ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

Distributed Computing

HS 2012 Prof. C. Stamm / K.-T. Förster, T. Langner, J. Seidel, Prof. R. Wattenhofer

Diskrete Ereignissysteme

Prüfung

Donnerstag, 31. Januar 2013, 9:00 – 12:00 Uhr

Nicht öffnen oder umdrehen bevor die Prüfung beginnt!

Die Prüfung dauert 180 Minuten und es gibt insgesamt 180 Punkte. Die Anzahl Punkte pro Teilaufgabe steht jeweils in Klammern bei der Aufgabe. Sie dürfen die Prüfung auf Englisch oder Deutsch beantworten. Begründen Sie alle Ihre Antworten und beschriften Sie Skizzen und Zeichnungen verständlich.

Schreiben Sie zu Beginn Ihren Namen und Ihre Legi-Nummer in das folgende dafür vorgesehene Feld und beschriften Sie jedes beschriebene Blatt mit Ihrem Namen und Ihrer Legi-Nummer.

Name	**Legi-Nr.**

Punkte

Aufgabe	**Erreichte Punktzahl**	**Maximale Punktzahl**
1		38
2		35
3		57
4		50
Summe		180

1 Formale Sprachen (38 Punkte)

a) [6] Wandeln Sie den regulären Ausdruck $(a^*(a^*b)^* \cup caa^*)^*$ in einen NFA um.

b) [6] Geben Sie einen DFA für den regulären Ausdruck aus Teil **a)** an.

c) [12] Wir betrachten das Alphabet

$$\Sigma = \left\{ \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} \right\} ,$$

das aus allen drei-elementigen Vektoren der Werte 0 und 1 besteht. Ein Wort über dem Alphabet Σ besteht aus drei Reihen von 0en und 1en. Wir betrachten jede Reihe als binäre Zahl und definieren die Sprache

$$\mathcal{L} = \{w \in \Sigma^* \mid \text{die unterste Reihe ist die Summe der beiden oberen Reihen}\} ,$$

und zusätzlich gilt $\varepsilon \notin L$.

Beispiele:

$$\begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \notin \mathcal{L} \quad \text{da} \quad 11_2 + 00_2 = 11_2 \neq 01_2 \qquad (3 + 0 = 3 \neq 1)$$

$$\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} \in \mathcal{L} \quad \text{da} \quad 011_2 + 010_2 = 101_2 \qquad (3 + 2 = 5)$$

Beweisen Sie, dass $\mathcal{L}$ regulär ist.

d) [6] Beweisen oder widerlegen Sie die Aussage: Die Sprache $\mathcal{L} = \{a^m b^n \mid m \neq n\}$ für $m, n \geq 0$ über dem Alphabet $\Sigma = \{a, b\}$ ist regulär.

e) [4] Zeigen Sie, dass die kontextfreie Grammatik mit den folgenden Produktionen *mehrdeutig* ist.

$$S \to AB$$
$$A \to AA \mid a$$
$$B \to BB \mid b$$

f) [4] Geben Sie eine eindeutige kontextfreie Grammatik für die Sprache aus **e)** an und begründen Sie, weshalb Ihre Grammatik eindeutig ist.

2 Markovketten (35 Punkte)

Wir betrachten das folgende Spiel für zwei Spieler S_1 und S_2: Beide Spieler wählen jeweils eine verschiedene Sequenz des Ausgangs zweier Münzwürfe (`KK`, `KZ`, `ZK`, `ZZ`, wobei `K` für Kopf und `Z` für Zahl steht). Es wird eine „unfaire“ Münze geworfen, die mit Wahrscheinlichkeit $\frac{2}{3}$ Kopf und mit $\frac{1}{3}$ Zahl zeigt. Dies wird wiederholt, bis zwei aufeinander folgende Münzwürfe der von einem der beiden Spieler gewählten Sequenz entsprechen. Es gewinnt der Spieler, dessen Sequenz zuerst geworfen wurde.

