Klausur Formale Systeme

Fakultät für Informatik

WS 2014/2015

Prof. Dr. Bernhard Beckert

6. März 2015

Vorname: **Vorname**
Name: **Familienname**
Matrikel-Nr.: **Matr.-Nr.**

Platz-Nr.: **Hörsaal** **Sitzplatz**
Code: **Nonce**

Die Bearbeitungszeit beträgt 60 Minuten.

A1 (10)	A2 (6)	A3 (8)	A4 (7)	A5 (9)	A6 (11)	A7 (9)	Σ (60)

Bewertungstabelle bitte frei lassen!

Gesamtpunkte:

*Name: **Vorname** **Familienname***
*Matr.-Nr.: **Matr.-Nr.***

1 Zur Einstimmung (5+5 = 10 Punkte)

a. Kreuzen Sie in der folgenden Tabelle alles Zutreffende an.

Für jede korrekte Antwort gibt es einen Punkt, **für jede falsche Antwort wird ein *halber Punkt* abgezogen!** (Dabei werden jedoch keinesfalls weniger als 0 Punkte für jede der zwei Teilaufgaben vergeben.)

Hinweise:

- „PL1“ steht für „Prädikatenlogik erster Stufe (mit Gleichheit $\doteq$)“, wie sie in der Vorlesung vorgestellt wurde. Auf diese beziehen sich in Teilaufgabe a. auch die Begriffe „erfüllbar“, „allgemeingültig“ und „unerfüllbar“.
- p, q und P_1, P_2 sind Prädikatensymbole, f ist ein Funktionssymbol, c, d sind Konstantensymbole und x, y sind Variablen.
- Es gelten die üblichen Klammereinsparungsregeln.

	keine Formel der PL1	allgemeingültig	erfüllbar, aber nicht allgemeingültig	unerfüllbar
$(p(c) \wedge \forall x(p(x) \rightarrow p(f(x)))) \rightarrow \forall x\, p(x)$			X	
$(P_1 \vee P_2) \wedge (P_1 \vee \neg P_2) \wedge (\neg P_1 \vee P_2) \wedge (\neg P_1 \vee \neg P_2)$				X
$(\forall x(x \doteq c)) \rightarrow (\forall x(x \doteq d))$		X		
$(\forall x\, p(x)) \rightarrow p(p(c))$	X			
$\forall x\big(p(x) \rightarrow \exists y\, q(y)\big) \quad \leftrightarrow \quad \big((\forall x\, p(x)) \rightarrow \exists y\, q(y)\big)$			X	

b. Bitte kreuzen Sie in der folgenden Tabelle das Zutreffende an. Für korrekte Antworten erhalten Sie einen Punkt, **für falsche Antworten wird *ein Punkt* abgezogen**. Dabei werden jedoch nie weniger als 0 Punkte für diese Teilaufgabe vergeben.

	Richtig	Falsch
Für jeden Büchi-Automaten B über $V = \mathbb{P}(\Sigma)$ gibt es eine LTL-Formel F über Σ, die genau in den omega-Strukturen gilt, die von B akzeptiert werden.		X
Jedes lokal konfluente und terminierende Termersetzungssystem ist konfluent.	X	
Die Allgemeingültigkeit für PL1-Formeln ohne Variablen ist entscheidbar.	X	
Für jede allgemeingültige PL1-Formel in NNF ist die zugehörige Formel in Skolemnormalform auch allgemeingültig.		X
Die modallogische Formel $\Box P \rightarrow \Diamond\Diamond P$ ist in allen reflexiven Kripkerahmen gültig (das heißt: Reflexivität von R impliziert $(S, R) \models \Box P \rightarrow \Diamond\Diamond P$).	X	

Zur Begründung:

a. **i.** Einerseits ist die Formel erfüllbar, z. B. wenn $D = \{a\}$ und $I(p) = \{a\}$. Andererseits ist sie nicht allgemeingültig, weil sie für nicht erfüllt ist für $D = \{c, a\}, I(c) = c, I(f)(x) = c, I(p) = \{c\}$.

ii. Diese KNF listet alle Klauseln, die mit P_1 und P_2 gebildet werden können, auf. Sie ist unerfüllbar. Die ersten beiden Klauseln können (z. B. per Resolution oder per Distributivgesetz) vereinfacht werden zu P_1, die letzten beiden zu $\neg P_1$.

