Leistungsnachweis in

Rechnerkommunikation

Sommersemester 2013
27. September 2013

Name: _____
Matrikelnummer: _____
Geburtsdatum: _____
Studienfach: _____
Fachsemester: _____
Angemeldet über: ◯ Mein Campus ◯ Lehrstuhl

- Bitte verwenden Sie einen blauen oder schwarzen Kugelschreiber (kein rot, keinen Bleistift).
- Schriftliche Aufzeichnungen (sowohl eigene Aufzeichnungen wie auch Bücher) sind als Hilfsmittel zugelassen. Auch ein Taschenrechner ist erlaubt und hilfreich. Nicht zugelassen sind dagegen Computer, PDAs, Mobiltelefone und sonstige Kommunikationsmittel.
- Legen Sie den Ausweis (mit Lichtbild) griffbereit auf den Platz.
- Bitte überprüfen Sie, ob Sie alle 15 Blätter erhalten haben.
- Schreiben Sie die Antworten jeweils in den freien Raum hinter den Fragen. Sollte dieser nicht ausreichen, steht noch freier Raum am Ende der Klausur zur Verfügung. Bitte kennzeichnen Sie dort deutlich, welche Aufgabe Sie bearbeiten. Gesondert beigelegte Blätter werden nicht bewertet!
- Schreiben Sie deutlich! Unleserliche Antworten können nicht in die Bewertung eingehen!
- Die Arbeitskopien können der Klausur entnommen werden und müssen nicht mit abgegeben werden.

Ich habe die Hinweise auf dieser Seite zur Kenntnis genommen und alle 15 Blätter der Klausur erhalten:

_____
Unterschrift

Bewertung:	1	2	3	4	Σ

1 Fragenkatalog (26 Punkte gesamt)

1.1 Allgemeine Fragen (5+3+2+2 Punkte)

a) Listen Sie die 5 Schichten des Internet-Protokollstapels von oben nach unten auf. Geben Sie für jede Schicht ein Beispiel (Protokoll, Verfahren, oder Anwendung) an.

b) Ein Client möchte eine TCP-Verbindung mit einem Server aufbauen. Beschreiben Sie kurz den Verbindungsaufbau des TCP-Protokolls, und erwähnen Sie, welche Flags in jedem Segment gesetzt werden.

c) Geben Sie eine Anwendung an, bei der das *CSMA*-Verfahren Vorteile gegenüber dem *ALOHA*-Verfahren hat. Nennen Sie eine weitere Anwendung bei der das *ALOHA*-Verfahren gegenüber dem *CSMA*-Verfahren Vorteile hat. Begründen Sie kurz ihre Antworten.

d) Geben Sie je einen Vorteil und einen Nachteil der Manchester-Codierung gegenüber der *NRZ*-Codierung an.

1.2 E-Mail (2 + 2 Punkte)

a) Listen Sie drei wichtige Anwendungsschicht-Protokolle auf, die bei der Übertragung von E-Mails verwendet werden.

b) Wieso sollte eine ASCII-Textdatei beim Übertragen mittels *SMTP* kodiert werden. Wo treten ohne Kodierung Probleme auf?

1.3 Latenzzeit (2 + 2 + 3 + 3 Punkte)

Host A möchte ein Objekt O an Host B senden. Zwischen Host A und Host B liegt ein weiterer *Store-and-Forward*-Host M. Eine 30 km lange Glasfaserleitung verbindet jeweils Host A mit Host M und Host M mit Host B (Netztopologie: $A \leftrightarrow M \leftrightarrow B$). Informationen breiten sich auf den Leitungen mit einer Geschwindigkeit von $v = 3 \times 10^8\,\mathrm{m\,s^{-1}}$ aus. Die Paketgröße beträgt $L = 1\,\mathrm{kB}$. Die Übertragungsrate jedes Links beträgt $R = 8\,\mathrm{Mbit\,s^{-1}}$. (Vereinfachungen: $1\,\mathrm{kB} = 10^3\,\mathrm{B}$. Es gibt keine Fehler und es weden keine Pakete verworfen.)

