Universität Karlsruhe (TH) – Fakultät für Informatik
Institut für Logik, Komplexität und Deduktionssysteme
Lehrstuhl Prof. Dr. P. Deussen

Informatik III – Prof. Dr. P. Deussen im Wintersemester 2001/2002

C. Bierwald, Th. Käufl, F. Schuster http://i10www.ira.uka.de/lehre/info3-winter2001 info3@ira.uka.de

Übungsblatt 14 – Lösungsvorschläge

1.2.2002

Aufgabe 1 Das kleine Einmaleins der Fixpunkttheorie (Käufl)
Im folgenden seien M und M' Bereiche und K eine Kette.

a) Zeigen Sie: Ist f: $M \to M'$ eine monotone Abbildung, dann ist $f(K)$ eine Kette.

Sind x, $y \in f(K)$, dann gibt es u, $v \in K$, sodaß $f(x) = u$, $f(y) = v$ und $u \le v$ (oder $v \le u$). Da f monoton und K Kette ist, erhalten wir $x \le y$.

b) Sei $\sup K \in K$ und f: $M \to M'$ eine monotone Abbildung des Bereichs M nach dem Bereich M'. Zeigen Sie $f(\sup K) = \sup f(K)$.

Da f monoton ist $\sup f(K) \le f(\sup K)$. Es bleibt noch die umgekehrte Ordnungsbeziehung nachzuweisen. Ist $\sup K \in K$, dann ist $f(\sup K) \in f(K)$. Also $f(\sup K) \le \sup f(K)$.

c) Seien M und M' Bereiche und $\sup K \in K$ für alle Ketten K in M. Zeigen Sie, daß jede monotone Funktion f: $M \to M'$ stetig ist.

Die Behauptung folgt sofort aus der Teilaufgabe b.

d) $N_\top$ sei die Menge der natürlichen Zahlen N, zu der wir das neue Element $\top$ hinzufügen. Zeigen Sie, daß $N_\top$ vollständig halbgeordnet ist, wenn $n \le \top$ für alle n und $\le$ die übliche Ordnung für n, $m \in N$ ist. Geben sie eine monotone, nicht stetige Funktion von $N_\top$ nach $N_\top$ an.

$N_\top$ ist total geordnet und hat 0 als kleinstes Element. Jede endliche Teilmenge von N hat ihr Supremum in N. Das Supremum von $N_\top$ ist $\top$. (Wäre $\sup N_\top \in N$, also $\sup N_\top = n$, dann wäre $\sup N_\top = n + 1$.) Also ist $N_\top$ vollständig halbgeordnet.

Die Abbildung f mit $f(\top) = \top$ und $f(n) = 0$ sonst, ist monoton. Sie ist nicht stetig, da $\sup f(N) = 0$ und $f(\sup N) = \top$.

e) Sei auch M'' ein Bereich und g: $M' \to M''$ stetig. Zeigen Sie, daß $f \circ g$: $M \to M''$ stetig ist.

Ist K eine Kette, dann auch $f(K)$.

$g(f(sup\ K))$	
$= g(sup\ f(K))$	da f stetig
$= sup(g(f(K))$	da g stetig und $f(K)$ Kette

Aufgabe 2 Ein paar stetige Funktionen (Käufl)
Mit $T = \{\bot, w, f\}$ und $\bot \le w$ und $\bot \le f$ erhalten wir den Bereich der Wahrheitswerte.

a) Geben Sie stetige Funktionen von $T \times T$ bzw. T nach T an, die beschränkt auf $\{w, f\}$ die Konjunktion bzw. die Negation sind. (Hinweis: Eine Funktion mehrerer Argumente ist genau-dann stetig, wenn sie für jedes einzelne Argument stetig ist.)