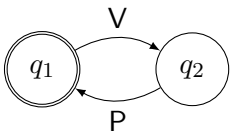
iii. Wenn die Prämisse in einem Modell wahr ist, gibt es nur ein Element u im Universum $D = \{u\}$. Also muss auch $I(d) = u = I(c)$ sein, und die Konklusion gilt. Wenn die Prämisse nicht wahr ist, ist die gesamte Formel auch wahr; insgesamt ist sie allgemeingültig.

iv. Ein Symbol ist entweder ein Prädikaten- oder ein Funktionssymbol. p wird hier aber wie ein Prädikaten- und gleichzeitig wie ein Funktionssymbol verwendet, das ist keine Formel der PL1.

v. Die Formel ist erfüllbar, z. B. für $D = \{a\}, I(q) = \{a\}$. Andererseits ist sie für $D = \{a, b\}, I(p) = \{a\}, I(q) = \emptyset$ nicht erfüllt.

Man kann auch Äquivalenzumformungen machen und erkennen, dass die rechte Seite der Äquivalenz zu $\exists x(p(x) \to \exists y\, q(y))$ äquivalent ist.

b. **i.** Es gibt Büchi-Automaten, für die es keine Entsprechung als LTL-Formel gibt, z. B. der Automat, der besagt, dass p wenigstens in jedem zweiten Zustand gilt:



ii. Satz aus der Vorlesung

iii. Eine PL1-Formel ohne Variablen kann als aussagenlogische Formel verstanden werden, in der die atomaren (Grund-)Formeln die Menge der AL-Variablen bilden. Bekanntermaßen ist die Allgemeingültigkeit von AL entscheidbar.

iv. Ein Gegenbeispiel ist das allgemeingültige $(\forall x\, p(x)) \to (\exists y\, p(y))$. Ihre Skolemnormalform lautet $p(c) \to p(d)$, was nicht allgemeingültig ist.

v. Wenn in einem Zustand s einer Kripkestruktur S mit reflexivem Rahmen R die Formel $\Box P$ gilt $(S, R, s \models \Box P)$, dann gilt wegen Reflexivität auch P (also $S, R, s \models P$). Wiederum wegen der Reflexivität, gibt es einen Nachfolger von s (nämlich s selbst), in dem P gilt: $S, R, s \models \Diamond P$ und auch $S, R, s \models \Diamond\Diamond P$.

*Name: **Vorname** **Familienname***
*Matr.-Nr.: **Matr.-Nr.***

2 Kalkülwahl (6 Punkte)

Ihnen wird die Aufgabe übertragen, ein Beweisersystem zu konzipieren, das die Allgemeingültigkeit von Formeln in PL1 überprüft.

Nennen Sie Vor- und Nachteile, die die Umsetzung des Beweisers im Sequenzenkalkül bzw. im Resolutionskalkül mit sich bringen und Ihre Entscheidung beeinflussen.

Hinweis: Die volle Punktzahl wird für diese Aufgabe erreicht, wenn für beide Kalküle zusammen insgesamt 6 Vor-/Nachteile richtig angegeben sind.

