Leistungsnachweis in

Rechnerkommunikation

Sommersemester 2010
4. Oktober 2010

Name: _____
Matrikelnummer: _____
Geburtsdatum: _____
Studienfach: _____
Fachsemester: _____

- 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 17 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 gehen nicht in die Bewertung ein!
- 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 17 Blätter der Klausur empfangen:

_____
Unterschrift

Bewertung:

1	2	3	4	Σ

1 Allgemeine Fragen (15 Punkte)

Die Fragen können mit Stichpunkten beantwortet werden.

a) [1 Punkt] Was bedeutet QoS?

b) [1 Punkt] Ein Rechner hat die IP-Adresse 192.168.20.5/16 (klassenlose Adressierung). Wie lautet die dazugehörige Subnetzadresse?

c) [1 Punkt] Zur Klassifikation von Kommunikationssystemen soll die Übertragungsart betrachtet werden. Nennen Sie zwei Klassen der Übertragungsrichtung.

d) [3 Punkte] Was bedeutet P2P? Nennen Sie außerdem die Grundidee, eine Eigenschaft, ein Anwendungsgebiet und ein konkretes Beispiel von P2P.

e) [1 Punkt] Was ist die Aufgabe der Sicherungsschicht?

f) [2 Punkte] Beantworten Sie jeweils für Flusskontrolle und Überlastkontrolle: Wer wird vor Überlast geschützt? Wer wird kontrolliert?

g) [3 Punkte] Was ist Modulation? Auf welcher Schicht des Internetprotokollstapels ist Modulation einzuordnen? Nennen Sie eine Modulationsart und hierfür ein Beispiel.

h) [3 Punkte] Bei einem Experiment sollen Daten im Meer via Schall übertragen werden. Knoten A und B sind $150\,m$ von einander entfernt. Die Ausbreitungsgeschwindigkeit auf dem Link beträgt $1{,}5\,km/s$. A sendet B $1{,}0\,kByte$ mit einer Datenrate von $80\,kbit/s$. Es treten keine weiteren Verzögerungen auf. Berechnen Sie eine erste Abschätzung der Übertragungszeit d für diese Annahmen.

2 Socket-Programmierung (30 Punkte)

Programmieren Sie einen vereinfachten POP3-Client in JAVA, der eine Verbindung zu einem POP3-Server aufbauen, sich authentifizieren und die Anzahl der vorhandenen E-Mails überprüfen kann. Leiten Sie hierzu die Klasse `Pop3` von `Base` ab und implementieren Sie die abstrakten Methoden. Halten Sie sich an das in Abbildung 1 gezeigte Verhalten. Gehen Sie davon aus, dass die Methoden `init()` und `test()` von außen aufgerufen werden, d.h., Sie benötigen kein `main()`. Nehmen Sie außerdem an, dass `BufferdReader in` und `BufferdWriter out` bei `authentificate();` und `countMails();` geöffnet sind. Exceptions müssen berücksichtigt, aber nicht behandelt werden. Berücksichtigen Sie die Konstanten, Variablen, Methoden und Kommentare von `Base`:

```
import java.io.*;     import java.net.*;

abstract class Base {
   abstract void init() throws Exception; // siehe Statechart!
   abstract void test() throws Exception; // siehe Statechart!

   // Authentifizierung beim POP3-Server. erfolgreich --> true, sonst false
   abstract boolean authentificate() throws Exception;
   // Frägt die Anzahl der E-Mails beim POP3-Server ab
   abstract int countMails() throws Exception;

   // Verwenden Sie diese Konstanten in den dahinter stehenden Methoden;
   final static String POP3_SERVER = "192.168.5.122";// für init();
   final static String USER        = "user";         // für authentificate();
   final static String PASSWORD    = "geheim";       // für authentificate();

   // Verwenden Sie diese Variablen für die Kommunikation mit dem POP3-Server
   Socket socket;         // für init();
   BufferedReader in;     // für init(); authentificate(); countMails();
   BufferedWriter out;    // für init(); authentificate(); countMails();

   // Verwenden Sie diese Variablen und Methoden wie im Statechart
   boolean authOK = false;      // Authentifizierung erfolgreich? ja --> true
   void print(int num){}        // Ausgabe einer Zahl
   void end() {}                // Beenden des POP3-Clients
}
```