a) Bestimmen Sie die Ausbreitungsverzögerung (d_{prop}) eines Links.

b) Bestimmen Sie die Übertragungsverzögerung (d_{trans}) eines Pakets auf einem Link.

c) Wie lange dauert es (d_1) ein Objekt der Größe $O = 1\,\text{kB}$ von A nach B zu übertragen?

d) Wie lange dauert es (d_2) ein Objekt der Größe $O = 3.5\,\text{kB}$ von A nach B zu übertragen?

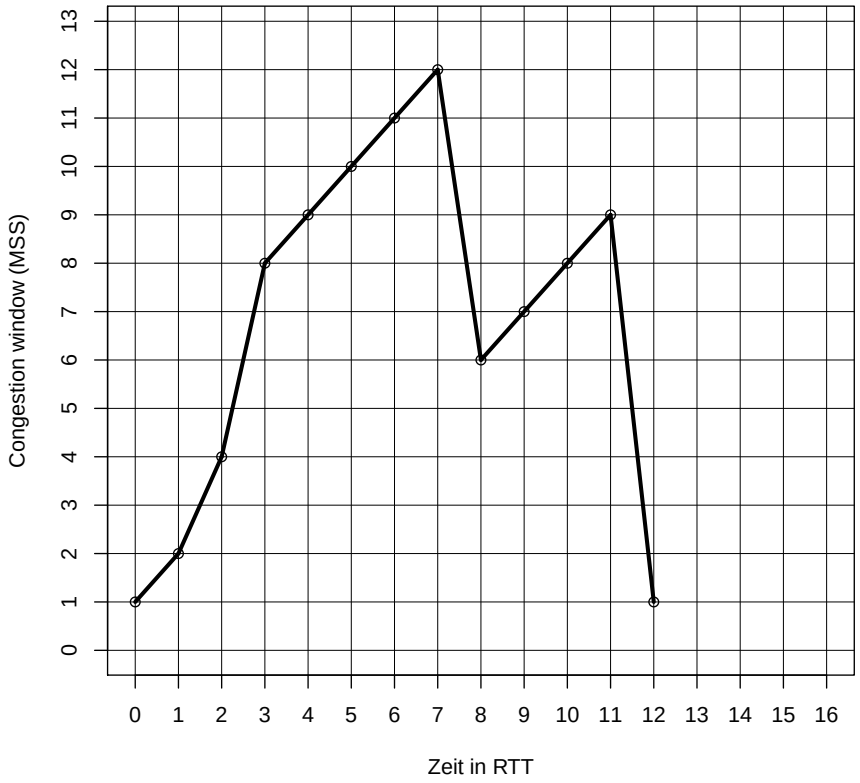

Abbildung 1: TCP-Fenstergröße als eine Funktion der Zeit

2 Transportschicht (26 Punkte gesamt)

2.1 TCP-Überlastkontrolle (7×2 Punkte)

Abbildung 1 stellt den Verlauf des *Congestion Windows* über die Zeit in *Round Trip Times* (RTT) dar. Nehmen Sie an, dass sich im Zustand des *Additive Increase* das *Congestion Window* um exakt eine *Maximum Segment Size* (MSS) pro RTT erhöht. Weiterhin hat der Threshold einen endlichen Initialwert.

a) Identifizieren Sie die Zeitintervalle, in denen TCP Slow Start in Betrieb ist.

b) In Abbildung 1 ist nach der $t = 7$ RTT ein Fehler aufgetreten. Wie wurde dieser erkannt? Begründen Sie Ihre Antwort.

c) In Abbildung 1 ist nach der $t = 11$ RTT ein Fehler aufgetreten. Wie wurde dieser erkannt? Begründen Sie Ihre Antwort.

d) Welchen Wert hat der Threshold zum Zeitpunkt $t = 2$ RTT? Begründen Sie ihre Antwort kurz.

e) Welchen Wert hat der Threshold zum Zeitpunkt $t = 9$ RTT? Begründen Sie ihre Antwort kurz.

f) Wurde vor der Zeit $t = 12$ RTT das 36. Segment mit Daten bereits verschickt? Falls ja, zu welcher RTT-Zeit?

g) Nehmen Sie nun an, dass keine weiteren Fehler auftreten. Zeichnen Sie das Kurvenverhalten für den Zeitraum $t \in (12, 16]$ RTT in Abbildung 1 ein.

