Göteborgs Universitet och Chalmers Tekniska Högskola — 21 januari 2002
Datavetenskap — INN111/TIN120

Tentamen i Beräkningsmodeller

Fredagen den 18 januari 2002, kl 8.45 – 12.45.
Ansvarig lärare: Bengt Nordström, tel 1033 eller 0707-87 77 29.
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 8 februari kl 13.15 i Bengt Nordströms tjänsterum. Tentamensresultat och lösningar till tentan kommer att finnas tillgängligt från kursens hemsida.

1. Redogör för Churchs tes. Vad säger den, varför heter den inte Churchs sats, varför tror vi på den? (10)

   Svar: sid 4 i boken

2. I λ-kalkyl kan man inte direkt uttrycka rekursiva funktionsdefinitioner. Redogör för hur man kan uttrycka dem! (20)

   Svar: Se läroboken sidan 175

3. Det finns två olika definitioner av vad det betyder att en mängd A är uppräkningsbar. En tredje skulle vara att mängdens element ryms i ett hotell med oändligt många rum, ett rum för varje naturligt tal. Att mängdens element ryms skulle betyda dels (1) att varje element har minst ett rum, samt (2) att varje rum har högst ett element (vi vill ju inte att alla element skall kunna dela på ett rum). Visa hur dessa två krav uttrycks i de två alternativa definitionerna av uppräkningsbarhet! (20)

   Svar: Den första definitionen är att A är uppräkningsbar om det finns en total surjektiv funktion $f \in \mathbf{N} \to A$. Surjektivitetskravet betyder att varje element har åtminstone ett rum (varje element är bilden av något tal), att f är en funktion betyder precis att varje naturligt tal avbildas av högst ett funktionsvärde, dvs varje rum har högst ett element. Den andra definitionen är att A är uppräkningsbar om det finns en total, injektiv funktion $f \in A \to \mathbf{N}$, där injektiviteten uttrycker att varje funktionsvärde (rum) avbildas av högst ett element, dvs varje rum har högst ett element och totaliteten uttrycker att varje element har minst ett rum.

4. Kan man räkna upp $\wp(\mathbf{N})$, mängden av alla delmängder av $\mathbf{N}$? Motivera! (30)

   Svar: Det kan man inte. Vi kör vårt gamla diagonaliseringsargument igen: Om man kunde det fanns det en total surjektiv funktion $F \in \mathbf{N} \to \wp(\mathbf{N})$ som räknar upp alla delmängder. Vi kan nu definiera en mängd D som inte finns med i uppräkningen genom att låta talet k finnas i mängden D om och endast om $k \notin F(k)$. Om D skulle finnas med i uppräkningen skulle $D = F(i)$, för något i. Vi får nu en motsägelse eftersom vi har att $i \in D$ om och endast om $i \notin F(i)$.

5. Skriv ett program i χ som ej innehåller **rec**-konstruktionen och som ej terminerar. (15)

   Svar: Eftersom χ är ett otypat språk kan vi använda följande term (som ej terminerar) från λ-kalkyl:

   $$(\lambda x.xx)(\lambda x.xx)$$

6. I den här uppgiften skall ni formalisera ett litet språk för aritmetik. Språkets konkreta syntax kan beskrivas på följande sätt: (40)

   $$e ::= x \mid n \mid e+e \mid e*e \mid e+e \mid \sum_{x=e}^{e} e$$

   vilket betyder att ett uttryck antingen är en variabel (t.ex. i), ett tal (t.ex.314), en addition (t.ex. $45+i$), en multiplikation (t.ex. $5*3$) eller en summation (t.ex.$\sum_{x=1}^{1000} 1 + x * i$). Vi antar att vi i en summation $\sum_{x=d}^{e} f$ inte har några förekomster av variabeln x i uttrycken d och e. Den konkreta syntaxen är lite tvetydig, detta har ingen betydelse för uppgiften. Vi kan anta att vi har något sätt att lösa upp tvetydigheterna.

   (a) Ge den abstrakta syntaxen för språket! Ni kan utgå från att vi redan har definierat Z, mängden av heltal.

   (b) Definiera vad det betyder att variabeln x är fri i uttrycket e!

   (c) Definiera substitutionsoperationen $e_1[x \leftarrow e_2]$, det uttryck man får genom att substituera uttrycket e_2 för alla fria förekomster av variabeln x i uttrycket e_1. Vi kan anta att uttrycket e_2 är slutet.

   (d) Varför blir det mer komplicerat att definiera substitutionsoperationen om uttrycket e_2 inte är slutet?

