DD1351 - DD1350 Logik för dataloger

TENTAMEN
17 april 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 $\neg p \to q,\ p \to r,\ \neg r \wedge t \ \vdash\ q$ med naturlig deduktion. Rita tydligt alla boxar för att visa räckvidden för alla antaganden i beviset. [4]

2. Vi betraktar de två formlerna $F_1 : \exists x\, \exists y\, (\neg P(x) \to Q(y))$ och $F_2 : \exists z\, (\neg P(z) \to Q(z))$ [6]

   (a) Gäller $F_1 \models F_2$?
   (b) Gäller $F_2 \models F_1$?

   I båda fallen ska du antigen ge ett bevis i naturlig deduktion eller ge ett motexempel.

3. Vi har de fyra formlerna $p \vee q$, $\neg q \vee r$, $r \to q \wedge \neg r$, $q \vee r \to p$. Vi säger att mängden av formler är satisfierbar om det finns minst en uppsättning värden på de tre variablerna som gör de fyra formlerna sanna *samtidigt*. Avgör med hjälp av sanningsvärdestabeller om mängden är satisfierbar eller inte. [5]

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

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

   För vilka tillstånd $s_i : s_1, s_2, s_3, s_4$ gäller $\mathcal{M}, s_i \models \mathsf{AG\ EF}\ p \vee q$? För vilka tillstånd gäller $\mathcal{M}, s_i \models \mathsf{EG\ AF}\ q \vee r$? Motivera dina svar utförligt.

5. Betrakta följande bevis: [4]

1	$y = g(z, y)$	Premiss
2	$y = z$	Premiss
3	$y = g(z, z)$	$=_e$ 2,1

   För att komma till rad 3 har vi använt oss av regeln =e, som beskrivs på formelbladet. I regeln omnämns en formel ϕ. Förklara hur denna formel ϕ ser ut i exemplet ovan.

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]

$$\exists\, x\, \exists\, y\, F(x,y),\ \forall\, x\, \neg F(x,x)\ \vdash\ \forall\, x \exists\, y\, \neg(x = y)$$

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.**

Uttrycket app som definieras här, kan användas såväll för att lägga ihop två listor till en tredje som för att dela upp en lista i två.

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

Ange en fråga (query) i Prolog som delar upp en lista i två delar på ett indeterministiskt sätt? (1p)

Vi vill generalisera detta predikat. Först definierar vi tvÂ predikar app3a och app3b som med användning av app lägger samman tre listor respektive delar upp en lista i tre delar.

$app3a(L1, L2, L3, L) :- app(L1, R, L), app(L2, L3, R).$

$app3b(L1, L2, L3, L) :- app(L2, L3, R), app(L1, R, L).$

Vad är skillnaden i beteende på de två predikaten app3a och app3b? (2p)

Vi vill generalisera predikatet ytterligare. Nu definierar vi ett predikat appn som med användning av app lägger samman ett antal listor till en lista respektive delar upp en lista i dellistor.

$appn([], []).\ appn([H|T], L) :- H = [\_|\_], app(H, R, L), appn(T, R).$

$appm([], []).\ appm([H], H) :- H = [\_|\_].$
$appm([H, G \mid T], L) :- H = [\_|\_], G = [\_|\_], appm([R \mid T], L), app(H, G, R).$

Vad är effekten av att skriva $H = [\_ \mid \_]$? (1p)

Vad är skillnaden i beteende på de två predikaten appn och appm? (2p)

Version DD1350: **De studenter som är registrerade på DD1350 ska göra denna uppgift.** [5]

Vi arbetar med listor och vill definiera ett predikat Riff(L_1, L_2, L_3) som är sant om och endast om L_3 är en blandning av listorna L_1 och L_2 på följande sätt: Om ingen av listorna är tomma så inleds L_3 med 1:a elementet i L_1 och därefter kommer 1:a elementet i L_2 och därefter kommer omväxlande varannat element fråm vardera listan ända tills vi när slutet på någon lista. Då läggs resten av den andra listan till på slutet.