2.2 TCP-Leistungsanalyse (12 Punkte)

2.2.1 Dynamisches Fenster (2+2+2 Punkte)

Betrachten Sie eine TCP-Verbindung mit dynamischen Fenstern. Gegeben sind folgende Größen. Die *Round Trip Times* (RTT) beträgt $RTT = 2\,\mathrm{s}$. Das Objekt O_1 hat eine Größe von 1 MB. Ein Segment L hat eine Größe von $L = 1\,\mathrm{kB}$. Die Bandbreite R der Verbindung beträgt $R = 100\,\mathrm{kbit\,s^{-1}}$. (Vereinfachung: $1\,\mathrm{MB} = 10^3\,\mathrm{kB} = 10^6\,\mathrm{B}$)

a) Bis zu welchem Fenster Q treten bei einem unendlich großen Objekt O_∞ Wartezeiten auf?

b) Wie viele Fenster K werden benötigt, um das Objekt O_1 zu übertragen?

c) Wie viele Wartezeiten P treten bei der Übertragung des Objekts O_1 auf?

2.2.2 HTTP-Antwortzeit (2+2+2 Punkte)

Eine Basis HTML-Datei mit 40 eingebetteten Bildern soll mittels HTTP übertragen werden. Gehen Sie vereinfachend davon aus, dass der Server immer kontinuierlich senden kann. D.h., es entstehen keine Slow-Start-Wartezeiten. Wie viele Wartezeiten in RTT sind dennoch erforderlich, bis die Basis-Datei und die Bilder unter den folgenden Varianten auf dem Client verfügbar sind?

a) Nicht-persistentes HTTP

b) Persistentes HTTP mit Pipelining

c) Nicht-persistentes HTTP mit 10 parallelen Verbindungen

3 Mail-Client (25 Punkte)

Implementieren Sie ein **vollständiges** Programm in JAVA, dass die Anzahl der auf einem POP3-Server befindlichen E-Mails erfragt und diese Information per Mail weiterleitet. Die Rückantworten der beteiligten Server sollen überprüft werden.

- Der POP3-Server liegt an `mail.rk.org`:110 (Benutzer: `rk`, Passwort: `rK!23`)
- Als SMTP-Server verwenden Sie: `cipmail.cs.fau.de`:25
- Die Nachricht an `mailCheck@cs.fau.de` soll lauten: `RK hat XX E-Mails`. Wobei `XX` die Anzahl der auf dem RK-Konto befindlichen Mails darstellt. Die E-Mail-Adresse des Absenders soll lauten: `status@rk.org`.

Um die Korrektheit der Rückantworten der Server zu überprüfen können Sie den dreistelligen numerischen Code (z.B. 220, 250, ...) bzw. `+OK`, der zu Beginn jeder Zeile des Servers gesendet wird, verwenden. Kommt eine falsche Antwort von einem der Server, solll die Methode `exit()` aufgerufen werden.Vervollständigen Sie den Code in Listing 3, indem Sie die beiden Methoden `void sendEmail(int count)` und `int getEmailsCount()` implementieren. Geöffnete Verbindungen müssen wieder geschlossen werden. Exceptions müssen berücksichtigt, aber nicht behandelt werden.

Tipp: Beachten Sie die beigefügten Beispiele in Listing 1 und Listing 2.