Beispiel: S_1 wählt `KK`, S_2 wählt `ZK`. Die Münzwürfe ergeben die Sequenz `KZZZK`. Es gewinnt S_2, da die Teilsequenz `ZK` zuerst geworfen wurde.

a) (i) `[8]` Modellieren Sie die wesentlichen Aspekte des obigen Spiels als Markovkette $\mathcal{M}_1$ unter der Annahme, dass noch kein Spieler seine Wahl getroffen hat.
Hinweis: $\mathcal{M}_1$ hat sieben Zustände.

(ii) `[2]` Ist die Markovkette $\mathcal{M}_1$ ergodisch? Begründen Sie Ihre Antwort!

(iii) `[5]` Hat $\mathcal{M}_1$ eine eindeutige stationäre Verteilung? Falls ja, geben Sie diese an, falls nein, begründen Sie weshalb nicht.

b) S_1 hat sich für die Sequenz `KK` entschieden, während S_2 die Sequenz `ZK` gewählt hat.

(i) `[4]` Geben Sie die Markovkette $\mathcal{M}_2$ an, die sich daraus ergibt.

(ii) `[2]` Zeigen Sie, dass S_2 bei dieser Wahl mit höherer Wahrscheinlichkeit als S_1 gewinnt.

(iii) `[2]` Geben Sie zwei verschiedene stationäre Verteilungen von $\mathcal{M}_2$ an.

c) `[12]` Beweisen Sie, dass S_2 sicherstellen kann, mit höherer Wahrscheinlichkeit zu gewinnen, wenn er seine Wahl erst trifft, nachdem er die Wahl von S_1 bereits kennt. In anderen Worten: Zeigen Sie, dass es für jede Wahl von S_1 eine Wahl für S_2 gibt, so dass S_2 mit höherer Wahrscheinlichkeit als S_1 gewinnt.

3 Unendliche endliche Automaten (57 Punkte)

Ein *Zellularautomat* ist eine Art „intelligentes unendliches Band“, das nicht (wie bei Turingmaschinen) durch einen mobilen Schreib-/Lesekopf bearbeitet wird, sondern bei dem (wie bei endlichen Automaten) eine einzelne *Zelle* des Bandes ihren *Zustand* (nicht aber ihre Position!) von sich aus verändern kann. Ein Zellularautomat wird durch **eine** Menge Q und **eine** Funktion δ beschrieben, wobei

- Q eine **endliche** Menge von *Zuständen*,
- $\sqcup \in Q$ ein reservierter *Schlafzustand*, und
- $\delta : Q^3 \to Q$ eine *Zustandsübergangs-Funktion* ist, für die $\delta(\sqcup, \sqcup, \sqcup) = \sqcup$ gelten muss.

Der Folgezustand einer Zelle z wird in Abhängigkeit des Zustandes q_z von z selbst sowie der Zustände q_{z-1} und q_{z+1} der linken und rechten *Nachbarzelle* $z-1$ bzw. $z+1$ bestimmt. Ein *Schritt* wird durchgeführt, indem alle Zellen **gleichzeitig** ihren Zustand ändern. Die *Konfiguration* c des Zellularautomaten ist eine Funktion $c \in Q^{\mathbb{Z}}$, welche jeder Zelle $z \in \mathbb{Z}$ den Zustand $c(z)$ zuordnet, d.h. nach Durchführung eines Schrittes gilt $c_{\text{neu}}(z) = \delta(c(z-1), c(z), c(z+1))$. Eine *Eingabe* erhält der Zellularautomat durch seine *Startkonfiguration* – das „Eingabealphabet“ ist also als Teil der Zustandsmenge aufgefasst. Links und rechts der gegebenen Eingabe sind, ähnlich wie bei Turingmaschinen, unendlich viele Zellen, die sich im Schlafzustand befinden. Die Ausführung *terminiert*, wenn sich die Konfiguration des Zellularautomaten durch einen Schritt nicht verändert.

