DD1351 - DD1350 Logik för dataloger

TENTAMEN
9 januari 2019, 8.00 - 13.00

Johan Karlander
KTH EECS

Skriv varje problem på ett separat blad. Skriv ditt namn på alla blad. Flera formelblad är bifogade; inga andra hjälpmedel är tillåtna.

Del E

Om du fick minst betyg E på kontrollskrivningen 2018 kan du hoppa över första uppgiften, som då redan är godkänd. För att uppnå betyg E krävs 16 poäng på E-delen, inklusive tillgodoräknade bonuspoäng från hemtal och labbar.

1. Bevisa sekventen $\neg q \to s,\ s \to p \ \vdash\ p \vee q$ med naturlig deduktion. Rita tydligt alla boxar för att visa räckvidden för alla antaganden i beviset. [4]

2. Vi låter $V(x, y)$ betyda att x är vän till y och $F(x, y)$ betyda att x är fiende till y. Vi antar att V och F är definierade så att $V(x, x)$ och $F(x, x)$ aldrig är sanna. Dessutom kan vi anta att V och F är definierade så att $V(x, y)$ är sann om och endast om $V(y, x)$ är sann och motsvarande för F (symmetri). [5]
   Vi har påståendet A: "Alla som har minst en vän har också minst en fiende" Betrakta de två formlerna F1: $\forall x\,(\exists y\,V(x, y) \to \exists z\,F(x, z))$ och F2: $\forall x\,\exists y\,(V(x, y) \wedge \exists z\,F(x, z))$.
   Visa med exempel att F1 och F2 inte är ekvivalenta. Ange med utförlig motivering vilken av F1 och F2 som är en korrekt översättning av påståendet A.

3. Bevisa med valfri metod vilken eller vilka (om någon) av följande logiska konsekvenser som gäller: [6]

   (a) $(p \vee q) \wedge r \models p \vee (q \wedge r)$
   (b) $p \wedge (q \vee r) \models p \wedge q$
   (c) $\neg\,(p \to (q \to r)) \models p$

4. Låt $Atoms \stackrel{\text{def}}{=} \{p, q\}$ och låt $\mathcal{M}$ vara en temporallogisk modell definierad som: [5]

$$\begin{array}{rcl} S & \stackrel{\text{def}}{=} & \{s_1, s_2, s_3, s_4\} \\ \to & \stackrel{\text{def}}{=} & \{(s_1, s_2), (s_2, s_3), (s_3, s_1), (s_3, s_4), (s_4, s_3), (s_4, s_4)\} \\ L & : & s_1 \mapsto \{p\},\ s_2 \mapsto \emptyset, s_3 \mapsto \{p, q\}, s_4 \mapsto \{q\} \end{array}$$

   För vilka tillstånd $s_i : s_1, s_2, s_3, s_4$ gäller $\mathcal{M}, s_i \models \mathsf{AF}\ \mathsf{AG}\ p$? För vilka tillstånd gäller $\mathcal{M}, s_i \models \mathsf{EF}\ \mathsf{EG}\ q$? Motivera dina svar med hänvisning till CTLs formella semantik.

5. Betrakta programmet Max nedan: 4

```
if (Y > X) {
  X = Y;
}
if (Z > X) {
  X= Z;
}
```

Programmet är tänkt att beräkna det största av tre tal eller mer exakt ett tal som inte är mindre än något av talen. Exakt hur gör programmet det? Ange lämpliga förvillkor (precond) och eftervillkor (postcond). Dessa villkor måste uttryckas med rätt syntax. Det är tillåtet att använda $>, =$ och $\geq$ t.ex. men inte Max som inte är definierat. Du behöver inte bevisa att programmet är korrekt (men det kan kanske ändå vara en hjälp för dig att prova att göra det eftersom du då får en kontroll av att villkoren är korrekt valda).

Del C

Om du fick betyg C på kontrollskrivningen 2018 kan du hoppa över första uppgiften, som då redan är godkänd. För att uppnå betyg C krävs 12 poäng, och för betyg D krävs 8 poäng på C-delen. I bägge fallen krävs även godkänt på E-delen.

6. Använd naturlig deduktion för att bevisa sekventen 5

$$\forall x\,(P(x) \to \exists y\, G(x,y)) \;\vdash\; \forall x\,\forall y\,\neg G(x,y) \to \forall x\,\neg P(x)$$

Rita tydligt alla boxar för att visa räckvidden för alla antaganden och nya variabler i beviset.