För att se ett exempel kan vi ta $L_1 = [3, 7, 5]$, $L_2 = [6, 2, 8, 4, 6]$. Då är $L_3 = [3, 6, 7, 2, 5, 8, 4, 6]$.

(a) Definiera predikatet Riff rekursivt.

(b) Vi kan se att det är uppenbart att om Riff(L_1, L_2, L_3) är sant så måste $length(L_3) = length(L_1) + length(L_2)$. Bevisa detta med strukturell induktion.

8. Låt oss anta att vi har n satslogiska variabler $p_1, p_2, \cdots p_n$. Låt oss anta att Γ är en mängd med satslogisk formler på formen $p_i \to p_j$ där vi har $i \neq j$. [4]

   (a) Finns det någon mängd Γ som inte är satisfierbar eller är Γ satisfierbar för alla val av Γ?

   (b) Nu tar vi en formel $p_1 \to \neg p_i$ där $i \neq 1$ och lägger till den så att vi har mängden $\{p_1 \to \neg p_i\} \cup \Gamma$. Är denna mängd alltid satisfierbar oberoende av vad Γ är?

   (c) Nu väljer vi dessutom en formel $\neg p_1 \to p_j$ där $j \neq 1, i$. Är mängden $\{p_1 \to \neg p_i, \neg p_1 \to p_j\} \cup \Gamma$ alltid satisfierbar oberoende av vad Γ är?

   Motivera dina svar utförligt genom exempel och resonemang kring möjliga valueringar.

9. I den här uppgiften arbetar vi med en ny typ av kvantifierare $\exists_x^F$. Vi definierar den så att $\exists_x^F \phi(x)$ är sann i en modell M om och endast om $\phi(a)$ är sann för *ändligt* många och inte fler än ändligt många element a i M:s universum U. ($\exists^F$ är definierad just för denna uppgift.) [4]

   Ett exempel är om vi låter M vara det vanliga universumet för envariabelkalkyl över de reella talen och $p(x)$ ett polynom av udda grad. Då gäller $\exists_x^F (p(x) = 0)$.

   Det går att se att om $\exists_x^F \phi(x)$ är sann i en modell M så är också $\exists x\, \phi(x)$ sann i modellen men att det omvända inte alltid gäller. Det går ändå att säga vissa saker om hur den nya kvantifieraren fungerar.

   Avgör med hjälp av resonemang kring möjliga modeller vilka (om någon) av följande konsekvenser som är giltiga:

   (a) $\exists_x^F \exists_y^F (A(x) \wedge B(y)) \models \exists_z^F (A(z) \wedge B(z))$

   (b) $\exists_z^F (A(z) \wedge B(z)) \models \exists_x^F \exists_y^F (A(x) \wedge B(y))$

   (c) $\exists_x^F \forall y\, G(x, y) \models \forall y \exists_x^F G(x, y)$

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. Om vi har en satslogisk formel ϕ så kan vi låta $kon(\phi)$ beteckna antalet konnektiv ($\vee$, $\wedge$, $\to$, $\neg$) som förekommer i ϕ vi kan förutsätta att det inte finns några dubbla negationer ($\neg\neg$) någonstans i formeln. Vi skulle vilja relatera $kon(\phi)$ till antalet variabler som förekommer i formeln. För att förenkla resonemangen lite kan vi förutsätta att de formler vi studerar är *enkla* i den meningen att ingen variabel förekommer *mer än en gång* i formeln. Om ϕ är en sådan formel låter vi $var(\phi)$ vara antalet variabler som förekommer i formeln. [9]

    (a) Använd strukturell induktion över formlernas uppbyggnad för att visa att $kon(\phi) \geq var(\phi) - 1$.

    (b) Använd strukturell induktion över formlernas uppbyggnad för att visa att $kon(\phi) \leq 4\, var(\phi) - 3$.

11. Programmet `Tre` är tänkt att avgöra om ett tal är delbart med 3 eller inte. 9

```
while (U >= 3){
  U = U-3;
}
if (U = 0) {
  R = 1;
} else {
  R = 0;
}
```

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