LÖSNINGAR, Algoritmik 5p, 18 dec 1997

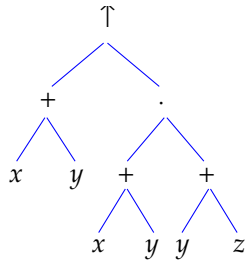
1 a) Visa hur $(x+y)^{(x+y)\cdot(y+z)}$ kan beräknas som en sammansättning av $Add(x,y)$, $Mult(x,y)$ och $Pot(x,y)$.

b) Rita ett binärt träd som representerar uttrycket $(x+y)^{(x+y)\cdot(y+z)}$.

c) Räkna upp det binära trädets noder i postordning.

LÖSNING a) $Pot(Add(x,y), Mult(Add(x,y), Add(y,z)))$

b)



c) $x\ y + x\ y + y\ z + \cdot\ \uparrow$

2 Betrakta stenhögsproceduren

```
r ← 0
Repetera med k = 0 till x{
    Repetera med i = 0 till k{
        Addera i till r}}
```

a) Vilket innehåll får r om proceduren körs med $x = 2$, $x = 3$?

b) Konstruera en primitivt rekursiv funktion som beräknar samma sak som proceduren.

LÖSNING

a) $x = 2$ ger $r = 0 + (0+1) + (0+1+2) = 4$,
$x = 3$ ger $r = 0 + (0+1) + (0+1+2) + (0+1+2+3) = 10$.

b) $\begin{cases} Pyramid(0) = 0 \\ Pyramid(Öka(x)) = Add(TriangelTal(Öka(x)), Pyramid(x)) \end{cases}$

3 Antag att vi är "ute efter" det största av tre naturliga tal. Konstruera för detta syfte

a) $StörstAvTre(x,y,z)$ inom klassen av primitivt rekursiva funktioner,

b) `StörstAvTre x,y,z i r` som en stenhögsprocedur.

LÖSNING

a) $StörstAvTre(x,y,z) = Störst(Störst(x,y), z)$
där $Störst(x,y) = Om(Sub(x,y), x, y)$.

b)
```
StörstAvTre x,y,z i r =
    r ← x
    Sålänge y>r {Öka r}
    Sålänge z>r {Öka r}
```

4 Funktionen *Måns* definieras med hjälp av funktionerna *Bill* och *Bull*. Bestäm $Måns(0)$, $Måns(1)$, $Måns(6)$ samt $Måns(x)$ för godtyckligt x.
Visa hur Du gjorde.

$$Måns(x) = Bill(x,x)$$

$$\begin{cases} Bill(x,0) = Noll?(x) \\ Bill(x, Öka(y)) = Om(Lika(x, Bull(y)), \\ \qquad\qquad\qquad 1, \\ \qquad\qquad\qquad Bill(x,y)) \end{cases}$$

$$Bull(x) = Mult(Mult(x, Öka(x)), Öka(Öka(x)))$$

LÖSNING

Först *Bull*:

x	0	1	2	3	4	5	6
$Bull(x)$	$0\cdot1\cdot2$	$1\cdot2\cdot3$	$2\cdot3\cdot4$	$3\cdot4\cdot5$	$4\cdot5\cdot6$	$5\cdot6\cdot7$	$6\cdot7\cdot8$
	$= 0$	$= 6$	$= 24$	$= 60$	$= 120$	$= 210$	$= 336$

Sedan *Måns*:

$$Måns(0) = Bill(0,0) = Noll?(0) = 1$$
$$Måns(1) = Bill(1,1) = Bill(1,0) = Noll?(1) = 0$$
$$1 \neq Bull(0)$$
$$Måns(6) = Bill(6,6) = Bill(6,5) = \ldots = Bill(6,2) = 1$$
$$6 \neq Bull(5) \quad 6 \neq Bull(4) \quad \cdots \quad 6 = Bull(1)$$

$Måns(x) = 1$ om x är en produkt av tre på varandra följande tal, som t ex $2\cdot3\cdot4$. Annars är $Måns(x) = 0$.

5 Vad beräknar funktionen *Maja?*

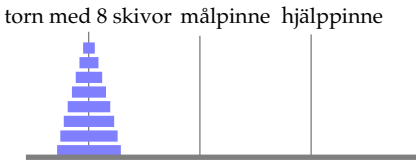
$$Maja(träd) = Öka(Gräddnos(UtomFörsta(träd)))$$

$$\begin{cases} Gräddnos([\,]) = 0 \\ Gräddnos([T|S]) = Störst(Maja(T), Gräddnos(S)) \end{cases}$$

LÖSNING

Maja beräknar ett träds höjd. (Dvs antalet nivåer.) Det hela sker i samarbete (rekursiv fläta) med *Gräddnos* som tar en skog som input beräknar höjden på det högsta trädet i skogen.

6 (Hanois torn)



I slutet av 1800-talet presenterade den franske matematikern Edouard Lucas ett slags matematiskt pussel eller spel (“recréation mathématiques”), som gick ut på att – under iakttagande av vissa spelregler (se nedan) – förflytta ett antal olika stora skivor, som låg travade på varandra (över en pinne) i storleksordning med den största skivan i botten. Efter en lyckad genomförd förflyttning av skivorna, slut de ligga travade (i samma ordning som från början) över en pinne (målpinne) bredvid den ursprungliga. En tredje pinne skulle finnas tillhands som hjälppinne (mellanlandningspinne).

