Karlsruher Institut für Technologie

Klausur Formale Systeme

Fakultät für Informatik

SS 2019

Prof. Dr. Bernhard Beckert

31. Juli 2019

Name: _____

Vorname: _____

Matrikel-Nr.: _____

Die Bearbeitungszeit beträgt 60 Minuten.

A1 (14)	A2 (8)	A3 (6)	A4 (8)	A5 (10)	A6 (7)	A7 (7)	Σ (60)

Bewertungstabelle bitte frei lassen!

Gesamtpunkte:

1 Zur Einstimmung (4+5+5 = 14 Punkte)

a. Seien p ein einstelliges Prädikatensymbol und f ein einstelliges Funktionssymbol.

Geben Sie für folgende prädigatenlogische Formeln – **falls möglich** – Interpretationen I über dem Universum $D = \{a, b\}$ an, und zwar jeweils

- eine Interpretation, in der die Formel **wahr** ist, und
- eine Interpretation, in der die Formel **falsch** ist.

In den Fällen, in denen eine Interpretation mit der gesuchten Eigenschaft **nicht existiert**, geben Sie dies an.

Hinweis: Es muss explizit angegeben werden, wenn eine passende Interpretation nicht existiert (schreiben Sie „existiert nicht“ neben den Kasten). Es genügt nicht, die Beschreibung der Interpretation leer zu lassen.

i. $(\forall x\, p(x)) \to (\forall y\, p(f(y)))$

Interpretation, in der die Formel wahr ist:

$I(p)(a) = \mathbf{F}$	$I(p)(b) = \mathbf{W}$	$I(f)(a) = a$	$I(f)(b) = b$

Interpretation, in der die Formel falsch ist:

$I(p)(a) =$	$I(p)(b) =$	$I(f)(a) =$	$I(f)(b) =$	**Nicht möglich!**

ii. $(\forall x\, p(f(x))) \to (\forall y\, p(y))$

Interpretation, in der die Formel wahr ist:

$I(p)(a) = \mathbf{W}$	$I(p)(b) = \mathbf{W}$	$I(f)(a) = a$	$I(f)(b) = b$

Interpretation, in der die Formel falsch ist:

$I(p)(a) = \mathbf{W}$	$I(p)(b) = \mathbf{F}$	$I(f)(a) = a$	$I(f)(b) = a$

Fortsetzung 1 Zur Einstimmung

b. Geben Sie kurze Antworten zu folgenden Fragen bzw. Aufgaben:

i. Geben Sie eine aussagenlogische Normalform an, für die man die Erfüllbarkeit von Formeln in Polynomialzeit entscheiden kann.

Horn, 2-KNF, DNF, Shannon-Normalform

ii. Geben Sie den Term an, der durch die Unifikation der Terme $f(x, g(d))$ und $f(g(c), y)$ entsteht. Dabei sind x, y Variablensymbole und c, d, f, g Funktionssymbole.

$f(g(c), g(d))$

iii. Geben Sie eine modallogische Formel an, die genau in **reflexiven** Kripkerahmen allgemeingültig ist.

$\Box P \to P$

iv. Wann während des Programmablaufs muss eine Schleifeninvariante wahr sein?

Vor der Schleife und nach jeder Iteration.

v. Die logischen Operatoren $\mathbf{U}, \mathbf{X}, \mathbf{1}, \neg, \wedge$ bilden eine LTL-Basis. Geben Sie eine LTL-Formel an, die zu der LTL-Formel $\Diamond A$ äquivalent ist und nur Operatoren aus dieser Basis enthält.

$\mathbf{1}\ \mathbf{U}\ A$

c. Zeigen Sie mit Hilfe des **Resolutionskalküls** der Aussagenlogik, dass folgende Klauselmenge unerfüllbar ist. Notieren Sie bei jedem Schritt die Klauseln, auf die die Resolutionsregel angewandt wird.

(1) $\{\neg A, B\}$ (2) $\{A, D\}$ (3) $\{\neg B, D\}$ (4) $\{C, \neg D\}$ (5) $\{\neg C, \neg D\}$

- (6) $\{\neg D\}$ aus $4, 5$
- (7) $\{\neg B\}$ aus $3, 6$
- (8) $\{B, D\}$ aus $1, 2$
- (9) $\{D\}$ aus $7, 8$
- (10) $\square$ aus $6, 9$

2 Theorie (4+4 = 8 Punkte)

a. Angenommen das Halteproblem wäre entscheidbar. Begründen Sie, warum unter dieser Annahme auch die Erfüllbarkeit von Formeln der Prädikatenlogik erster Stufe entscheidbar wäre.