Resolutionskalkül	**Sequenzenkalkül**
Vorteile:	**Vorteile**:
Nur eine Regel und Normalform günstig für Implementierbarkeit	Keine Normalform erforderlich
Nur eine Regel, daher günstig für Nachweis der Korrektheit von Beweisen	Für interaktives Beweisen geeignet, für Menschen gut lesbar
	Gegenbeispiele aus offen gebliebenen Sequenzen direkt ablesbar
Nachteile:	**Nachteile**:
Ungeeignet für interaktives Beweisen	Viel Schreibarbeit, darum für Papier-und-Bleistift-Beweise ungeeignet
Normalform für Menschen schwer lesbar	Viele Regeln, darum großer Implementierungsaufwand und Korrektheit der Implementierung schwieriger sicherzustellen

3 DPLL (8 Punkte)

Gegeben sei die folgende Menge M aussagenlogischer Klauseln:

$$\big\{\{A, \neg B, C, \neg D, \neg E\},\ \{\neg B, E\},\ \{A, C\},\ \{\neg A, E\},\ \{B, \neg C\},\ \{C, \neg E\},\ \{\neg B, \neg C, \neg E\}\big\}$$

Zeigen Sie mit Hilfe des Davis-Putnam-Loveland-Algorithmus, dass M unerfüllbar ist. Geben Sie die Zwischenschritte an.

Da M keine Unit-Klauseln enthält, beginnen wir mit einer Fallunterscheidung. Als Variable dafür bietet sich z.B. A an, da dadurch Unit-Klauseln entstehen. Die Ergebnisse der Unit-Propagation sind dabei schrittweise von links nach rechts abgebildet.

- Fall $A \leftarrow 0$

$\underline{A \leftarrow 0}$	$\underline{C \leftarrow 1}$	$\underline{B \leftarrow 1}$	$\underline{E \leftarrow 1}$
$\{\neg B, C, \neg D, \neg E\}$	–	–	–
$\{\neg B, E\}$	$\{\neg B, E\}$	$\{E\}$	–
$\{C\}$	–	–	–
–	–	–	–
$\{B, \neg C\}$	$\{B\}$	–	–
$\{C, \neg E\}$	–	–	–
$\{\neg B, \neg C, \neg E\}$	$\{\neg B, \neg E\}$	$\{\neg E\}$	$\square$

- Fall $A \leftarrow 1$

$\underline{A \leftarrow 1}$	$\underline{E \leftarrow 1}$	$\underline{C \leftarrow 1}$	$\underline{B \leftarrow 1}$
–	–	–	–
$\{\neg B, E\}$	–	–	–
–	–	–	–
$\{E\}$	–	–	–
$\{B, \neg C\}$	$\{B, \neg C\}$	$\{B\}$	–
$\{C, \neg E\}$	$\{C\}$	–	–
$\{\neg B, \neg C, \neg E\}$	$\{\neg B, \neg C\}$	$\{\neg B\}$	$\square$

Jede Fallunterscheidung führt demnach zu einem Konflikt (leerer Klausel). Damit ist M unerfüllbar.

4 Formalisieren in PL1 (1+2+2+2 = 7 Punkte)

Formalisieren Sie die vier folgenden Aussagen über die Knoten und Kanten in einem eingefärbten Graphen mittels Prädikatenlogik erster Stufe mit Gleichheit. Benutzen Sie dafür jeweils die angegebenen interpretierten Symbole.

Sie können davon ausgehen, dass nur solche prädikatenlogischen Interpretationen (D, I) zur Auswertung der Formeln verwendet werden, deren Universum D gleich der Menge der Knoten des Graphen ist.

a. Jeder Knoten ist entweder schwarz oder weiß.
Prädikat(e): $S(\cdot), W(\cdot)$.

$\boxed{\forall x.\, \neg(S(x) \leftrightarrow W(x))}$

---

b. Es gibt mindestens zwei verschiedene schwarze Knoten.
Prädikat(e): $S(\cdot)$.

$\boxed{\exists x.\, \exists y.\, (S(x) \wedge S(y) \wedge \neg x \doteq y)}$

---

c. Es gibt höchstens zwei schwarze Knoten.
Prädikat(e): $S(\cdot)$.

$\boxed{\forall x.\, \forall y.\, \forall z.\, ((S(x) \wedge S(y) \wedge S(z) \wedge \neg y \doteq z) \rightarrow (x \doteq y \vee x \doteq z))}$

---

d. Von jedem schwarzen Knoten geht eine Kante zu mindestens einem weißen Knoten.
Prädikat(e): $S(\cdot), W(\cdot), K(\cdot, \cdot)$.

