate subexpression e_1 first in the conditional, and we have a *left*-first reduction strategy for pairs. You need to specify the following:

Normal forms:

One hole reduction contexts: $C_1^{(L)}[\,] ::=$

New Reduction Rules, with redex underlined:

0/12

Q6. (8 marks) **Static Semantics: Typing Rules.** Consider the following language of lists (for Q6 and Q7):

$$e ::= \ldots \mid [\,] \mid e_1 :: e_2 \mid hd\ e \mid tl\ e$$

where $[\,]$ is the empty list, and $::$ is the operator that prefixes the element e_1 to the list e_2, and hd and tl extract the first element and the rest of the list respectively of the given list e.

Provide typing rules for the four new kinds of expressions dealing with list construction and deconstruction.

0/8

Name: Entry: Gp:

Q7. (8 marks) **Big-step Semantics: Calculation.** Extend the big-step semantics for expressions by providing the input-output calculation rules for (only) these four new kinds of expressions, i.e., define the new rules for the relation $\gamma \vdash e \Longrightarrow a$, after clarifying what the set *Answers* is.

Normal Forms. $a \in Answers ::=$ ___

0/8