Erläutern Sie dazu kurz, wie sich das Entscheidungsproblem der Prädikatenlogik auf das Halteproblem reduzieren lässt.

Hinweis: Eine kurze Begründung genügt (wenige Sätze).

- Sei ϕ eine prädikatenlogische Formel, deren Erfüllbarkeit entschieden werden soll.
- Sei K ein Kalkül, mit dem die Unerfüllbarkeit prädikatenlogischer Formeln überprüft werden kann und der korrekt und vollständig ist (bspw. der Tableaukalkül).
- Schreibe ein Programm (bzw. konstruiere eine Turing-Maschine), die alle Ableitungen mit Regeln aus K, die mit ϕ beginnen aufzählt (im Tableaukalkül sind das alle Tableaus für ϕ), jeweils prüft, ob es sich um einen geschlossenen Beweis für die Unerfüllbarkeit von ϕ handelt, und anhält, wenn ein Beweis gefunden ist.
- Wenn das Halteproblem entscheidbar wäre, könnte man entscheiden, ob dieses Programm terminiert. Es terminiert genau dann, wenn es einen Beweis für die Unerfüllbarkeit von ϕ gibt (da alle Beweiskandidaten aufgezählt werden), den es genau dann gibt, wenn ϕ tatsächlich unerfüllbar ist (wegen der Korrektheit und Vollständigkeit des Kalküls).

b. Was besagt der Gödelsche Unvollständigkeitssatz?

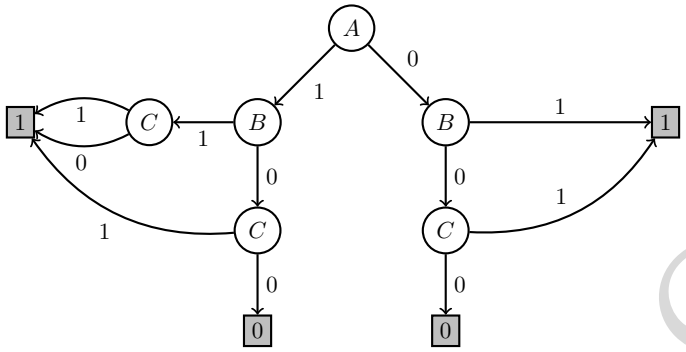
Die Theorie der natürlichen Zahlen (mit Multiplikation und Addition) ist nicht rekursiv aufzählbar / nicht (in PL1) axiomatisierbar.

ODER

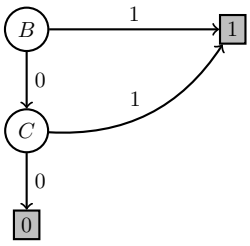
Jedes hinreichend mächtige, rekursiv aufzählbare formale System ist entweder widersprüchlich oder unvollständig

3 Shannongraphen (2+2+2 = 6 Punkte)

Gegeben sei der folgende Shannongraph $\mathcal{G}$, der eine aussagenlogische Formel F repräsentiert.



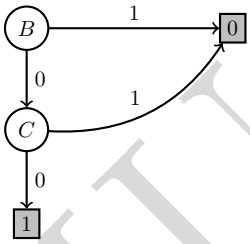
a. Geben Sie den reduzierten Shannongraphen zu $\mathcal{G}$ an (der ebenfalls F repräsentiert).



b. Geben Sie eine aussagenlogische Formel (ohne *sh*-Operatoren) an, die zu F äquivalent ist.

$\boxed{B \vee C}$

c. Zeichnen Sie einen Shannongraphen, der $\neg F$ repräsentiert. Der Shannongraph muss nicht unbedingt reduziert sein.



4 Formalisieren in PL1 (2+2+2+2 = 8 Punkte)

Gegeben sei die prädikatenlogische Signatur $\Sigma = (\{\mathit{divergent}, \mathit{equiv}\}, \{\mathit{comp}\}, \alpha)$. Sie enthält die Prädikatensymbole $\mathit{divergent}(\cdot)$ und $\mathit{equiv}(\cdot,\cdot)$, sowie das Funktionssymbol $\mathit{comp}(\cdot,\cdot)$.

