מושגים בשפות תכנות

תרגיל 3

להגשה עד 13/05/2019

הנחיות כלליות:

- "הספר" מתייחס ל: Benjamin C. Pierce, Types and Programming Languages פרקים 5,9.

1. לכל אחת מהמחרוזות הבאות, קבעו האם ניתן לפרש אותה כמילה חוקית בשפת Unyped Lambda Calculus עפ"י הדקדוק והמוסכמות התחביריות (Syntactic Conventions) שלמדנו. אם כן, ציירו את העץ שמייצג את המילה, בדומה לעצים שהוצגו בשיעור ובתרגול (ובספר).
    a. x y z
    b. x (y z)
    c. λx. λy. λz. x y z
    d. λx. y λz.
    e. λf. (λx. f (x x)) (λx. f (x x))
    f. λf. (λx. f (λy. x x y)) (λx. f (λy. x x y))

2. נתון הביטוי הבא:

(λx. λx. (λx. x x) x)) ((λx. x x x) λx. x)

    a. כתבו ביטוי שמתקבל מהביטוי הנ"ל ע"י alpha-renaming, כך שבביטוי החדש אין משתנה שמופיע בשתי אבסטרקציות שונות.
    b. ציירו את ה AST המתאים לביטוי שמצאת בסעיף a.
    c. כתבו סדרת חישוב (reduction) לביטוי מסעיף a תחת call by value semantics.
    d. כתבו סדרת חישוב (reduction) לביטוי מסעיף a תחת lazy evaluation semantics.
    e. כתבו סדרת חישוב (reduction) לביטוי מסעיף a תחת normal order semantics.

3. נתונות ההגדרות הבאות (Church Booleans):

```
tru = λt. λf. t
fls = λt. λf. f
test = λl. λm. λn. l m n
or = λb. λc. b tru c
```

    a. מצאו חישוב (reduction) תחת call-by-value semantics לביטוי test (or tru fls) a b תחת ההנחה ש a,b מייצגים ערכים (כלומר שייכים לקטגוריה הסינטקטית V משקף 15 של תירגול 5 - בהגדרה אינדוקטיבית (בלי preference) של call by value semantics).
    b. הגדירו ביטוי עבור הפונקציות הלוגיות xor ו-nand (מבלי להשתמש באופרטורים הלוגיים האחרים שהגדרנו בתירגול כ-blackbox). הסבירו מדוע הביטויים שכתבתם אכן מממשים את הפונקציות הלוגיות המתאימות.

4. נתונות ההגדרות הבאות (Church Numerals):

$c_0 = \lambda s.\ \lambda z.\ z$
$c_1 = \lambda s.\ \lambda z.\ s\ z$
$c_2 = \lambda s.\ \lambda z.\ s\ (s\ z)$
$c_3 = \lambda s.\ \lambda z.\ s\ (s\ (s\ z))$
… (c_k for any natural number k)
$scc = \lambda n.\ \lambda s.\ \lambda z.\ s\ (n\ s\ z)$
$plus = \lambda m.\ \lambda n.\ \lambda s.\ \lambda z.\ m\ s\ (n\ s\ z)$
$times = \lambda m.\ \lambda n.\ m\ (plus\ n)\ c_0$

a. מצאו חישוב (reduction) תחת normal order semantics לביטוי $scc\ c_0$. האם התוצאה שווה ל c_1?
b. מצאו חישוב (reduction) תחת call-by-value semantics לביטוי $scc\ c_0$. האם התוצאה שווה ל c_1? באיזה מובן התוצאה שקולה ל c_1?
c. מצאו דרך אחרת להגדיר את scc, שתהיה עדיין פונקציית העוקב עבור Church Numerals הסבירו מדוע הביטוי שכתבתם עונה על הדרישה.
d. הגדירו פונקציה power להעלאת מספר בחזקה. הסבירו מדוע הביטוי שכתבתם עונה על הדרישה.
e. הגדירו פונקציה isodd, שתקבל Church Numeral ותחזיר Church Boolean, ותאפשר לבדוק האם מספר הוא אי-זוגי או לא. הסבירו מדוע הביטוי שכתבתם עונה על הדרישה.

5.

a. השתמשו ב Y-combinator כדי להגדיר פונקציה sum-l שתעבוד ב lazy evaluation semantics, ובהינתן Church Numeral עבור k, תחשב Church Numeral שמתאים לסכום כל המספרים הקטנים או שווים ל k. לדוגמה:

$sum\text{-}l\ c_0 \Rightarrow^* c_0$
$sum\text{-}l\ c_3 \Rightarrow^* c_6$ // 6 = 1 + 2 +3
$sum\text{-}l\ c_{10} \Rightarrow^* c_{55}$ // 55 = 1 + … + 9 + 10
$sum\text{-}l\ c_k \Rightarrow^* c_{1+\ldots+k}$

לצורך הגדרת הפונקציה sum-l, ניתן להשתמש בכל הפונקציות שמופיעות בשאלות 3 ו- 4, וכן בפונקציה prd שמקיימת:

$prd\ c_0 \Rightarrow^* c_0$
$prd\ c_{k+1} \Rightarrow^* c_k$

הערה: אין צורך להגדיר את הפונקציה prd. ההגדרה המלאה שלה מופיעה בעמוד 62 בספר.

b. כתבו סדרת חישוב לביטוי $sum\text{-}l\ c_1$ תחת lazy evaluation semantics. בסדרה ניתן לבצע חישוב של prd בצעד אחד. לאחר שהגעתם לביטוי שלא מתקדם ב lazy evaluation semantics, המשיכו לפתח אותו תחת normal order semantics.
c. מה יקרה אם נחשב את הביטוי $sum\text{-}l\ c_1$ תחת call by value semantics? הסבירו והדגימו.
d. השתמשו ב Z-combinator ובפונקציה prd כדי להגדיר פונקציה sum-s שתעבוד ב ב call by value semantics ותקיים: $sum\text{-}s\ c_k \Rightarrow^* c_{1+\ldots+k}$.

e. כתבו סדרת חישוב תחת call by value semantics ל c_1 sum-s. בסדרה ניתן לבצע חישוב של prd בצעד אחד. לאחר שהגעתם לביטוי שלא מתקדם ב call by value semantics, המשיכו לפתח אותו תחת normal order semantics.
רמז: ב call by value semantics בעת הפעלת test על Church Boolean, גם ביטוי ה then וגם ביטוי ה else מחושבים - לכן, וודאי היטב שהפונקציה sum-s אכן מסתיימת!

6. Simply Typed Lambda Calculus

בכל אחד מקביעות הטיפוס הבאות, מצאו טיפוס T כך שהקביעה תתקיים, וכתבו עץ גזירה שמוכיח אותה לפי כללי הטיפוס (typing rules):

a. f:Bool→Bool ⊢ (f (if true then false else true)):T
b. f:Bool→Bool ⊢ (λx:Bool. f (if x then false else true)):T
c. x:Bool, y:Bool, f:T ⊢ (f x y):Bool
d. f:T, y:Bool ⊢ (f (λx:Bool. y) y):Bool
e. ⊢ (λx:Bool. λy:T. y x):Bool→T→Bool→Bool

בהצלחה!