Assignment5

1.b

צ"ל: $(pipe\$\ fs\ cont) = (cont\ (pipe\ fs))$

טענה: $(pipe\$\ fs\ cont) = (cont(pipe\ fs))$

נוכיח באינדוקציה על גודל הרשימה:

בסיס האינדוקציה: *n=0*:

$$a - e[\ (pipe\$\ '()\ cont)] ==> a - e[\ cont('())] = a - e[\ cont\ (pipe\ '())]$$

הנחת האינדוקציה: הטענה מתקיימת לכל $i \le k$ כלומר עבור f_i רשימה בגודל i:

$$(pipe\$\ f_i\ cont) = (cont\ (pipe\ f_i))$$

צעד האינדוקציה: יהי $k \in N$, $n = k + 1$ אזי:

$$a - e[\ (pipe\$\ f_n\ cont)] ==>*$$
$$a - e[\ \big(pipe\$\ (cdr\ f_n)\ (lambda\ (res)(\ compose\$(res\ (car\ f_n)\ cont)))\big) ==>*$$

מהנחת האינדוקציה, נקבל:

$$a - e[\ \Big(\big(lambda\ (res)\ compose\$(res\ f_1\ cont\ )\big)(pipe\ (cdr\ f_n))\Big) ==>*$$

מכך ש-*compose$* שווה ערך ל-*compose* (טענת עזר בהמשך):

$$a - e\Big[\Big(lambda(res)\big(cont(compose\ res\ f_1)\big)\Big)\big(pipe\ (cdr\ f_n)\big)\Big] =$$
$$= a - e\Big[\Big(\Big(cont\big(compose\ \big(pipe\ (cdr\ f_n)\big)f_1\big)\Big)\Big)\Big] =$$

מהגדרת *compose*:

$$= a - e[(cont(\ (\ pipe\ (cdr\ f_n)\ f_1\ ))] =$$
$$(\text{מהגדרת } pipe) = a - e[(cont(pipe\ (f_n)))]$$

המשך בעמוד הבא

כעת נוכיח טענת עזר:

We will prove that $compose\$(f\ g\ cont) = cont(compose(f\ g))$:

Suppose:
for procedures f, g, f$ and g$ are the equivalent CPS procedures.

Proof:

$$((compose\$\ g\$\ f\$\ cont1)x\ cont2)$$

$$= ((lambda(g\$\ f\$\ cont1)(cont1 \left(lambda(x\ c) \left(f\$\ x(lambda(res)(g\$\ res\ c))\right)\right) x\ cont2)$$

Since we assumed f$ and g$ are the equivalent CPS procedures:

$$(f\$\ x\ (lambda(res)(g\$\ res\ c))) = ((lambda(res)(g\$\ res\ c))(f\ x) = (g\$\ (f\ x)\ c) = \left(c\ (g\ (f\ x))\right) \Rightarrow$$

$$((compose\$\ g\$\ f\$\ cont1)x\ cont2) = \left(lambda(g\$\ f\$\ cont1)\left(cont1\ \left(lambda(x\ c)\left(c\ (g\ (f\ x))\right)\right)\right)\right) =$$

$$(\ cont1 \left(lambda(x\ c) \left(\ c(g(\ f\ x))\right)\right))$$

2.a

נאמר כי שתי רשימות עצלות lzl1 ו-lzl2 הן שוות אם:
לכל n מתקיים כי האיבר במקום הn של lzl1 ו-lzl2 שווים.

2.b

נוכיח באינדוקציה על i המייצג את האיבר במיקום ה -i ברשימות:

<u>בסיס עבור i=1 :</u>

לפי הגדרת fib1 מתקיים כי יוחזר האיבר הראשון (head) 0

לפי הגדרת fib2 מתקיים כי יוחזר האיבר הראשון (head) 0 – נשים לב כי פונקציה זו מוגדרת על ידי הזוג (...)0,lambda) ואכן האיבר הראשון הוא 0.