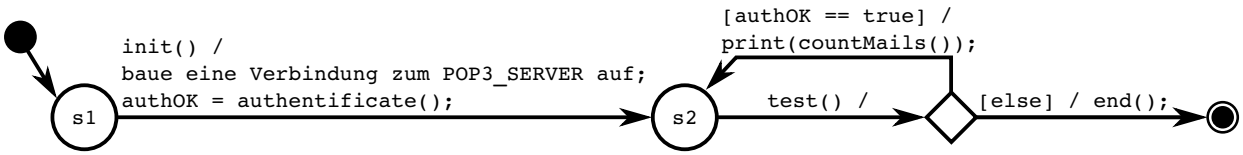

Abbildung 1: Vollständiges Statechart eines vereinfachten POP3-Clients

3 POP3-Leistungsanalyse (29 Punkte)

Ein Programm auf Host C holt die erste E-Mail aus dem Postfach von Host S ab. Dazu baut das Programm eine POP3-Verbindung auf (gemäß Spezifikation), ruft die erste E-Mail im Postfach ab und beendet die Verbindung anschließend wieder. Der für die Authentifizierung benötigte Benutzername ist: *user*. Das Passwort lautet: *geheim*.

Folgende Annahmen treffen immer zu: Es kann davon ausgegangen werden, dass sich immer mindestens eine E-Mail auf dem Server befindet (kein *list* Befehl notwendig). Bei der Übertragung treten keine Fehler auf. Die Verbindung zwischen Host C und Host D ist nicht symmetrisch (ähnlich wie zu einer DSL-Verbindung). In beiden Hosts treten keine Verzögerungen auf. Weiterhin gelten die Konstanten gemäß Tabelle 1.

Ausbreitungsverzögerung:	$d_{prop,SC} = 10\,\text{ms}$ (Host S $\rightarrow$ Host C) $d_{prop,CS} = 100\,\text{ms}$ (Host C $\rightarrow$ Host S)
Raten:	$R_{SC} = 10\ \frac{Mbit}{s}$ (Host S $\rightarrow$ Host C) $R_{CS} = 1\ \frac{Mbit}{s}$ (Host C $\rightarrow$ Host S)
Paketlänge:	$L = 1500\ Byte\ (= MSS)$

Tabelle 1: Konstanten

3.1 Round Trip Time [3 Punkte]

Gibt es auf Grund der asymmetrischen Leitung ausgehend von Host C bzw. von Host S verschiedene Round Trip Times (RTT)? Begründen Sie Ihre Antwort und berechnen Sie die RTT.

3.2 Kommunikations-Verlauf [13 Punkte]

Tragen Sie in Abbildung 2 den Ablauf für den Paketaustausch zwischen Host C und Host S auf TCP- und Anwendungsebene ein. Setzen Sie für jedes gesendete Paket die passenden TCP-Flags (keine Sequenz- und Acknowledgement-Nummern) und zeigen Sie an, ob Daten im Paket transportiert werden. Annotieren Sie zusätzlich die übertragenen POP3-Befehle. Vereinfachend wird hier davon ausgegangen, dass die POP3-Befehle eine Größe von 0 Bytes aufweisen (keine Daten). Einzig die übertragene E-Mail hat ein von 0 verschiedenes Datenvolumen. Die Abkürzung CW in Abbildung 2 steht für *Congestion Window* und wird mit 1 *Maximum Segment Size* (MSS) initialisiert (CW = 1 MSS). Nennen Sie weiterhin den Wert des CW nach jedem Paket das bei Host S eingetroffen ist (Berechnung gemäß TCP-Spezifikation aus der Vorlesung). Gehen Sie davon aus, dass das *Advertized Window* von Host C immer groß genug ist. Zeichnen Sie das Diagramm, bis eine E-Mail der Größe $O = 9000 Bytes$ vollständig übertragen ist.