Damit wir eine stetige Funktion k erhalten, die als Konjunktion verwendet werden kann, muß $k(\perp, x) = \perp$ und $k(x, \perp) = \perp$. Für die Negation n muß $n(\perp) = \perp$ gelten. (Die Monotonie ist hinreichend für die Stetigkeit.)

b) Sei M ein Bereich. Zeigen Sie, daß die Funktion $b \to x, y$, die $T \times M \times M$ nach M abbildet stetig in allen Argumenten ist, wenn $\perp \to x, y = \perp$, $w \to x, y = x$ und $f \to x, y = y$. (Beachten Sie den Hinweis in Teilaufgabe a.)

Auf Grund der Definition ist die Funktion im ersten Argument monoton und somit stetig.

Aufgabe 3 Sprachen als Fixpunkte (Käufl)

Zeigen Sie, daß die Grammatik mit den Nichtterminalzeichen $N = \{S, G, U\}$, den Terminalzeichen $T = \{0, 1\}$, dem Startzeichen S und den Produktionen

$$S ::= GG \mid U \mid \varepsilon,\ G ::= 1 \mid 0G \mid G0 \mid GGG,\ U ::= 0 \mid 0U$$

nur Worte erzeugt, die eine gerade Anzahl von 1 enthalten.

Verwenden Sie dabei die Fixpunktgleichung, die die erzeugte Sprache als Lösung hat. (Siehe Skriptum Satz 15.2.) Zeigen Sie zuerst, daß in jeder Näherung der Lösung die aus S ableitbaren Terminalwörter eine gerade Anzahl von 1 enthalten.

Das Gleichungssystem, dessen Lösung die erzeugte Sprache ist, lautet:

$$\overline{\Pi}_1(\bar{S}, \bar{G}, \bar{U}) = \bar{G}\bar{G} \cup \bar{U} \cup \{\varepsilon\}$$

$$\overline{\Pi}_2(\bar{S}, \bar{G}, \bar{U}) = \{1\} \cup \{0\}\bar{G} \cup \bar{G}\{0\} \cup \bar{G}\bar{G}\bar{G}$$

$$\overline{\Pi}_3(\bar{S}, \bar{G}, \bar{U}) = \{0\} \cup \{0\}\bar{U}$$

Dabei ergibt $\overline{\Pi}_1$ die erzeugte Sprache. Die Näherungen für die Lösung sind

$$\overline{\Pi}_1^0(\emptyset, \emptyset, \emptyset) = \{\varepsilon\}$$

$$\overline{\Pi}_2^0(\emptyset, \emptyset, \emptyset) = \{1\}$$

$$\overline{\Pi}_3^0(\emptyset, \emptyset, \emptyset) = \{0\}$$

und

$$\overline{\Pi}_1^{k+1}(\emptyset, \emptyset, \emptyset) = \overline{\Pi}_2^k\,\overline{\Pi}_2^k \cup \overline{\Pi}_3^k \cup \{\varepsilon\}$$

$$\overline{\Pi}_2^{k+1}(\emptyset, \emptyset, \emptyset) = \{1\} \cup \{0\}\overline{\Pi}_2^k \cup \overline{\Pi}_2^k\{0\} \cup \overline{\Pi}_2^k\,\overline{\Pi}_2^k\,\overline{\Pi}_2^k$$

$$\overline{\Pi}_3^{k+1}(\emptyset, \emptyset, \emptyset) = \{0\} \cup \{0\}\overline{\Pi}_3^k$$

Wir zeigen durch Induktion

Die Wörter in $\overline{\Pi}_1^k$ *haben eine gerade Anzahl von* 1 *und die in* $\overline{\Pi}_2^k$ *eine ungerade. In den Wörtern in* $\overline{\Pi}_3^k$ *kommt* 1 *nicht vor.*

(i) Für k = 0 liest man die Behauptung unmittelbar aus den Gleichungen für die Näherungen der Lösungen ab.
(ii) Den Gleichungen für die Näherungen $k + 1$ entnimmt man:

Die Wörter in $\overline{\Pi}_3^{k+1}$ enthalten 1 nicht, wenn dies für $\overline{\Pi}_3^k$ der Fall ist.