לכן מתקיים לפי ההגדרה כי לכל i<=1 האיבר במקום ה -1 של Lzl-1 ו של Lzl-2, לכן Lzl-2 ו Lzl-1 שווים.

<u>בסיס עבור i=2 :</u>

עבור fib1 ניתן לראות כי יופעל ה tail אשר יחזיר את ה closure שנוצר על ידי הlambda.

ב closure זה מתבצע החישוב של 0 ועוד 1 ששניהם התקבלו כפרמטר להפעלה ראשונית והפעם יוגדר 1 להיות ה -head.

עבור fib2 יופעל ה tail שהוא זוג שהאיבר הראשון שלו הוא 1 והאיבר השני הוא פונקציית המשך. לכן גם כאן מוגדר 1 להיות ראש הזוג , כנדרש.

הנחה :

נניח כי מתקיים לכל k<i כי האיבר במקום ה k- של Lzl-1 ו של Lzl-2 שווה.

צעד : נראה עבור2< i.

נתבונן ב fib1 : תהליך החישוב של האיבר ה i- בסדרה התקבל כך :

ראשית חושבו בתהליך האיברים במיקום i-1, i-2 . נתבונן בשלב החישוב של i :

(lambda (i-2, i-1) (cons-lzl i-2 (lambda () (fibgen i-1 (i))))))

נשים לב כי לאחר שתי איטרציות יופיע i כאיבר הראשון בזוג החדשה :

(i,lambda()->…)

לפי הנחת האינדוקציה מתקיים כי האיברים במיקום i-1 ו i-2 ברשימה fib1 שווים לאיברים במיקום אלו ברשימה fib2. לכן, כיוון שהראינו כי i מתקבל מסכום האיברים במיקום זה, כל מה שעלינו להראות הוא שגם ב fib2- זה קורה.

נתבונן ב fib2 : נתבונן באופן פעולת הפונקציה על האיבר ה- i-1 עד שנגיע לאיבר i ונראה כי i מתקבל מהסכום הנדרש.

בשלב בו נוצרi-1 המצב של הרשימות הוא כזה(בהתבסס על נכונות הפונקציה) :

Lzl1 : cons(i-2, lambda() - >…)

Lzl2 : cons(i-1,lambda()->…)

באיטרציה הבאה יקרה התהליך הבא :

- Lambda של הפונקציה lz-lst-add תקבל כפרמטר את שתי הרשימות
- יוגדר בפונקציה הזוג (i-1+i-2,lambda()->…)

ואכן קיבלנו כי האיבר ה-i התקבל מהסכום הנדרש ולכן בשתי הרשימות מתקיים כי האיבר במקום זה שווה.

כנדרש.

3.1

a. unify[p(v(v(d(M),M,ntuf3),X)),
p(v(d(B),v(B,ntuf3),KtM))]

S={}
A*S = p(v(v(d(M),M,ntuf3),X)),
B*S= p(v(d(B),v(B,ntuf3),KtM))

$FAIL: v\ has\ different\ structure$- takes: three arguments in A, two arguments in B

3.2

b. unify[n(d(D),D,d,k,n(N),K),
n(d(d),D,d,k,n(N),d)]

S={}
A*S = n(d(D),D,d,k,n(N),K)
B*S = n(d(d),D,d,k,n(N),d)