Tipp: Beachten Sie den TCP-Slow-Start.

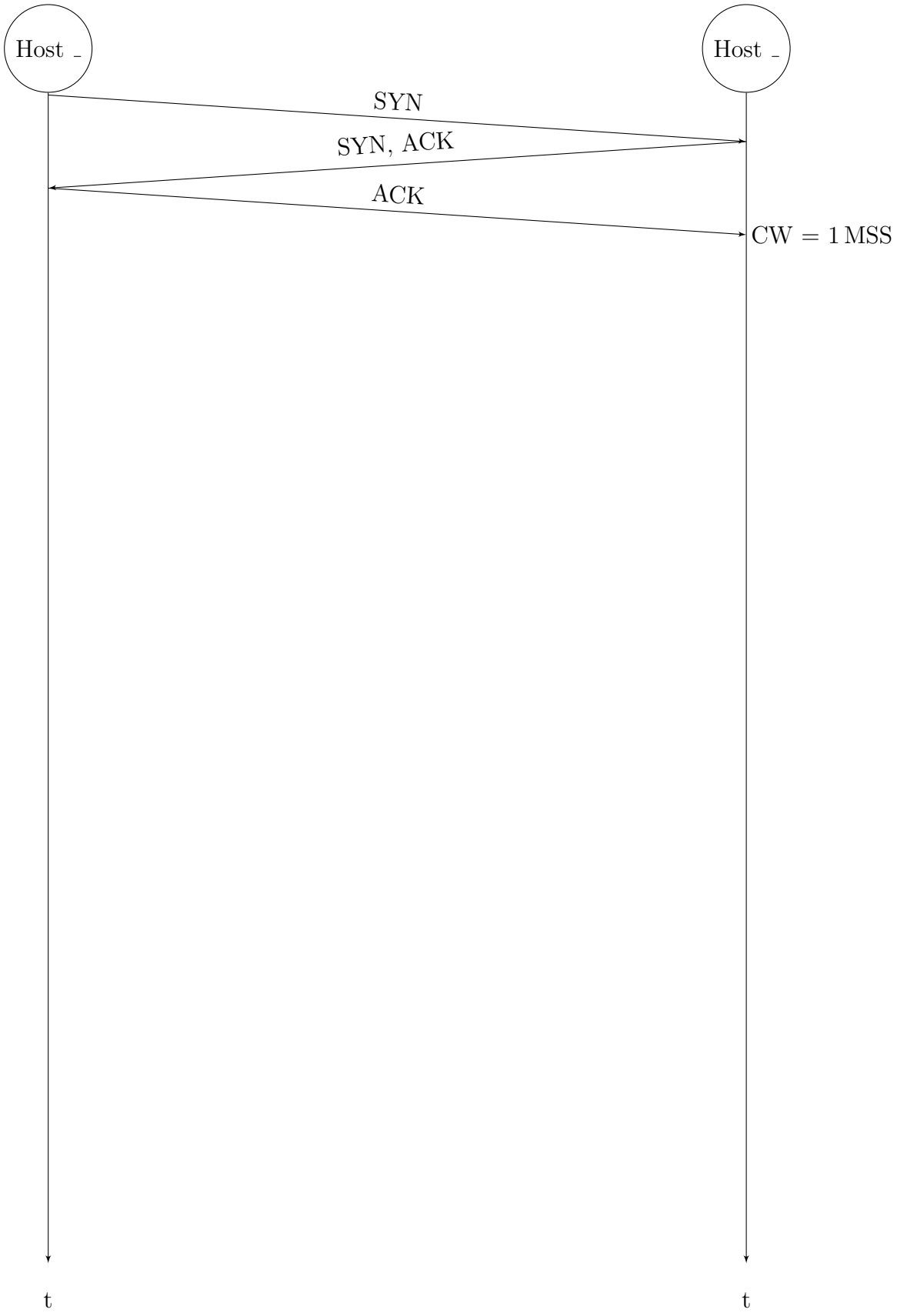

Abbildung 2: Ablaufdiagramm für eine TCP/POP3-Verbindung

3.3 Leistungsanalyse [13 Punkte]

Im Folgenden soll die Verzögerungszeit zwischen Verbindungsaufbau und vollständigem Eintreffen der ersten E-Mail der Größe O bei Host C ermittelt werden. Wie bereits in Aufgabe 3.2 angenommen, können Sie davon ausgehen, dass nur Pakete, die die eigentliche E-Mail enthalten, eine Größe verschieden von 0 aufweisen.

Modifizieren Sie die aus der Vorlesung bekannte Formel für die TCP-Latenzberechnung mit Slow-Start (1) so, dass die POP3-Anwendung (wie in Aufgabe 3 spezifiziert) berücksichtigt wird. Berücksichtigen Sie auch die für die Berechnung benötigten Wartezeiten $P = min(Q, K-1)$, wobei Q die Anzahl der Wartezeiten für ein unendlich großes Objekt und K die Anzahl der Sendefenster darstellt. Vereinfachen Sie die Formeln für Q und K (geometrische Reihe: $\sum_{k=0}^{m} a_0 q^k = a_0 \frac{q^{m+1}-1}{q-1}$). Welche Rate aus Tabelle 1 muss für die Formel 1 verwendet werden?

Berechnen Sie damit die Latenzzeit, die benötigt wird, um ein E-Mail der Größe $O = 9000\,\text{Byte}$ zu übertragen (Zeit bis die Daten verfügbar sind). Die benötigten Konstanten entnehmen Sie bitte Tabelle 1 sowie der Aufgabe 3.1.