Enligt spelreglerna skulle skivorna flyttas en i taget, från pinne till pinne. Och ingen skiva tilläts någonsin att hamna ovanpå en mindre skiva, eller utanför de tre pinnarna.

STRATEGI: Låt oss kalla ett torn bestående av n skivor för ett *n-torn*. För att flytta ett *n-torn* till en viss målpinne, *måste* man förr eller senare flytta tornets bottenskiva. Men för att komma åt denna, *måste* man först flytta (till hjälppinnen) det $n-1$ *-torn* som bildas av de skivor som ligger ovanpå bottenskivan. Därefter kan man flytta bottenskivan till målpinnen. Sedan *måste* man åter flytta det nämnda $n-1$ *-tornet*. Nu så att det hamnar på målpinnen ovanpå bottenskivan. Man *måste* således utföra minst två förflyttningar av $n-1$ *-tornet* och en förflyttning av bottenskivan. T ex kan en lyckad förflyttning av ett 3-*torn* beskrivas på följande sätt, om $[A\ \ B]$ representerar förflyttning av en skiva från pinne A till pinne B. (Kontrollera detta!)

$[[A\ \ B]\ \ [A\ \ C]\ \ [B\ \ C]\ \ [A\ \ B]\ \ [C\ \ A]\ \ [C\ \ B]\ \ [A\ \ B]]$

NU TILL SJÄLVA UPPGIFTEN: Skriv en listfunktion $Hanoi(n, A, B, C)$ som returnerar den lista (av ovanstående typ) som representerar just de förflyttningar som strategin ger upphov till vid förflyttning av ett *n-torn* från pinne A till pinne B, med C som hjälppinne.

LÖSNING

$$\begin{cases} Hanoi(0, A, B, C) = [\,] \\ Hanoi(Öka(n), A, B, C) = FogaSamman(Hanoi(n, A, C, B), \\ \qquad\qquad FogaIn([A\ \ B], Hanoi(n, C, B, A)) \end{cases}$$

Provkörning:

$$\begin{aligned} Hanoi(1, A, B, C) &= FogaSamman(Hanoi(0, A, C, B), \\ &\qquad FogaIn([A\ \ B], Hanoi(0, C, B, A)) \\ &= FogaSamman([\,], FogaIn([A\ \ B], [\,]) \\ &= FogaSamman([\,], [[A\ \ B]] \\ &= [[A\ \ B]] \end{aligned}$$

$$\begin{aligned} Hanoi(2, A, B, C) &= FogaSamman(Hanoi(1, A, C, B), \\ &\qquad FogaIn([A\ \ B], Hanoi(1, C, B, A)) \\ &= FogaSamman([[A\ \ C]], FogaIn([A\ \ B], [[C\ \ B]]) \\ &= FogaSamman([[A\ \ C]], [[A\ \ B]\ \ [C\ \ B]] \\ &= [[A\ \ C]\ [A\ \ B]\ \ [C\ \ B]] \end{aligned}$$

$$\begin{aligned} Hanoi(3, A, B, C) &= FogaSamman(Hanoi(2, A, C, B), \\ &\qquad FogaIn([A\ \ B], Hanoi(2, C, B, A)) \\ &= FogaSamman([[A\ \ B]\ [A\ \ C]\ \ [B\ \ C]], \\ &\qquad FogaIn([A\ \ B], [[C\ \ A]\ [C\ \ B]\ \ [A\ \ B]]) \\ &= FogaSamman([[A\ \ B]\ [A\ \ C]\ \ [B\ \ C]], \\ &\qquad [[A\ \ B]\ [C\ \ A]\ [C\ \ B]\ \ [A\ \ B]] \\ &= [[A\ \ B]\ [A\ \ C]\ \ [B\ \ C]\ [A\ \ B]\ [C\ \ A]\ [C\ \ B]\ \ [A\ \ B]] \end{aligned}$$

7 Konstruera en funktion som för två träd reder ut om de är lika. Du får här (om Du vill) använda hjälpfunktionen $Lika(x, y)$ som reder ut om två tal är lika. Men du har inte tillgång till någon likhetstestare för listor.

LÖSNING

a) Tag hänsyn enbart till själva trädstrukturerna, inte till nodinnehåll.

a) $$LikaTrädstruktur([nod|S], [nod'|S']) = LikaSkogstruktur(S, S')$$

$$\begin{cases} LikaSkogstruktur([\,], [\,]) = 1 \\ LikaSkogstruktur([\,], [T'|S']) = 0 \\ LikaSkogstruktur([T|S], [\,]) = 0 \\ LikaSkogstruktur([T|S], [T'|S']) = Och(LikaTrädstruktur(T, T'), \\ \qquad\qquad LikaSkogstruktur(S, S')) \end{cases}$$

b) Tag hänsyn inte enbart till trädstrukturerna utan även nodinnehåll.

b) $$LikaTräd([nod|S], [nod'|S']) = Och(Lika(nod, nod'), \\ \qquad LikaSkog(S, S'))$$

$$\begin{cases} LikaSkog([\,], [\,]) = 1 \\ LikaSkog([\,], [T'|S']) = 0 \\ LikaSkog([T|S], [\,]) = 0 \\ LikaSkog([T|S], [T'|S']) = Och(LikaTräd(T, T'), \\ \qquad\qquad LikaSkog(S, S')) \end{cases}$$