Die Wörter in $\overline{\Pi}_2^{k+1}$ enthalten eine ungerade Anzahl von 1, wenn dies für $\overline{\Pi}_2^k$ zutrifft.

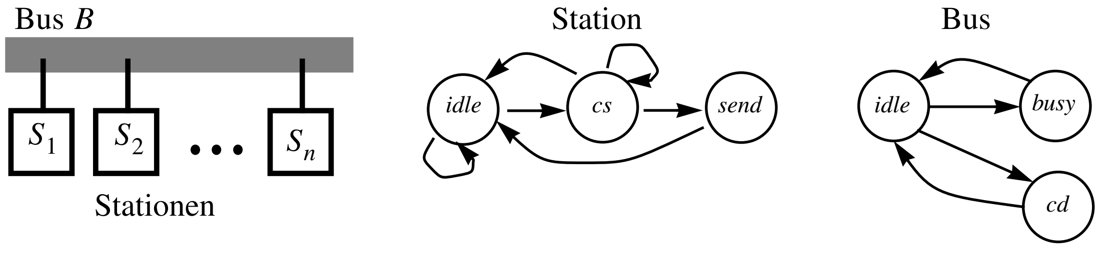
Die Wörter in $\overline{\Pi}_1^{k+1}$ enthalten eine gerade Anzahl von 1, wenn die Wörter in $\overline{\Pi}_2^k$ eine ungerade Anzahl enthalten und wenn 1 in den Wörtern in $\overline{\Pi}_3^k$ nicht vorkommt.

Die Induktionsvoraussetzung erfüllt die Voraussetzungen in diesen drei Aussagen.

Ist w ein Wort der erzeugten Sprache, dann gibt es ein k, sodaß $w \in \overline{\Pi}_1^k$. Also kommt in w nur eine geradzahlige Anzahl von 1 vor.

Aufgabe 4 *Ethernet* in a nutshell (Bierwald)

Ein Computernetzwerk bestehe aus n Stationen $S_1, \ldots, S_n$, die an einem gemeinsamen Datenbus B hängen. Das folgende Protokoll regelt den Zugriff auf den gemeinsamen Bus:
1. Eine Station kann im Zustand *idle* verbleiben.
2. Eine Station, die senden möchte, wechselt vom Zustand *idle* in den Zustand *cs* (carrier sense).
3. Wenn der Bus dann frei ist (Bus im Zustand *idle*), beschreibt ihn die Station und wechselt dazu vom Zustand *cs* in den Zustand *send* und der Bus gleichzeitig in den Zustand *busy.*
4. Danach wechseln Station und Bus wieder in den Zustand *idle*.
5. Wenn der Bus nicht frei ist, bleibt die Station im Zustand *cs*.
6. Es kann passieren, daß mehrere Stationen zur gleichen Zeit den Bus beschreiben. Dann wechselt der Bus nicht in den Zustand *busy* sondern in den Zustand *cd* (collision detected).
7. Anschließend gehen *alle* Stationen und der Bus in den Zustand *idle* über.



Zustandsmengen: $S_{\text{Station}} = \{idle, cs, send\}$ $S_{\text{Bus}} = \{idle, busy, cd\}$

Betrachten Sie den Fall $n = 2$.
a) Geben Sie die Zustandsmenge $S \subseteq S_{\text{Station}} \times S_{\text{Station}} \times S_{\text{Bus}}$ und die Zustandsübergangsrelation $R \subseteq S \times S$ an, die dieses Computernetzwerk beschreibt. Betrachten Sie nur Zustände, die vom Startzustand $s_0 = (idle, idle, idle)$ aus erreichbar sind. Bilden Sie die Übergänge gemäß der obigen Regeln[1]. Alle Stationen schalten gleichzeitig (synchrone Übergänge).

1. Bei in natürlicher Sprache beschriebenen Protokollen tritt häufig das Problem der Mehrdeutigkeit auf. Falls Sie eine solche in dieser Beschreibung entdecken, entscheiden Sie sich für eine Lesart mit der Sie die Aufgabe zu Ende bearbeiten.

