Göteborgs Universitet och Chalmers Tekniska Högskola 2000–02–02
Datavetenskap INN11

Tentamen i Beräkningsbarhet och lambda-kalkyl

Fredagen den 11 februari 2000, kl 8.45 – 13.45 i sal TL1, Hörsalarna
Ansvarig lärare: Bengt Nordström, tel 1033, eller 070-27 000 27
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 av den totala examinationen, den andra delen består av de inlämingsuppgifter som har delats ut under kursens gång. För 1999 års elever gäller att summan av poängen från inlämningsuppgifterna och den skriftliga tentan skall vara minst 100 poäng för att få godkänt på kursen. Examensvisning kommer att äga rum den 25 februari kl 14.15 i MD8.

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

   Svar: Se läroboken sidan 175

2. Som alla vet, är ju List(N) en uppräknelig mängd. Vad är felet i följande "bevis" av motsatsen:

   Antag att List(N) är uppräknelig. Då finns en uppräkning, en surjektiv funktion $\mathsf{f} \in \mathrm{N} \to \mathrm{List}(\mathrm{N})$. Vi kan nu skapa en lista l som inte finns med i uppräkningen på följande sätt: Vi ser till att l skiljer sej från listan $\mathsf{f(0)}$ genom att låta ls första element vara $\mathsf{f(0)}$s första element plus ett. Om $\mathsf{f(0)}$ inte har något första element låter vi ls första element vara $\mathsf{0}$. Vi fortsätter på samma sätt genom att låta ls element nummer i vara strikt större än $\mathsf{f(i)}$s element nummer i om elementet $\mathsf{f(i)}$ existerar. Annars låter vi ls element nummer i vara $\mathsf{0}$.

   Vi ser nu att listan l inte kan finnas med i uppräkningen. Om den funnes skulle $\mathsf{l} = \mathsf{f(j)}$ för något j. Men l är konstruerad så att ls j:te element skiljer sig från $\mathsf{f(j)}$s j:te element. Vi har fått en motsägelse och alltså måste antagandet att List(N) är uppräkneligt vara felaktigt. (20)

   Svar: Problemet är att l inte blir en lista. Den blir oändligt lång och därmed inte induktivt genererad. Kom ihåg att ett element i en induktivt definierad mängd alltid kan fås genom att applicera klausulerna ett ändligt antal gånger.

3. Ge ett exempel på ett öppet λ-uttryck (dvs ett uttryck som har en fri variabel) med sluten normalform. Visa också hur uttrycket normaliseras. (20)

   Svar: I uttrycket $(\lambda x.\lambda y.y\ z)$ finns variabeln z fri. Uttrycket är ett β-redex och om vi reducerar får vi uttrycket $\lambda y.y$ som är slutet.

4. Beskriv språket PRF, mängden av de primitivt rekursiva funktionerna. Ge den abstrakta syntaxen, ge en informell beskrivning av semantiken för de olika konstruktionerna i språket. Ge ett exempel (bevis ej nödvändigt) på ett program som ej kan uttryckas i språket. (30)

   Svar: Se föreläsningsanteckningarna eller läroboken (sid 23). Ett exempel på ett program som inte kan uttryckas i språket är ett program som inte terminerar.

5. Följande uppgift består i att med hjälp av ändliga tillståndsautomater implementera en syntax-kontroll och en additionsfunktion av tal i unär representation.

   Ge först en ändlig automat som accepterar det reguljära uttrycket $0^*1^*01^*$. Man kan se en sträng i detta språk som två tal i unär representation, talparet $(2, 3)$ kodas t.ex. av strängen 0110111. Ge nu en ändlig tillståndsmaskin som implementerar additionsfunktionen! Dvs indatasträngen skall tas från språket ovan och resultatet skall vara en sträng som kodar summan av talen i indatasträngen. Det är inte nödvändigt att bevisa att implementeringen är riktig. (20)

   Svar: Jag har inte lyckats göra en maskinläsbar bild av automaten!

6. Skriv ett program i χ som ej terminerar. Visa varför! (20)

   Svar: Programmet **rec** $x = x$ **end** terminerar ej. Programmet beräknas i ett steg till sig själv (vi skall byta ut alla fria förekomster av variabeln x i uttrycket x mot **rec** $x = x$ **end**).

7. Skriv ett program i λ-kalkyl som inte har någon normalform. Visa varför! (20)

   Svar: Se boken sid 163.

Lycka till!

Bengt

P.S. Jag kan själv inte komma och svara på frågor, i stället kommer Peter Dybjer att göra det. Han är ansvarig för motsvarande kurs på Chalmers.