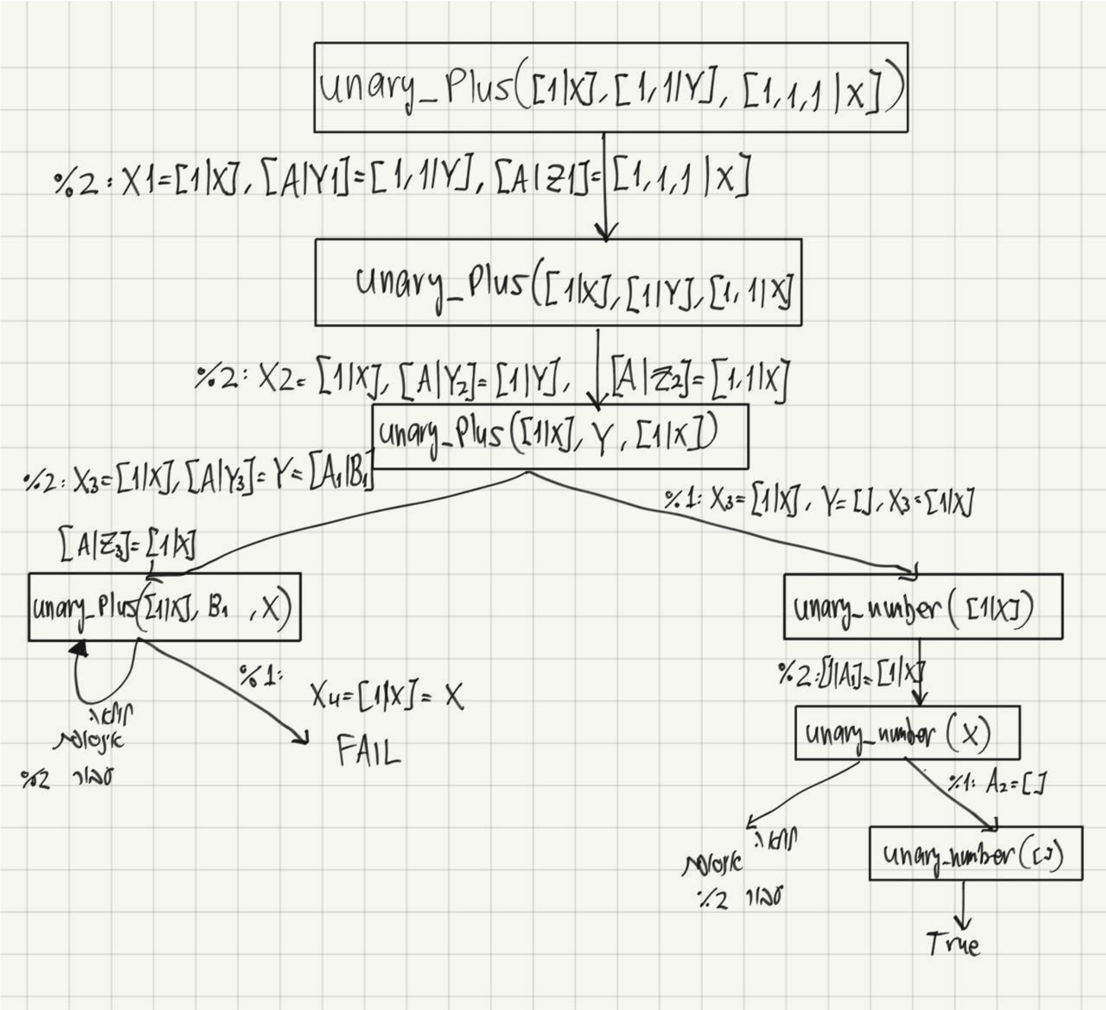
S = S*{D=d} = {D=d}
A*S = n(d(d),d,d,k,n(N),K)
B*S = n(d(d),d,d,k,n(N),d)

S = S*{K=d} = {D=d,K=d}
A*S = n(d(d),d,d,k,n(N),d)
B*S = n(d(d),d,d,k,n(N),d)

⇒ S = {D=d,K=d}

3.3

a.



b. This tree is a success tree since it has a success path.

c. This tree is an infinite tree since the goal ?-unary_number(X) has infinite substitutions.

d. Yes, as we can see in the right sub-tree there is a path to true.

e. L9 is not decidable.

In class we studied that although the number of rules and goals is finite, the number of cons isn't. in addition, in class we added functors and there are finite options of cons.