$\boxed{\forall x.\, (S(x) \rightarrow \exists y.\, (K(x, y) \wedge W(y)))}$

---

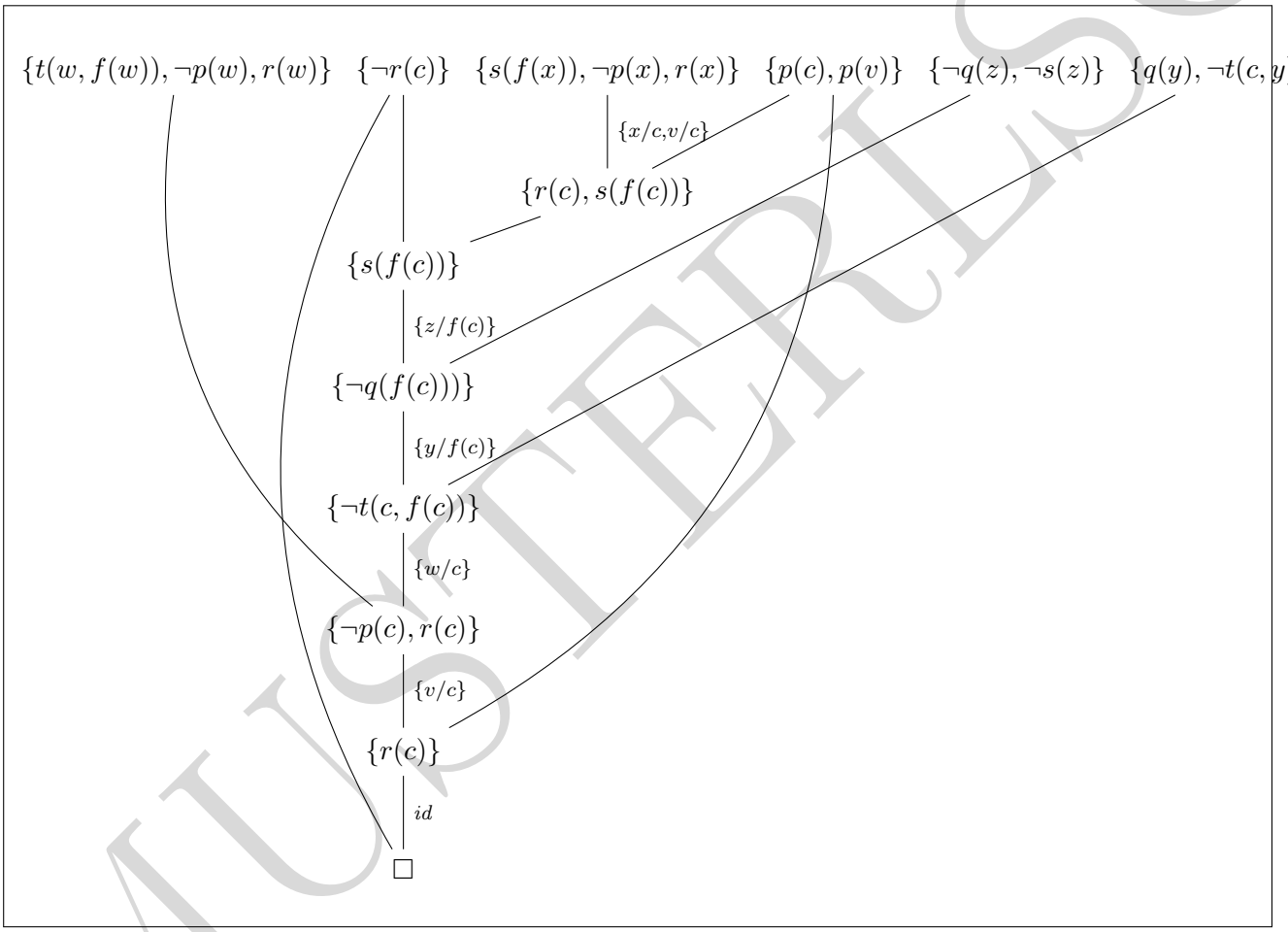
5 Resolution (9 Punkte)

Zeigen Sie mittels Resolution, dass die Menge folgender Klauseln unerfüllbar ist.

