Klausur Formale Systeme

Fakultät für Informatik

SS 2020

Prof. Dr. Bernhard Beckert

31. Juli 2020

Name: _____

Vorname: _____

Matrikel-Nr.: _____

Die Bearbeitungszeit beträgt 60 Minuten.

A1 (11)	A2 (10)	A3 (6)	A4 (8)	A5 (8)	A6 (9)	A7 (8)	Σ (60)

Bewertungstabelle bitte frei lassen!

Gesamtpunkte:

1 Zur Einstimmung (4 + 4 + 3 = 11 Punkte)

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

i. Ergänzen Sie: "Ein Kalkül heißt vollständig, wenn ..."

ii. Was ist eine Substitution?

iii. Nennen Sie zwei Theorien, die ein moderner SMT-Solver entscheiden kann.

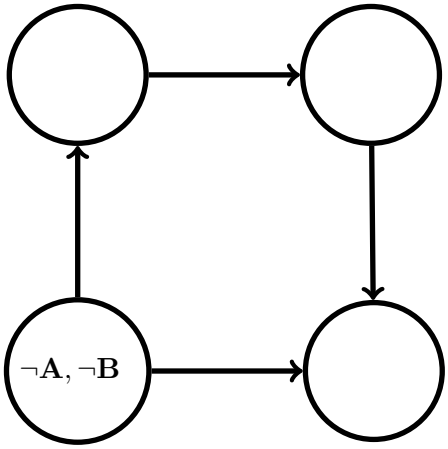
iv. Wann, per Definition, akzeptiert ein Büchi-Automat $\mathcal{A} = (S, V, s_0, \delta, F)$ ein unendliches Wort $w \in V^\omega$?

Fortsetzung 1 Zur Einstimmung

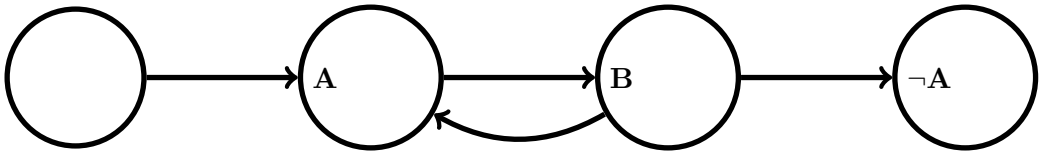
b. Gegeben seien die unten dargestellten Kripkestrukturen mit den teilweise festgelegten Variablenbelegungen über der Signatur $\Sigma = \{A, B\}$.

Ergänzen Sie den Wahrheitswert für jede aussagenlogische Variable in jeder Welt, sodass die angegebene modallogische Formel in allen Welten der Kripkestruktur erfüllt ist.

i. $(A \to \Box\Diamond B) \wedge (\neg B \to \Diamond A)$

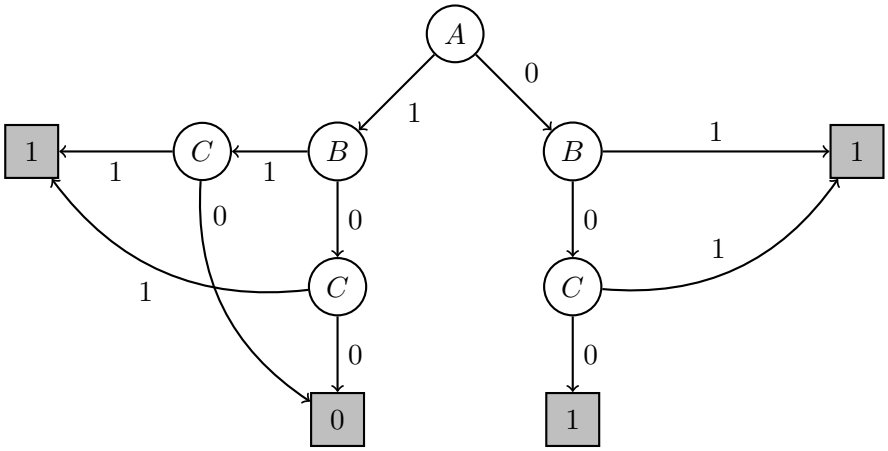


ii. $(\Box A \to B) \wedge (\Diamond B \to \Diamond\Diamond A)$



Fortsetzung 1 Zur Einstimmung

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



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

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

2 Modallogik mit Quantoren (4 + 2 + 4 = 10 Punkte)

Man kann Modallogik und Prädikatenlogik kombinieren, indem man Quantoren, Prädikate und Funktionen zur Modallogik hinzunimmt. Dann hat jede mögliche Welt in einer Kripkestruktur ein prädikatenlogisches Modell (und nicht mehr nur eine aussagenlogische Belegung).