7. Version DD1351: De studenter som är registrerade på DD1351 ska göra denna uppgift. 5

I prolog kan man uttrycka ett program som delar upp en lista i två delar genom att använda ett predikat app som är definierat så här:

$app([], L, L)$

$app([H|T], L, [H|R]) :- app(T, L, R)$

anropet $app(L1, L2, [1,2,3])$ kan lyckas på flera sätt

1. $L1 = [], L2 = [1,2,3]$

2. $L1 = [1], L2 = [2,3]$

3. $L1 = [1,2], L2 = [3]$

4. $L1 = [1,2,3], L2 = []$

Nu vill vi på liknande sätt använda app i en definition för att dela upp en lista i tre delar.

Vi skriver:

$app3(L1, L2, L3, L) :- app(L1, L2, Lt), app(Lt, L3, L)$

Vi provar med anropet $app3([A|L1], [B|L2], [C|L3], [1,2,3])$

Det fungerar inte riktigt, eftersom det hamnar i en loop när alla lösningar presenterats.

Varför loopar programmet?

Hur kan man ändra definitionen av app3 så att det inte loopar för något delbevis?

Det skall fortfarande kunna dela upp listan L i tre delar på alla möjliga sätt.

*****

Version DD1350: De studenter som är registrerade på DD1350 ska göra denna uppgift.

Låt L vara en heltalslista och låt Halv vara operationen som bildar en ny lista genom att välja ut 1:a elementet ur L och sedan forsätta välja vartannat element ur L. Mer exakt betyder $Halv(L_2, L_1) = TRUE$ att L_2 är L_1 efter ovanstående operation. Till exempel är $Halv([1,7,3,1],[1,9,7,1,3,2,1])$ sann och $Halv([2,3,5],[2,5,4,2,6])$ falsk.

Definiera $Halv$ rekursivt.

8. Fråga: Vilket alternativ är rätt svar på denna fråga? 4

   (a) Alla nedanstående alternativ är rätt.
   (b) Inga nedanstående alternativ är rätt.
   (c) Alla ovanstående alternativ är rätt.
   (d) Minst ett av ovanstående alternativ är rätt.
   (e) Inga ovanstående alternativ är rätt.
   (f) Inga ovanstående alternativ är rätt.

   Vi inför 6 satslogiska variabler a, b, c, d, e, f där $a = T$ betyder att påståendet (a) är sant o.s.v. Varje påstående kan skrivas om till ett ekvivalent påstående. Låt $p \leftrightarrow q$ stå för $(p \rightarrow q) \wedge (q \rightarrow p)$ (d.v.s. p och q är ekvivalenta). Vi kan då se att för påståendet a gäller $a \leftrightarrow b \wedge c \wedge d \wedge e \wedge f$.

   Skriv om alla påståenden på detta sätt. Använd valfri metod för att avgöra vilka av påståendena som är sanna.

9. Vi studerar binära träd rekursivt definierade (som i kursen). Vi säger att ett träd T_1 är ett delträd till T_2 om det finns en nod x i T_2 sådan att trädet med i T_2 med x som nod är T_1. 4

   (a) Låt $DT(T_1, T_2)$ vara ovanstående relation. Visa hur man kan definierar den *rekursivt*.
   (b) Vi kan förstås se utan problem att om $n(T)$ är antalet noder i ett träd T och om $DT(T_1, T_2)$ är sant så måste $n(T_1) \leq n(T_2)$. Definiera $n(T)$ och använd strukturell induktion för att visa att $n(T_1) \leq n(T_2)$.

Del A

För att uppnå betyg A krävs 12 poäng, och för betyg B krävs 8 poäng på A-delen. I bägge fallen krävs även betyg C på C-delen.

10. Programmet `Div` är tänkt att utföra en heltalsdivision av X med Y. [9]

```
K=0;
R=X;
while (R >= Y) {
  R = R - Y;
  K = K + 1;
}
```

Ange lämpliga för- och eftervillkor och bevisa med formell metod att programmet är korrekt. Ange alla *bevisförpliktelser* med ett X.

11. Formelbladet som är bifogat ger de 12 grundläggande reglerna för naturlig deduktion i predikatlogik. De är $\wedge_i$, $\wedge_{e1}$, $\wedge_{e2}$, $\vee_{i1}$, $\vee_{i2}$, $\vee_e$, $\rightarrow_i$, $\rightarrow_e$, $\neg_i$, $\neg_e$, $\bot_e$, $\neg\neg_e$. De övriga reglerna som står angivna kan *härledas* från de första. [9]

    (a) Regeln LEM går att härleda ur de övriga reglerna. Visa hur det går till.

    (b) Med hjälp av LEM och de övriga reglerna, utom $\neg_i$, kan vi faktiskt härleda $\neg_i$. Närmare bestämt är det så att om vi utan att använda $\neg_i$ kan härleda $\bot$ från antagandet att ϕ gäller så kan vi härleda $\neg\phi$ (fortfarande utan att använda $\neg_i$). Visa hur.