Tentamen i Beräkningsmodeller

Lördagen den 24 Februari 2001, kl 8.45 – 13.45 i VV11
Ansvarig lärare: Bengt Nordström, tel 55 45 25
Tillåtna hjälpmedel: Inga.
Börja varje uppgift på nytt blad. Skriv endast på en sida av papperet. Varje svar skall motiveras! Den här skriftliga tentamen utgör en del (75 %) av den totala examinationen, den andra delen (dvs. 25 %) består av de inlämningsuppgifter som har delats ut under kursens gång. För årets och förra årets elever gäller alltså att summan av poängen från inlämningsuppgifterna och den skriftliga tentan skall vara minst 100 för att få godkänt på kursen. Examensvisning kommer att äga rum fredagen den 9 Mars kl 11.00 i Bengt Nordströms tjänsterum. Lösningar till den här tentan kommer att finnas tillgänglig från kursens hemsida.

1. Ge ett exempel på ett λ-uttryck som terminerar vid normal evauleringsordning men inte vid applikativ evalueringsordning. Motivera! (20)

2. Visa hur man kan representera elementen i mängden **Bool** i lambdakalkyl respektive χ! Visa också hur man kan representera if-funktionen samt att den har egenskapen att (⌜true⌝ är representationen av värdet true i respektive beräkningsmodell). (10)

   $$\mathsf{if}\ \ulcorner \mathsf{true} \urcorner\ \mathsf{d}\ \mathsf{e} = \mathsf{d}$$

3. Varför är PRF (mängden av de primitivt rekursiva funktionerna) en dålig modell för allmännt beräkningsbara funktioner? (20)

4. Enligt läroboken är en icketom mängd A uppräkningsbar om det finns en total surjektiv funktion $f \in \mathbf{N} \to A$.

   (a) Är det väsentligt att funktionen skall vara total? (15)

   (b) Är det väsentligt att funktionen skall vara surjektiv? (15)

   Om svaret är ja, skall du motivera det genom att visa att de reella talen skulle vara uppräkneliga om kravet inte finns med i definitionen. Om svaret är nej, skall du visa hur man givet en funktion som inte uppfyller kravet kan konstruera en funktion med kravet uppfyllt.

5. Bevisa att man i Haskell (eller något annat typat funktionellt språk) inte kan skriva en funktion `halt :: (Nat -> Nat) -> Nat -> Bool` som är sådan att `(halt f i)` evaluerar till `true` om `(f i)` terminerar och annars evaluerar till `false` . (20)

6. Antag att vi skulle använda programspråket Java (ni kan byta ut Java mot Ada, Pascal, Basic, Fortran eller Cobol) som beräkningsmodell!

- Vad betyder det att en partiell funktion $f \in \mathbf{N} \tilde{\to} \mathbf{N}$ är Java-beräkningsbar? (15)
- Beskriv vad som behöver göras för att bevisa att den operationella semantiken till Java är Java-beräkningsbar! (15)

7. Antag att $\ulcorner \mathsf{if} \urcorner$ representerar if-funktionen, $\ulcorner \mathsf{iszero} \urcorner$ representerar iszero-funktionen, $\ulcorner \mathsf{succ} \urcorner$ representerar efterföljarfunktionen, $\ulcorner 0 \urcorner$ representerar talet 0, $\ulcorner \mathsf{mult} \urcorner$ representerar multiplikation och $\ulcorner \mathsf{pred} \urcorner$ representerar predecessor-funktionen i λ-kalkyl.

   Varför kan vi inte införa följande definition av fakultetsfunktionen i λ-kalkyl:

$$
\begin{aligned}
\mathsf{fac} = \lambda n.\ulcorner \mathsf{if} \urcorner &(\ulcorner \mathsf{iszero} \urcorner\ n) \\
&(\ulcorner \mathsf{succ} \urcorner\ \ulcorner 0 \urcorner) \\
&(\ulcorner \mathsf{mult} \urcorner\ n\ (\mathsf{fac}(\ulcorner \mathsf{pred} \urcorner\ n)))
\end{aligned}
$$

   Visa hur man uttrycker funktionen fac på ett riktigt sätt i λ-kalkyl! (20)

Lycka till!