Dabei stellt sich die Frage, ob die Modelle in den verschiedenen möglichen Welten einer Struktur alle das gleiche Universum haben müssen, oder ob man unterschiedliche Universen zulässt. Diese Frage ist nicht leicht zu beantworten; darüber streiten Philosophen und Logiker seit Jahrzehnten.

a. Diskutieren Sie anhand der Formel

$$\forall x \, \Box \, \exists y \, (x \doteq y)$$

den Unterschied der beiden Möglichkeiten (unterschiedliche Universen erlaubt oder nicht):

- Erläutern Sie, warum die obige Formel allgemeingültig ist, wenn alle Universen einer Struktur gleich sein müssen.

- Zeigen Sie, dass die Formel *nicht* allgemeingültig ist, wenn man unterschiedliche Universen zulässt, indem Sie eine Kripkestruktur angeben, die ein Gegenbeispiel ist.
  Hinweis: Ein Gegenbeispiel kann mit zwei Welten konstruiert werden.

Fortsetzung 2 Modallogik mit Quantoren

b. Ein Spezialfall von Modallogik mit Quantoren ist Lineare Temporallogik (LTL) mit Quantoren. Formalisieren Sie in LTL mit Quantoren

> Es wird in Zukunft (mindestens) einen Mensch auf dem Mars geben.

Verwenden Sie dazu die Prädikate

- $istMensch(\cdot)$
- $auf(\cdot, \cdot)$

und die Konstante

- $mars$

---

c. Nehmen wir an, man würde für LTL mit Quantoren unterschiedliche Universen zulassen, wobei die Universen genau diejenigen Menschen enthalten, die zu dem jeweiligen Zeitpunkt leben.

Warum ist es dann schwierig, Sätze wie

> In Zukunft wird es einen Mensch auf dem Mars geben,
> dessen Vorfahren schon heute leben.

sinnvoll zu formalisieren, wenn man dafür ein Prädikat $vorfahr(\cdot, \cdot)$ verwenden will?

---

3 Markierungsalgorithmus für Hornformeln (2 + 2 + 2 = 6 Punkte)

Überprüfen Sie jeweils die folgenden Hornformeln auf Erfüllbarkeit. Benutzen Sie den in der Vorlesung vorgestellten Markierungsalgorithmus. **Unterstreichen** Sie dazu die zu markierenden Literale in der Formel **und** geben Sie unter Schritt n an, **welche(s) Literal(e) im n-ten Schritt markiert** wurde(n). Geben Sie **zudem** ein **Modell** an **oder** benennen Sie die **Hornformel**, aufgrund derer der Algorithmus mit „unerfüllbar“ abbricht!

Hinweis: Bei einem Modell ist für jedes Atom explizit anzugeben, ob es zu wahr oder falsch auswertet.

a. $(P_1 \wedge P_2 \wedge P_3 \rightarrow P_2) \wedge P_2 \wedge (P_2 \rightarrow P_3)$

Schritt 1:	Schritt 2:	Schritt 3:

Ergebnis:

b. $(\neg P_1 \vee \neg P_2 \vee \neg P_3) \wedge (\neg P_2 \vee \neg P_4) \wedge (\neg P_1 \vee \neg P_3)$

Schritt 1:	Schritt 2:	Schritt 3:	Schritt 4:

Ergebnis:

c. $P_1 \wedge (P_2 \wedge P_1 \rightarrow P_3) \wedge P_2 \wedge (P_3 \rightarrow \mathbf{0})$

Schritt 1:	Schritt 2:	Schritt 3:

Ergebnis:

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

Gegeben sei die prädikatenlogische Signatur $\Sigma = (\{tm, lang, decidable, accepts\}, \alpha)$. Sie enthält die Prädikatensymbole $tm(\cdot)$, $lang(\cdot)$, $decidable(\cdot)$ und $accepts(\cdot, \cdot)$.

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

- das Universum D die Menge aller Sprachen und aller Turingmaschinen ist,
- das Prädikat $tm(x)$ genau dann wahr ist, wenn x eine Turingmaschine ist,
- das Prädikat $lang(x)$ genau dann wahr ist, wenn x eine Sprache ist,
- das Prädikat $decidable(x)$ genau dann wahr ist, wenn x entscheidbar ist,
- das Prädikat $accepts(x, y)$ genau dann wahr ist, wenn y von x akzeptiert wird.

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

a. Turingmaschinen sind keine Sprachen.

---

b. Nicht jede Sprache wird von einer Turingmaschine akzeptiert.

---

c. Wenn eine Sprache entscheidbar ist, dann wird sie von einer Turingmaschine akzeptiert.

---

d. Jede Turingmaschine akzeptiert genau eine Sprache.

---

5 Resolution (8 Punkte)

Beweisen Sie mit Hilfe des Resolutionskalküls:

$$\left\{ \begin{array}{l} \forall x \forall y \big(\neg p(x,y) \vee p(x, f(y)) \big), \\ \forall x \forall y \exists z (p(f(z), y) \vee \neg p(x, f(y))) \end{array} \right\} \vdash \forall x (p(x,x) \rightarrow \exists z \; p(f(z), f(x))) \tag{1}$$

a. Überführen Sie (1) in eine Klauselmenge, auf die Sie den Resolutionskalkül anwenden können.