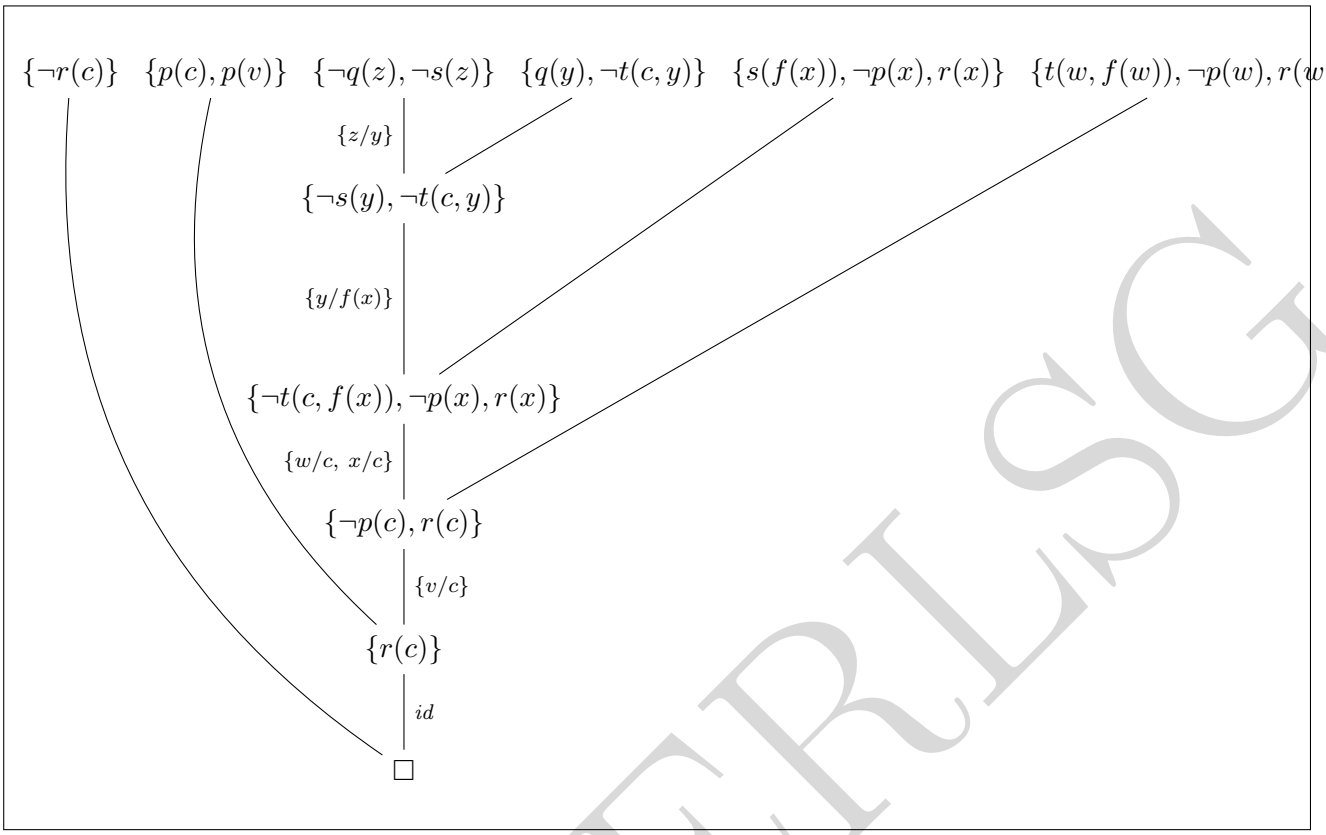
Notieren Sie Ihren Beweis so, dass bei jeder neu entstehenden Klausel klar erkennbar ist, aus welchen Elternklauseln sie entsteht und welche Substitution verwendet wurde.