Zur Auswertung der Formeln werden nur solche Interpretationen (D, I) über Σ verwendet, in denen

- das Universum D eine Menge von Computer-Programmen ist,
- das Prädikat $\mathit{div}(x)$ genau dann wahr ist, wenn das Programm x divergent ist,
- das Prädikat $\mathit{equiv}(x_1, x_2)$ genau dann wahr ist, wenn die Programme x_1 und x_2 zueinander äquivalent sind,
- die Funktion $\mathit{comp}(x_1, x_2)$ die Komposition der Programme x_1 und x_2 zurückliefert.

Geben Sie jeweils eine Formel der Prädikatenlogik mit Gleichheit über Σ an, die folgende Sachverhalte darstellt:

a. Jedes Programm ist äquivalent zu sich selbst.

$\boxed{\forall x\ \mathit{equiv}(x, x)}$

---

b. Zwei äquivalente Programme sind entweder beide divergent oder beide nicht divergent.

$\boxed{\forall x_1\ \forall x_2\ (\mathit{equiv}(x_1, x_2) \rightarrow (\mathit{div}(x_1) \leftrightarrow \mathit{div}(x_2)))}$

---

c. Die Komposition zweier Programme ist genau dann divergent, wenn mindestens eines der beiden Programme divergent ist.

$\boxed{\forall x_1\ \forall x_2\ (\mathit{div}(\mathit{comp}(x_1, x_2)) \leftrightarrow (\mathit{div}(x_1) \vee \mathit{div}(x_2)))}$

---

d. Die Komposition von zwei Programmen x_1 und x_2 ist äquivalent zu der Komposition beliebiger jeweils zu x_1 bzw. x_2 äquivalenter Programme.

$\boxed{\forall x_1\ \forall x_2\ \forall x_3\ \forall x_4\ ((\mathit{equiv}(x_1, x_3) \wedge \mathit{equiv}(x_2, x_4)) \rightarrow \mathit{equiv}(\mathit{comp}(x_1, x_2), \mathit{comp}(x_3, x_4)))}$

---

5 Tableaukalkül (10 Punkte)

Seien r und p binäre Prädikatensymbole.

Vervollständigen und schließen Sie den folgenden Tableau-Beweis.

Notieren Sie dabei:

- den Regeltyp $(\alpha, \beta, \gamma, \delta)$ und die Formel, auf die eine Regel angewendet wird,
- bei Abschlüssen die beiden Partner,
- sowie die schließende Substitution.

Sie können die folgenden Regeln vom Typ β für die Äquivalenz verwenden:

$1\ A \leftrightarrow B$	
$1\ A$	$0\ A$
$1\ B$	$0\ B$

$0\ A \leftrightarrow B$	
$1\ A$	$0\ A$
$0\ B$	$1\ B$

$1\ \exists x\, \exists y\, (p(x,y) \wedge \neg p(y,x))$ (1)

$1\ \forall x\, \forall y\, (r(x,y) \rightarrow p(y,x))$ (2)

$1\ \forall x\, \forall y\, (r(x,y) \leftrightarrow p(x,y))$ (3)

$1\ \exists y\, p(c,y) \wedge p(y,c)$ (4: δ(1))

$1\ p(c,d) \wedge p(d,c)$ (5: δ(4))

$1\ p(c,d)$ (6: α(5))

$1\ \neg p(d,c)$ (7: α(6))

$0\ p(d,c)$ (8: α(7))

$1\ \forall y\, r(X,y) \leftrightarrow p(X,y)$ (9: γ(3))

$1\ r(X,Y) \leftrightarrow p(X,Y)$ (10: γ(9))

$1\ r(X,Y)$ (11: β(10))	$0\ r(X,Y)$ (13: β(10))
$1\ p(X,Y)$ (12: β(10))	$0\ p(X,Y)$ (14: β(10))
	$\star(5,14)$

$1\ \forall y\, r(X_2,y) \rightarrow p(y,X_2)$ (15: γ(2))

$1\ r(X_2,Y_2) \rightarrow p(Y_2,X_2)$ (16: γ(15))

$0\ r(X_2,Y_2)$ (17: β(16))	$1\ p(Y_2,X_2)$ (18: β(16))
$\star(11,17)$	$\star(8,18)$