Listing 1: POP3 Beispiel

```
S: +OK POP3 server ready
C: user bob
S: +OK
C: pass hungry
S: +OK user successfully logged on
C: list
S: 1 498
S: 2 912
S: 3 812
S: .
C: retr 1
S: <Inhalt Nachricht 1>
C: dele 1
S: +OK message marked for delete
C: quit
S: +OK POP3 server signing off
```

Listing 2: SMTP Beispiel

```
S: 220 hamburger.edu
C: HELO crepes.fr
S: 250 Hello crepes.fr, pleased to meet you
C: MAIL FROM: <alice@crepes.fr>
S: 250 alice@crepes.fr... Sender ok
C: RCPT TO: <bob@hamburger.edu>
S: 250 bob@hamburger.edu ... Recipient ok
C: DATA
S: 354 Enter mail, end with "." on a line by itself
C: How are you?
C: .
S: 250 Message accepted for delivery
C: QUIT
S: 221 hamburger.edu closing connection
```

Listing 3: Code

```
import java.io.*;
import java.net.*;
public class RKMailClient {
    public static void main(String[] args) throws Exception {

        sendEmail(getEmailsCount());

        }

        public static void exit() {
          System.out.println("Fehler aufgetreten");
          System.exit(-1);
        }
```

Code (fortgesetzt)

4 Routingverfahren (23 Punkte gesamt)

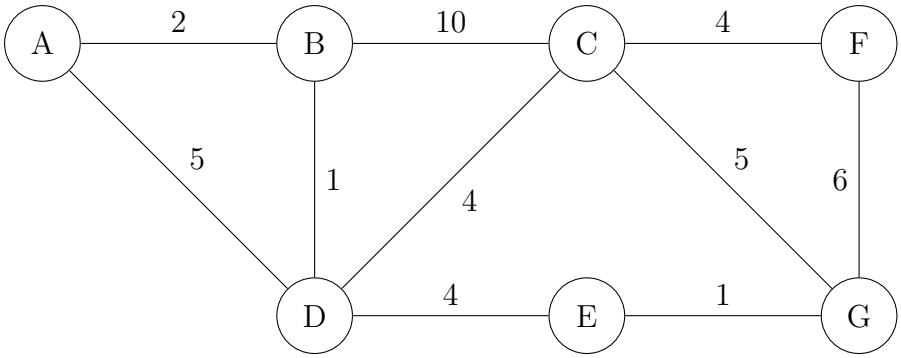

Abbildung 2: Netzwerktopologie für Dijkstra-Verfahren

4.1 Dijkstra-Verfahren (9 Punkte)

Abbildung 2 zeigt eine Netzwerktopologie mit sieben Knoten. Führen Sie aus Sicht des Knotens A das Dijkstra-Verfahren in der Tabelle 1 durch, um minimale Pfade zwischen den Knoten zu erhalten. Verwenden Sie bitte die gleiche Notation wie in der Vorlesung.

S	N'	$D(B), p(B)$	$D(C), p(C)$	$D(D), p(D)$	$D(E), p(E)$	$D(F), p(F)$	$D(G), p(G)$
1							
2							
3							
4							
5							
6							
7							
8							

Tabelle 1: Dijkstra-Verfahren für Knoten A

4.2 Minimal aufspannender Baum (6 Punkte)

Zeichnen Sie auf Basis von Tabelle 1 (Dijkstra-Verfahren für Knoten A) die daraus resultierende minimale Baumstruktur in Abbildung 3 ein.