Mögliche Übergänge sind zum Beispiel:

von	nach	S_1	S_2	B
$(idle, idle, idle)$	$(cs, idle, idle)$	Regel 2	Regel 1	-
$(cs, idle, idle)$	$(send, cs, busy)$	Regel 3	Regel 2	Regel 3
$(send, cs, busy)$	$(idle, cs, idle)$	Regel 4	Regel 5	Regel 4

b) Ist das System verklemmungsfrei, d.h. hat jeder erreichbare Zustand einen Nachfolgerzustand? (Hinweis. Das sollte der Fall sein.)

c) Betrachten Sie nun die Kripke-Struktur $K = (S, R, A, L)$ mit den atomaren Aussagen $A = \{collision\}$ und der Markierung L der Zustände definiert durch, $L((send, send, cd)) = \{collision\}$ und $L(s) = \emptyset$ für die übrigen Zustände $s \in S$. Zeigen Sie mit dem Markierungsverfahren aus der Vorlesung, daß Kollisionen möglich sind ($s_0 \models \mathrm{E}\Diamond collision$).

d) Welche der folgenden Aussagen sind wahr?

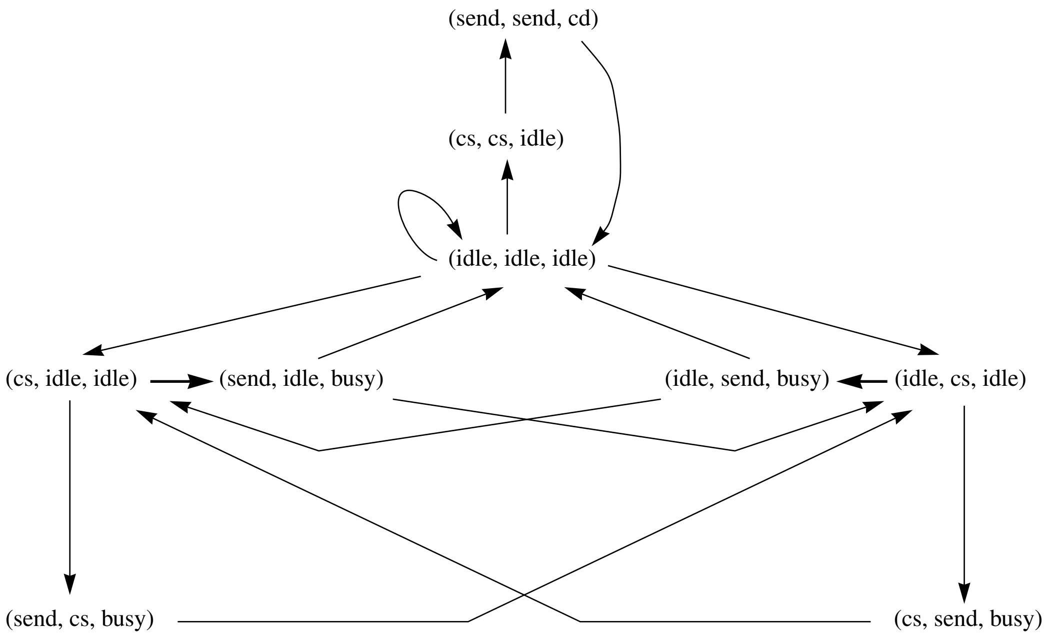
1. $s_0 \models \mathrm{A}\Diamond collision$
2. $s_0 \models \mathrm{A}\Box f$ mit f = „Wenn der Bus im Zustand *cd* ist, sind mindestens zwei Stationen im Zustand *send*.“
3. $s_0 \models \mathrm{A}\Box f$ mit f = „Wenn mindestens zwei Stationen im Zustand *send* sind, ist der Bus im Zustand *cd*.“

e) In diesem Modell gibt es eine Zustandsübergangsbahn, auf der es immer wieder zu Kollisionen kommt: $(idle, idle, idle)$, $(cs, cs, idle)$, $(send, send, cd)$, $(idle, idle, idle)$, $(cs, cs, idle)$ und so fort.