Schließende Substitution: $\sigma = \{X/c, Y/d, X_2/c, Y_2/d\}$

6 Spezifikation mit der Java Modeling Language

(4+3 = 7 Punkte)

a. Geben Sie die Bedeutung der Nachbedingung (**ensures-Klauseln**) in dem folgenden JML-Methodenvertrag in natürlicher Sprache wieder.

```
public class A {
    public int[] t;

    /*@  requires 0 <= p && p < t.length;
      @  ensures (\forall int i; 0 <= i && i < p;
      @              t[i] == \old(t[t.length - p + i]));
      @  ensures (\forall int i; p <= i && i < t.length;
      @              t[i] == \old(t[i - p]));
      @*/
    public void m(int p) { ... }
}
```

Wenn `m` aufgerufen wird, terminiert `m` und und es gilt nach `m`'s Ausführung:

- Das Array `t` enthält dieselben Einträge wie im Vorzustand, jedoch um `p` Stellen nach rechts verschoben.
- Die letzten `p` Einträge von `t` stehen in der Reihenfolge des Vorzustands an den ersten `p` Stellen.

Fortsetzung 6 Spezifikation mit der Java Modeling Language

b. Sei eine Methode `f` der Klasse `A` durch die untenstehende Implementierung gegeben. Der untenstehende Vertrag spezifiziert, dass die Methode `f` die Summe der Elemente aus dem Array `a`, die ohne Rest durch `b` teilbar sind, zurückgibt (Wenn keines der Elemente ohne Rest durch `b` teilbar ist, so wird der Wert `0` zurückgegeben.).

Die untenstehende Schleifeninvariante ist unvollständig. Ergänzen Sie diese, sodass daraus eine korrekte Invariante entsteht, mit der die Nachbedingung des Methodenvertrags bewiesen werden kann.

```
public class A {

   /*@ public normal_behaviour
     @  requires 1 < b && a != null;
     @  requires (\forall int i; 0 <= i && i < a.length; 0 < a[i]);
     @  assignable \nothing;
     @  ensures \result ==
     @      (\sum int i; 0 <= i && i < a.length; a[i] % b == 0 ? a[i] : 0);
     @*/
   public int f(int[] a, int b) {
     int s = 0;
     /*@ loop_invariant
       @
       @
       @
       @
       @ assignable \nothing;
       @ decreases a.length - i;
       @*/
     for (int i = 0; i < a.length; i++) {
       int v = a[i];
       if (v % b == 0) s += v;
     }
     return s;
   }
}
```

```
/*@ loop_invariant 0 <= i && i <= a.length
  @    && s == (\sum int j; 0 <= j && j < i;
  @                a[j] % b == 0 ? a[j] : 0);
  @*/
```

7 Lineare Temporale Logik (LTL)

(2+5 = 7 Punkte)

a. Seien P und Q LTL-Formeln. Dann ist die Semantik von $P \,\mathbf{B}\, Q$ ("P begins Q") folgendermaßen definiert:

$$\xi \models P \,\mathbf{B}\, Q \iff \begin{array}{l}\text{Für jedes } n \in \mathbb{N}, \text{ für das } \xi_n \models P \text{ gilt,} \\ \text{gilt für jedes } k \geq n \text{ die Aussage } \xi_k \models Q\end{array}$$

Geben Sie einen zu $P \,\mathbf{B}\, Q$ äquivalenten LTL-Ausdruck an, der den Operator **B** nicht verwendet.

$\Box(P \rightarrow \Box Q)$

b. Zeigen Sie die Allgemeingültigkeit folgender LTL-Formel:

$$\big(\Diamond(p \wedge \Diamond q)\big) \rightarrow \big((\Diamond q) \,\mathbf{U}\, p\big) \ .$$

Hinweis: Um Allgemeingültigkeit zu sein, muss die Formel in allen ω-Strukturen ξ gelten.

Die Aussage ist allgemeingültig.

Aus $(\Diamond(p \wedge \Diamond q)$ folgt, dass zwei Zeitpunkte existieren, für die $t_0 \leq t_1$ und $\xi, t_0 \models p$ sowie $\xi, t_1 \models q$ gelten. Dies erfüllt $(\Diamond q) \,\mathbf{U}\, p$, denn $(\Diamond q) \,\mathbf{U}\, p$ besagt, dass für jeden Zeitpunkt $s_0 \leq t_0$ (mit $\xi, t_0 \models p$ n. V.), ein Zeitpunkt s_{s_0} existiert, in dem q gilt. Nach Voraussetzung ist dies $s_{s_0} = t_1$.