3.1 Ein einfacher Zellularautomat (12 Punkte)

Zellularautomat:

$$Q := \{a, b, \times, \sqcup\}$$

q_{z-1}	q_z	q_{z+1}	$\delta(\cdot,\cdot,\cdot)$
$\sqcup$	$\times$	?	a
a	$\times$	?	b
b	$\times$	?	a
a	$\sqcup$	?	b

Ausführung:

$\cdots$	$\sqcup$	$\sqcup$	$\times$	$\times$	$\times$	$\sqcup$	$\sqcup$	$\cdots$
$\cdots$	$\sqcup$	$\sqcup$	a	$\times$	$\times$	$\sqcup$	$\sqcup$	$\cdots$
$\cdots$	$\sqcup$	$\sqcup$	a	b	$\times$	$\sqcup$	$\sqcup$	$\cdots$
$\cdots$	$\sqcup$	$\sqcup$	a	b	a	$\sqcup$	$\sqcup$	$\cdots$

Ein Fragezeichen (?) bedeutet, dass die entsprechende Nachbarzelle nicht beachtet wird, und es werden nur diejenigen Übergänge angegeben, die den Zustand der Zelle verändern.

a) [2] Hat die angegebene Ausführung terminiert?

b) [3] In welcher Konfiguration terminiert der Zellularautomat, wenn er in der folgenden Konfiguration gestartet wird?

$\cdots$	$\sqcup$	$\sqcup$	$\times$	$\times$	$\sqcup$	$\sqcup$	$\cdots$

c) [4] Beschreiben Sie in einfachen Worten die Konfiguration, in der die Ausführung terminiert, wenn die Eingabe für den angegebenen Zellularautomaten aus n aufeinander folgenden $\times$-Zuständen besteht.

d) [3] Eine Zelle z heisst *wach*, wenn $c(z) \neq \sqcup$. Wie viele Zellen können in einem Zellularautomaten nach t Schritten höchstens wach sein, wenn die Startkonfiguration aus k zusammenhängenden wachen Zellen besteht? Begründen Sie Ihre Antwort.

3.2 Zellularsprachen (45 Punkte)

Man kann Zellularautomaten wie folgt verwenden, um eine formale Sprache $\mathcal{L}$ über dem Alphabet Σ zu erkennen, wenn Σ vollständig in der Zustandsmenge Q des Zellularautomaten enthalten ist und $\varepsilon \notin \mathcal{L}$. Der Zellularautomat wird in einer Konfiguration gestartet, in der das zu erkennende Wort $w \in \Sigma^+$ beidseitig von $\llcorner$-Zuständen umgeben ist. Der Zellularautomat *erkennt* $\mathcal{L}$, falls er, wenn er auf diese Weise gestartet wird, in einer Konfiguration c terminiert, für die gilt:

- Es gibt eine Zelle z mit $c(z) = J$, falls $w \in \mathcal{L}$ und $c(z) = N$, falls $w \notin \mathcal{L}$, und
- es gilt $c(z') = \llcorner$ für alle übrigen Zellen $z' \neq z$.

Hinweis: Fassen Sie ähnliche Zustandsübergänge geeignet zusammen, um Zeit und Platz zu sparen, und verwenden Sie wie in Aufgabe 3.1 Fragezeichen (?), wenn bestimmte Nachbarzellen nicht beachtet werden müssen. Achten Sie jedoch darauf, dass die Zustandsübergänge eindeutig spezifiziert sind! Werte, für die δ den Zustand einer Zelle nicht ändert, müssen nicht angegeben werden.

a) [7] Geben Sie einen Zellularautomaten an, der $\mathcal{L} = \{w \in \{a, b, c\}^+ : |w| \text{ ist durch 3 teilbar}\}$ erkennt. Geben Sie die Schritte an, die Ihr Automat für das Wort $w = bcb$ ausführt. Verwenden Sie die gleiche Notation wie im Beispiel aus Aufgabe 3.1.

b) [10] Geben Sie ein Verfahren an, mit dem ein Zellularautomat erstellt werden kann, der eine gegebene reguläre Sprache $\mathcal{L}_{\text{reg}}$ über dem Alphabet Σ erkennt, wenn $\varepsilon \notin \mathcal{L}_{\text{reg}}$. Geben Sie die dazu die Zustandsmenge Q und die Zustandsübergangs-Funktion δ an. Bedenken Sie, dass die Startkonfiguration vorgegeben ist.

c) In dieser Aufgabe soll schließlich eine Turing-Maschine $(Q_T, \Sigma, \Gamma, \delta_T, q_0, q_A, q_R)$ mit Hilfe eines Zellularautomaten simuliert werden. Das Ziel ist die Konstruktion eines Zellularautomaten, welcher die gleiche Sprache wie die angegebene Turing-Maschine erkennt.