   Svar:

   (a) Vi antar att vi har en given oändlig mängd **Id**. Den abstrakta syntaxen definieras genom följande induktiva definition av mängden Exp:

   $$\begin{array}{rcl}
   \mathbf{var}(i) \in \mathsf{Exp} & \text{om} & i \in \mathbf{Id} \\
   \mathbf{int}(j) \in \mathsf{Exp} & \text{om} & j \in Z \\
   \mathbf{plus}(d, e) \in \mathsf{Exp} & \text{om} & d, e \in \mathsf{Exp} \\
   \mathbf{mult}(d, e) \in \mathsf{Exp} & \text{om} & d, e \in \mathsf{Exp} \\
   \mathbf{sum}(i, d, e, f) \in \mathsf{Exp} & \text{om} & i \in \mathbf{Id}, d, e, f \in \mathsf{Exp}
   \end{array}$$

   (b) Summationsuttrycket $\sum_{x=e_1}^{e_2} e_3$ binder variabeln x i uttrycket e_3. Vi kan ju byta ut variabeln x mot någon annan variabel om vi bara gör samma byte i uttrycket e_3. Därför får vi följande definition av att en variabel är fri i ett uttryck:

Variabeln x är fri i uttrycket x.

Variabeln x är fri i uttrycket $d + e$ om x är fri i d eller i e.

Variabeln x är fri i uttrycket $d * e$ om x är fri i d eller i e.

Variabeln x är fri i uttrycket $\sum_{y=d}^{e} f$ om x inte är lika med y och x är fri i d eller i e.

(c) Jag ger en definition av substitutionsoperationen med hjälp av följande induktion över den abstrakta syntaxen:

$$\begin{aligned}
\mathbf{var}(j)[i \leftarrow e] &= \mathbf{var}(j) \quad \text{om } i \neq j \\
\mathbf{var}(i)[i \leftarrow e] &= e \\
\mathbf{int}(j)[i \leftarrow e] &= \mathbf{int}(j) \\
\mathbf{plus}(d, f)[i \leftarrow e] &= \mathbf{plus}(d[i \leftarrow e], f[i \leftarrow e]) \\
\mathbf{mult}(d, e)[i \leftarrow e] &= \mathbf{mult}(d[i \leftarrow e], f[i \leftarrow e]) \\
\mathbf{sum}(j, d, f, g)[i \leftarrow e] &= \mathbf{sum}(j, d[i \leftarrow e], f[i \leftarrow e], g[i \leftarrow e]) \\
&\quad \text{om } i \neq j \\
\mathbf{sum}(i, d, f, g)[i \leftarrow e] &= \mathbf{sum}(i, d, f, g)
\end{aligned}$$

(d) Antagandet att det uttryck som man substituerar skall vara slutet leder till en enklare definition av substitutionsoperationen av samma skäl som i lambda-kalkyl. Dvs man kan lätt hamna i en situation där de fria variablerna skulle bindas vid substitutionen. Man blir i så fall tvungen att först byta namn på variablerna.

7. Hur formulerar man och bevisar stopp-problemet för Turing-maskiner? (15)

Svar: Anta att vi kan lösa stopp-problemet, dvs. att den finns en Turing-maskin X med egenskapen

$$< X, (\ulcorner T \urcorner, \alpha) > \Rightarrow \begin{cases} 1 & \text{om } (T, \alpha) \text{ terminerar,} \\ 0 & \text{för övrigt} \end{cases}$$

Vi låter $(\ulcorner T \urcorner, \alpha)$ vara en positionerad remsa som kodar paret av $\ulcorner T \urcorner$ och α.

Definiera nu en ny Turing-maskin Y genom att lägga till instruktioner till X på följande sätt: Om $< X, (\ulcorner T \urcorner, \alpha) >$ stannar på remsan 1 så låter vi Y fortsätta att gå åt höger i all oändlighet. Annars låter vi Y terminera.

Vi får alltså att $< Y, (\ulcorner T \urcorner, \alpha) >$ terminerar om och endast om $< T, \alpha >$ ej terminerar.

Nu konstruerar vi en ny Turing-maskin genom att lägga till instruktioner till Y på följande sätt. Låt Z först kopiera sitt indata-band β till (β, β) och exekvera sedan Y.

Terminerar $< Z, \ulcorner Z \urcorner >$? Den kommer att uppföra sig som maskinen $< Y, (\ulcorner Z \urcorner, \ulcorner Z \urcorner) >$. Men $< Y, (\ulcorner Z \urcorner, \ulcorner Z \urcorner) >$ terminerar ej om $< Z, \ulcorner Z \urcorner >$ terminerar. Och $< Y, (\ulcorner Z \urcorner, \ulcorner Z \urcorner) >$ terminerar om $< Z, \ulcorner Z \urcorner >$ ej terminerar ! Motsägelse. Vårt antagande att vi kan lösa stopp-problemet måste alltså vara felaktigt.

Lycka till!