Wie geht man beim echten Ethernet-Protokoll mit diesem Problem – und es ist ein Problem, wird doch jeder Sendeversuch einer Station von einer anderen gestört! – um?

Lösungsvorschläge.

a)



b) Das System ist verklemmungsfrei.

c) Wir bestimmen den kleinsten Fixpunkt der Funktion $g(Z) = Ext(collision) \cup R \cdot Z$.

$Ext(collision) = \{(send, send, cd)\}$

$$\begin{aligned} g^1(\emptyset) &= Ext(collision) \cup R \cdot \emptyset \\ &= \{(send, send, cd)\} \cup \emptyset \\ &= \{(send, send, cd)\} \end{aligned}$$

$$\begin{aligned} g^2(\emptyset) &= Ext(collision) \cup R \cdot g^1(\emptyset) \\ &= \{(send, send, cd)\} \cup R \cdot \{(send, send, cd)\} \\ &= \{(send, send, cd)\} \cup \{(cs, cs, idle)\} \\ &= \{(send, send, cd), (cs, cs, idle)\} \end{aligned}$$

$$\begin{aligned} g^3(\emptyset) &= Ext(collision) \cup R \cdot g^2(\emptyset) \\ &= \{(send, send, cd)\} \cup R \cdot \{(send, send, cd), (cs, cs, idle)\} \\ &= \{(send, send, cd)\} \cup \{(send, send, cd), (cs, cs, idle), (idle, idle, idle)\} \\ &= \{(send, send, cd), (cs, cs, idle), (idle, idle, idle)\} \end{aligned}$$

usw.

Nach 6 Schritten ist $g^6(\emptyset) = g^7(\emptyset) = S$ und es verändert sich nichts mehr. Da der Startzustand s_0 im Fixpunkt enthalten ist, gilt die Behauptung $s_0 \models \mathrm{E}\Diamond collision$.

d)

1. Die Aussage besagt, daß auf *jeder* vom Startzustand ausgehenden Bahn *irgendwann* ein Zustand kommt, in dem die Aussage $collision$ wahr ist oder daß – mit anderen Worten – eine Kollision unvermeidbar ist. Das ist falsch. Ein Gegenbeispiel ist die folgende Bahn: $(idle, idle, idle)$, $(cs, idle, idle)$, $(send, idle, busy)$, $(idle, idle, idle)$, $(cs, idle, idle)$, $(send, idle, busy)$ und so weiter, bei der sich die ersten drei Zustände immer wiederholen und die Aussage $collision$ nie wahr wird.

2. Die Aussage besagt, das auf *jeder* vom Startzustand ausgehenden Bahn *immer* die Eigenschaft f gilt. Mit anderen Worten: Immer wenn der Bus im Zustand cd ist, sind auch mindestens zwei Stationen gleichzeitig am senden.

3. Die Aussage besagt, das auf *jeder* vom Startzustand ausgehenden Bahn *immer* die Eigenschaft f gilt. Mit anderen Worten: Immer wenn mindestens zwei Stationen gleichzeitig senden, ist der Bus im Zustand cd.

Die Aussagen 2 und 3 sind wahr, da jeder von s_0 aus erreichbare Zustand die jeweilige Eigenschaft f erfüllt. Damit ist zum einen gezeigt, daß der Bus nicht grundlos den Zustand cd (collision detected, Kollision erkannt) annimmt und zum anderen, daß keine Kollision unentdeckt bleibt.

e) Beim echten Ethernet-Protokoll warten die an einer Kollision beteiligten Stationen eine zufällige Zeitspanne bis sie einen erneuten Sendeversuch wagen. Je häufiger es zu Kollisionen kommt, um so länger dauert diese Zeitspanne.