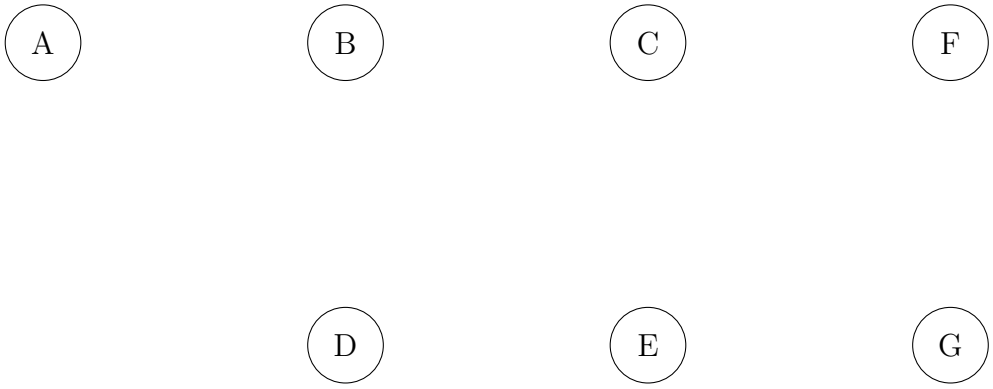

Abbildung 3: Minimaler aufspannender Baum

4.3 Poisoned Reverse (4+4 Punkte)

Poisoned Reverse ist ein Verfahren zur Vermeidung von Routingschleifen beim Distanz-Vektor-Routing.

a) Tabelle 2 zeigt die konvergierten Routing-Informationen für das Netzwerk in Abbildung 4. Nehmen Sie an, dass nun die Kosten der Kante zwischen den Knoten Z und Y auf $c(Z, Y) = 10$ erhöht werden. Im Folgenden soll ausschließlich der kürzeste Weg zu Knoten W betrachtet werden. Zeigen Sie in Tabelle 3, wie sich die Routing-Informationen **mit** Poisoned Reverse an den beteiligten Knoten ändern bis erneute Konvergenz erreicht wird.

Die Tabellen stellen einen zeitlichen Verlauf dar. Beispiel: In Tabelle 3 aktualisiert im ersten Schritt der Knoten Y seine Routing-Information zu Knoten W auf die Kosten 20 über Knoten Z.

b) Nehmen Sie unabhängig von der Teilaufgabe a) nun an, dass die Kosten der Kante zwischen den Knoten Z und W in Abbildung 4 auf $c(W, Z) = 40$ erhöht worden sind ($c(Z, Y) = 2$). Zeigen Sie in Tabelle 4, wie sich die Routing-Informationen **mit** Poisoned Reverse an den beteiligten Knoten ändern, bis erneute Konvergenz erreicht wird.

Bewerten Sie die Wirksamkeit von Poisoned Reverse in dieser Situation.

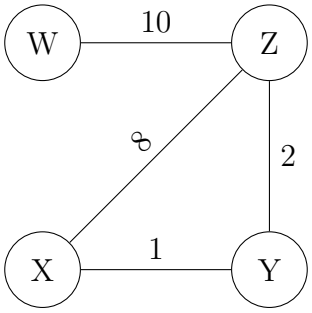

Abbildung 4: Netzwerktopologie für Poisoned Reverse

von Z zu	$D_Z(\cdot)$	$nh_Z(\cdot)$
W	10	W

von X zu	$D_X(\cdot)$	$nh_X(\cdot)$
W	13	Y

von Y zu	$D_Y(\cdot)$	$nh_Y(\cdot)$
W	12	Z

Tabelle 2: Routing Informationen für den kürzesten Weg zu W in Abbildung 4 **vor** der Änderung der Kosten (also mit: $c(Z, Y) = 2$)

Die Änderung der Routing Informationen über die Zeit für den kürzesten Weg zu W:

Zeit	Von ... zu W	$D_{...}(W)$	$nh_{...}(W)$
t_0	Y	20	Z
t_1			
t_2			
t_3			
t_4			
t_5			
t_6			
t_7			
t_8			
t_9			

Tabelle 3: Mit Poisoned Reverse
a) $c(Z, Y) = 10$

Time	Von ... zu W	$D_{...}(W)$	$nh_{...}(W)$
t_0			
t_1			
t_2			
t_3			
t_4			
t_5			
t_6			
t_7			
t_8			
t_9			

Tabelle 4: Mit Poisoned Reverse
b) $c(Z, Y) = 2$, $c(W, Z) = 40$

Zusatzblatt