Signatur: p, q, r, s, t sind Prädikate; c ist eine Konstante; f ist eine Funktion; v, w, x, y, z sind Variablen.

$\{\neg r(c)\}$ $\{p(c), p(v)\}$ $\{\neg q(z), \neg s(z)\}$ $\{q(y), \neg t(c, y)\}$ $\{s(f(x)), \neg p(x), r(x)\}$ $\{t(w, f(w)), \neg p(w), r(w)\}$



*Name: **Vorname** **Familienname***
*Matr.-Nr.: **Matr.-Nr.***



6 Spezifikation mit der Java Modeling Language

((2+2+3)+(1+1+2) = 11 Punkte)

a. Gegegeben sei die Java-Klasse `BasedArray`, in der ein Array `a` von ganzzahligen Werten zusammen mit einem Basiswert `base` gespeichert wird. Ein Objekt vom Typ `BasedArray` stellt eine Folge $\langle e_0, \ldots, e_{\texttt{a.length}-1} \rangle$ dar, wobei $e_i = \texttt{base} + \texttt{a}[i]$ (für $0 \leq i < \texttt{a.length}$).

Beispiel:
Ein `BasedArray`-Objekt `ba` mit `ba.a = {-2,1,0}` und `ba.base = 3` repräsentiert $\langle 1, 4, 3 \rangle$.

Ein `BasedArray`-Objekt heißt normalisiert, wenn

(N1) das Array `a` (mindestens) einen Eintrag mit dem Wert 0 hat,

(N2) kein Eintrag in `a` negativ ist.

Beispiel:
Ein `BasedArray`-Objekt `nba` mit `nba.a = {0,3,2}` und `nba.base = 1` repräsentiert (wie das Objekt `ba`) die Folge $\langle 1, 4, 3 \rangle$*; zudem ist `nba` normalisiert.*

Ergänzen Sie im folgenden Quelltext den JML-Vertrag einer Methode `normalize`, indem Sie die obigen Normalisierungsbedinungen (N1) und (N2) in JML formalisieren und zudem:

(N3) das Objekt repräsentiert nach dem Aufruf von `normalize` die gleiche Folge wie zuvor.

```
class BasedArray {
   int[] a;
   int base;

   /*@ public normal_behaviour
     @   requires a.length > 0;
     @
     @   ensures (N1)
     @
     @      (\exists int i; 0 <= i && i < a.length; a[i] == 0);
     @
     @
     @   ensures (N2)
     @
     @      (\forall int i; 0 <= i && i < a.length; a[i] >= 0);
     @
     @
     @   ensures (N3)
     @
     @      (\forall int i; 0 <= i && i < a.length; a[i] + base == \old(a[i] + base));
     @
     @
     @   assignable base, a[*];
     @*/
   void normalize() { /* ... */ }
}
```

Fortsetzung 6 Spezifikation mit der Java Modeling Language

b. Gegeben sei die folgende Klassendefinition:

```
class C {
  int[] a;
  /*@ nullable @*/ C n;
}
```

Geben Sie für die folgenden JML-Klasseninvarianten die Bedeutung in natürlicher Sprache wieder:

i. `/*@ invariant (\forall int i; 0<=i && i<a.length-1; a[i] <= a[i+1]); */`

Die Einträge im Array `this.a` sind aufsteigend sortiert.
oder
Für jeden Eintrag im Array, der einen Nachfolger hat (/außer dem letzten) gilt, dass er kleiner oder gleich seinem Nachfolger ist.

ii. `/*@ invariant n != null ==> n.a.length > a.length; */`

Wenn die Referenz `this.n` von `null` verschieden ist, so ist das Array, auf das `this.n.a` zeigt, länger als das Array, auf das `this.a` zeigt.

iii.
```
/*@ invariant a.length < 5 ==>
  @          n != null && (\sum int i; 0<=i && i<a.length; a[i]) == n.a[0]; */
```

Wenn das Array, auf das `this.a` zeigt, (echt) weniger als fünf Einträge hat, so ist `this.n` nicht `null` und das erste Element des Arrays, auf das `this.n.a` zeigt, enthält die Summe der Werte im Array `this.a`.