Geben Sie an, welche neuen Funktionssymbole durch Skolemisierung entstanden sind.

Hinweise: Sie können Zwischenschritte angeben, müssen es aber nicht. Die beiden Voraussetzungen sollten je in eine Klausel münden und durch die Konklusion sollten ebenfalls zwei Klauseln entstehen.

b. Geben Sie einen Resolutionsbeweis an, der aus den Klauseln aus Teilaufgabe **a** die leere Klausel ableitet. Notieren Sie bei jedem Schritt die Klauseln, auf die die Resolutionsregel angewandt wurde, sowie **die verwendete Substitution**.

6 Spezifikation mit der Java Modeling Language (5 + 4 = 9 Punkte)

a. Geben Sie die Bedeutung des folgenden JML-Methodenvertrags in natürlicher Sprache wieder.

```
public class A {

    /*@ public normal_behaviour
      @  requires 0 < q && q <= p;
      @  assignable \nothing;
      @  ensures \result.length == q;
      @  ensures (\forall int i; 0 <= i && i < q; 0 <= \result[i]);
      @  ensures (\sum int i; 0 <= i && i < q; \result[i]) <= p;
      @  ensures (\forall int i; 0 <= i && i < q;
      @      (\sum int j; 0 <= j && j < q; \result[j]) <= \result[i] * q);
      @*/
    public int[] m(int p, int q) { ... }
}
```

Fortsetzung 6 Spezifikation mit der Java Modeling Language

b. Gegeben sei weiterhin die unten dargestellte Methode `f`.

Vervollständigen Sie den nachstehenden JML-Methodenvertrag, so dass er Folgendes besagt:

> Wenn vor Aufruf von `f`
>
> - die Vorbedingungen der beiden Aufrufe von `m` (im Methodenrumpf von `f`) erfüllt sind,
> - der Wert des Parameters `p1` echt kleiner als der Wert von `p2` ist,
> - und die Werte der Parameter `q1` und `q2` gleich sind,
>
> dann ist nach Aufruf von `f` die Summe aller Werte des Arrays `res1` nicht größer als die Summe aller Werte des Arrays `res2`. Außerdem gilt dann, dass die Methode `f` terminiert, keine Exception verursacht, und ausschließlich die Felder `res1` und `res2` ändert.

Hinweis: Für diese Aufgabe können Sie den JML-Vertrag der Methode `m` annehmen. Dieser ist unten noch einmal abgedruckt.

```
public class A {
    public int[] res1;
    public int[] res2;

     /*@ ____________________________________________________________

       @  requires ____________________________________________________

       @  ____________________________________________________________

       @  assignable __________________________________________________

       @  ensures _____________________________________________________

       @  ____________________________________________________________

       @  ____________________________________________________________

       @*/
     public void f(int p1, int q1, int p2, int q2) {
         res1 = m(p1, q1);
         res2 = m(p2, q2);
     }

     /*@ public normal_behaviour
       @  requires 0 < q && q <= p;
       @  assignable \nothing;
       @  ensures \result.length == q;
       @  ensures (\forall int i; 0 <= i && i < q; 0 <= \result[i]);
       @  ensures (\sum int i; 0 <= i && i < q; \result[i]) <= p;
       @  ensures (\forall int i; 0 <= i && i < q;
       @      (\sum int j; 0 <= j && j < q; \result[j]) <= \result[i] * q);
       @*/
     public int[] m(int p, int q) { ... }
}
```

7 Lineare Temporale Logik (LTL) und Büchi-Automaten (4 + 4 = 8 Punkte)

a. Formalisieren Sie folgende zwei Sachverhalte als LTL-Formeln über der Signatur $\Sigma = \{P, Q, R\}$.

i. Wenn irgendwann in der Zukunft Q gilt, darf bis dahin nicht P gelten.

---

ii. Immer wenn Q gilt, muss auf P immer irgendwann R folgen.

---

b. Geben Sie einen nicht-deterministischen Büchi-Automaten an, dessen akzeptierte Sprache den Modellen (ω-Wörtern) der LTL-Formel

$$(\Diamond\Box a) \rightarrow \Box\Diamond b$$

über der Signatur $\Sigma = \{a, b\}$ entspricht.

Für das Vokabular $V = \mathbb{P}(\Sigma)$ (Potenzmenge von Σ) werden die folgenden, aus der Vorlesung bekannten Abkürzungen definiert:

$$A = \{M \in V \mid a \in M\} \subset V \qquad B = \{M \in V \mid b \in M\} \subset V$$
$$\bar{A} = \{M \in V \mid a \notin M\} \subset V \qquad \bar{B} = \{M \in V \mid b \notin M\} \subset V$$