UPDATE

CSCI 3155 : Practice Midterm Exam 2 (conceptual)

Summer 2018 : Tuesday, June 27, 2017

- Please begin by acknowledging the CU Honor Code Pledge by signing in the space below the pledge

  On my honor, as a University of Colorado Boulder student, I will neither give nor receive unauthorized assistance on this work.

  Signature: _____

- Please then **print your name on this page** in the space provided below. Please **ALSO write your name at the top of each page** in case the pages become separated.

- You have 80 minutes for this exam.

- You may use one side of a single 8.5"x11" sheet of handwritten notes you created. Please turn in this page of notes with your exam – please make sure your name is on it if you would like it back. Otherwise, this exam is closed book, closed computer, and closed mobile device.

- This exam has 5 questions for a total of 100 points. Plus a 6th question that counts for 5 bonus points. The maximum score you can receive on the exam is 100/100.

- Answer the Questions in the space provided on the front side of the question sheets. You may use the back side for scratch work. The back side should **NOT** contain answers unless clearly marked and referenced.

- This exam has 11 pages, front only, including 1 scratch page, 1 info page, and this cover page. You are **encouraged** to tear off the last 2 pages of the exam to use as you see fit while you are taking the exam, and then re-staple them to the exam upon submission.

Name (please print): _____

Question :	1	2	3	4	Total	5	Raw	Reported
Points :	25	20	20	35	100	5	105	100
Score :								

1. **25 points : Understanding Grammars** :: Each of the sub-questions of question 1 is dependent on the following grammar, Grammar 1 :

   Grammar 1 :

   S ::= n | S & n | n * S

   n is meta-variable for a number

   a. **7 points :** The sentence “1 & 2” exist in the language defined by Grammar 1. Use the space below to draw a parse tree to prove that the sentence exists in the language.

   b. **5 points :** Grammar 1 is ambiguous. Identify a sentence is ambiguously defined by the grammar.

   c. **12 points :** Demonstrate the ambiguity of Grammar 1 by writing two parse trees for the sentence identified in question 1.b.

Parse Tree 1 | Parse Tree 2

2. **20 points : Writing inference rules::** Write a set of inference rules that operates on Grammar 2 (a language over linked lists) using the operational semantic `(S).append(n) ⇨ ***S'`.*** Here '().append()=>' is the judgment form that we could view as ' (Input1).append(Input2) => Output'. The output ***S'*** should represent the result of appending a n to the input list S. Below, we have provided a few judgments that your inference rules should be able to derive.

   Grammar 2 :

   S ::= n -> S | End

   n is a meta-variable for numbers

   Example judgments : (you do NOT need to derive these)

   i. (1->End).append(2) => 1->2->End
   ii. (3->5->End).append(1) => 3->5->1->End

3. **20 points: Performing Derivations::** Consider the information provided bellow, use it to complete question 3.a. and 3.b.

Grammar 3 (binary tree):

S ::= X | (S) <- n -> (S)

Operational semantics: Σ S = n

Rules of Inference:

Sum_X

$$\frac{}{\Sigma\ X = 0}$$

Sum_SnS

$$\frac{\Sigma\ S_1 = n_1 \qquad \Sigma\ S_2 = n_2 \qquad n_{sum} = n_1 + n_2 + n}{\Sigma\ (S_1) \leftarrow n \rightarrow (S_2) = n_{sum}}$$

a. **10 point:** Using the inference rules provided, derive that
` $\Sigma$ ((X)<- 5 -> (X))<- 10 -> (X) = 15`

b. **10 points:** Complete the following subparts using the information provided at the top of in question 3:
   i. **2 points:** Identify a sentence representing a binary tree with 3 data points that exists in the language defined by Grammar 3

   ii. **3 points:** If I apply the operational semantic '**Σ** S = n' to the sentence identified in question 3.b.i. what the value of n would be?

   iii. **5 points**: Prove your solution to question 3.b.ii.

4. **35 points: Coding::** All subparts of question 4 use the information provided on the last sheet of the exam. Feel free to tear off that sheet if you have not already. You can staple it back to your exam upon submission.

    a. **10 points :** In Scala, implement the logic of the provided inference rules over the operational sematic `S.max() = n` by writing a function named **'max`** that takes as input something of type **'S'** and returns something of type **'Int`**.

    b. **10 points :** In Scala, implement the logic of the provided inference rules over the operational sematic `S.min() = n` by writing a function named **'min`** that takes as input something of type **'S'** and returns something of type **'Int`**.

c. **15 points :** In Scala, implement the logic of the provided inference rules over the operational sematic `S.ordered() = b` by writing a function named **'ordered`** that takes as input something of type **'S'** and returns something of type **'Boolean`.** You may call the functions defined in 4.a. and 4.b.

BONUS QUESTIONS :

5. **5 Bonus Points :** the bonus questions are worth 5 points collectively. I encourage you to try them **AFTER** you have completed the exam. The maximum score that you can receive on this exam is 100/100. If you write a perfect exam with the bonus point you'll get 100/100 not 105/100. I am not putting these questions in the practice exam but I'll note what kinds of questions you might see here on the exam.
    a. Write assertions in Scala syntax that test your work
    b. Provided the description of a language that you have already seen in the course, be able to write an unambiguous grammar for the language in BNF form.
    c. Be able to identify the shortest ambiguous sentence in a grammar.

Scratch Sheet:

Grammar 4 :

S ::= e | (S) <- n -> (S)

Meta-variables : n denotes the language of computer Integers. b denotes Booleans.

Operational Semantics : S.ordered() = b S.max() = n S.min() = n

Inference rules :

ordered_e $\dfrac{}{e.ordered() = true}$

ordered_SnS $\dfrac{S_1.min() < n >= S_2.max() \quad S_1.ordered() = b_1 \quad S_2.ordered() = b_2 \quad b = b_1 \,\&\&\, b_2}{S_1 \text{ n } S_2 .ordered() = b}$

min_e $\dfrac{}{e.min() = ???}$ min_enS $\dfrac{}{e \text{ n } S .min() = n}$ min_SnS $\dfrac{S_1.min() = n_1}{S_1 \text{ n } S_2 .min() = n_1}$

max_e $\dfrac{}{e.max () = ???}$ max_Sne $\dfrac{}{S \text{ n } e .max () = n}$ max _SnS $\dfrac{S_2.max() = n_2}{S_1 \text{ n } S_2 .max() = n_2}$

AST definition:

```
/* S */
sealed abstract class S
/* e */
case object Nil extends S
/*  (Sl) <- n -> (Sr) */
case class Node(l:S, d:Int, r:S) extends S
```

A few examples of parsing: The sentence ((e)<-1->(e))<-2->((e)<-3->(e)) is exists in the language defined by our grammar and can be represented as an AST of type S as
Node(Node(Nil,1, Nil),2,Node(Nil,3, Nil))


RAWR!