*Name: **Vorname** **Familienname***
*Matr.-Nr.: **Matr.-Nr.***

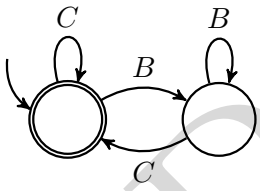
7 Büchi-Automaten und LTL (4+3+2 = 9 Punkte)

Gegeben sei eine AL-Signatur Σ, die die drei Variablen a, b und c enthält. Für das Vokabular $V = \mathbb{P}(\Sigma)$ (Potenzmenge von Σ) werden die folgenden aus der Vorlesung bekannten Abkürzungen definiert:

$$A = \{M \in V : a \in M\} \subset V$$
$$B = \{M \in V : b \in M\} \subset V$$
$$C = \{M \in V : c \in M\} \subset V$$

a. Kreuzen Sie für den folgenden Büchi-Automaten $\mathcal{A}$ über dem Alphabet $V = \mathbb{P}(\Sigma)$ und die folgenden LTL-Formeln φ_i je an, ob $L^\omega(\mathcal{A}) = \{\xi \in V^\omega : \xi \models \varphi_i\}$ gilt oder nicht (ob also $\mathcal{A}$ und φ_i äquivalent sind oder nicht).

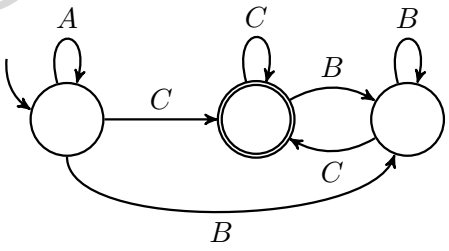
Falls diese beiden Sprachen nicht gleich sind, geben Sie ein Gegenbeispiel ξ an, so dass $\xi \models \varphi_i$ und $\xi \notin L^\omega(\mathcal{A})$ gilt. Sie können eine omega-Struktur $\xi : \mathbb{N} \to V$ als Gegenbeispiel angeben, indem Sie den Anfang der Zustandsfolge als Folge von Elementen aus V ausdrücken.



Gilt $L^\omega(\mathcal{A}) = \{\xi \in V^\omega : \xi \models \varphi_i\}$?

$\varphi_1 \equiv \Box(b \,\mathbf{U}\, c)$	[X] Ja [ ] Nein, Gegenbeispiel:	_____
$\varphi_2 \equiv (\Box b) \,\mathbf{U}\, c$	[ ] Ja [X] Nein, Gegenbeispiel:	$\xi(0) = \{\{c\}\}, \xi(i) = \{\{b\}\}$ für alle $i \in \mathbb{N}, i > 0$
$\varphi_3 \equiv \Box((b \vee c) \wedge \Diamond c)$	[X] Ja [ ] Nein, Gegenbeispiel:	_____

b. Geben Sie zu folgendem Büchi-Automaten $\mathcal{A}$ eine LTL-Formel φ an, so dass $L^\omega(\mathcal{A}) = \{\xi \in V^\omega : \xi \models \varphi\}$ gilt.



Lösung: $a \,\mathbf{U}\, \Box(b \,\mathbf{U}\, c)$

c. Geben Sie zu der folgenden LTL-Formel eine äquivalente LTL-Formel an, die als temporale Modaloperatoren nur $\Box$ und $\Diamond$ verwendet (d. h., **U** kommt nicht vor):

$$(\Box a) \,\mathbf{U}\, (\Box b) \quad \leftrightarrow \quad (\Box a \vee \Box b) \wedge \Diamond \Box b$$