(i) [8] Beschreiben Sie in klaren Worten Ihre Idee und begründen Sie, warum sie funktioniert. Gehen Sie dabei auch auf die Terminierung der Turing-Maschine und des Zellularautomaten ein.

(ii) [20] Geben Sie die Zustandsmenge Q für einen Zellularautomaten an, der die Turingmaschine simuliert. Erklären Sie in Worten die Funktion der Zustände. Listen Sie dann die zugehörige Zustandsübergangs-Funktion δ auf, und geben Sie an, wozu die jeweiligen Übergänge dienen. Bedenken Sie, dass die Startkonfiguration vorgegeben ist.

4 Memory (50 Punkte)

Das populäre Gesellschaftsspiel Memory kann wie folgt beschrieben werden: Es gibt $2n$ Karten, die auf der Vorderseite mit den Zahlen $1, 1, 2, 2, 3, 3, \ldots, n-1, n-1, n, n$ beschrieben sind. Auf der Rückseite sehen alle Karten identisch aus. Die Karten werden gemischt und verdeckt vor einer Spielerin abgelegt, so dass die Zahlen nicht sichtbar sind. Ein Spielzug besteht darin, dass zuerst eine gewählte Karte aufgedeckt wird und dann eine zweite Karte zum Aufdecken ausgewählt wird. Zeigen beide Karten die gleiche Zahl an, dann wird das Kartenpaar entfernt, andernfalls werden beide Karten wieder verdeckt. Ziel des Spieles ist es, in möglichst wenigen Zügen alle Kartenpaare zu entfernen. Nehmen Sie weiter an, dass der optimale Offline-Spieler alle Karten kennt, das heisst, mit aufgedeckten Karten spielt. Wir nehmen an, dass die Spielerin sich an alles erinnern kann.

a) [7] Beschreiben Sie eine Strategie, die im Worst-Case höchstens 2–kompetitiv ist.

b) [10] Geht es deterministisch überhaupt besser als 2–kompetitiv? Zeigen Sie wie. Weisen Sie auch eine möglichst gute untere Schranke für deterministische Strategien nach.

c) [10] Was ist, wenn man nicht den Worst-Case wie in **a)** betrachtet, sondern den Erwartungswert der benötigten Spielzüge? Beschreiben Sie dazu eine entsprechende Strategie (im Worst-Case höchstens 2–kompetitiv) und zeichnen Sie die dazugehörige Markovkette für $n = 3$ auf.

d) [10] Betrachten Sie Ihre Strategie aus Aufgabe **c)** für beliebiges $n \gg 3$. Erklären Sie überzeugend, warum Ihre Strategie im Erwartungswert asymptotisch besser ist als 2–kompetitiv.

e) [10] Das Erinnerungsvermögen der Spielerin ist nun beschränkt, sie kann sich nur an maximal k Karten erinnern, wobei k eine Konstante ist und $k \ll n$ gilt. Wenn Sie eine neue Karte aufdeckt, und ihr Kartenspeicher schon voll ist, so kann sie sich nach dem Spielzug nur an die neue Karte erinnern, wenn sie eine Karte aus dem Speicher löscht. Welche Kompetitivität kann eine deterministische Strategie hier erreichen?

f) [3] Welche erwartete Anzahl von Spielzügen braucht eine randomisierte Strategie, bei der überhaupt kein Kartenspeicher existiert: Ziehe zufällig gleichverteilt zwei Karten, bis alle Paare aufgedeckt sind.