Tipp: Das erste Sendefenster der E-Mail hat eine Größe verschieden von 1 MSS.

$$d = 2RTT + \frac{O}{R} + P\left(RTT + \frac{L}{R}\right) - \left(2^P - 1\right)\frac{L}{R} \tag{1}$$

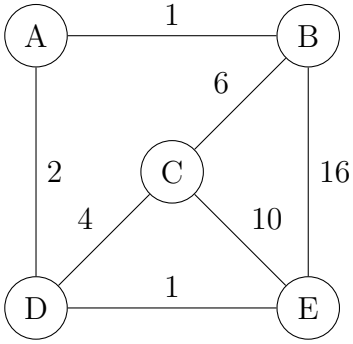

Abbildung 3: Netzwerk

4 Routingverfahren (16 Punkte)

Abbildung 3 zeigt eine Netzwerktopologie mit fünf Knoten. Simulieren Sie das Dijkstra-Verfahren in Tabelle 2 und den Forward-Search-Algorithmus in Tabelle 3, um minimale Pfade zwischen den Knoten zu erhalten. Verwenden Sie für das Dijkstra-Verfahren Knoten D und für den Forward-Search-Algorithmus den Knoten B als Startknoten. Vervollständigen Sie die Forwarding-Tabelle 4 auf Basis von Tabelle 3 und dem Forward-Search-Algorithmus.

Bitte benutzen Sie dieselben Symbole wie in der Vorlesung.

Schritt	V'	$D(A), p(A)$	$D(B), p(B)$	$D(C), p(C)$	$D(E), p(E)$
0					
1					
2					
3					
4					
5					
6					

Tabelle 2: Dijkstra-Verfahren für Knoten **D**

Schritt	bestätigte Liste	vorläufige Liste
0		
1		
2		
3		
4		
5		
6		

Tabelle 3: Forward-Search-Algorithmus für Knoten **B**

Ziel	Link
A	
C	
D	
E	

Tabelle 4: Forwarding-Tabelle für Knoten **B**

Zeichnen Sie auf Basis von Tabelle 2 (Dijkstra-Verfahren mit Startknoten D) die daraus resultierende minimale Baumstruktur in Abbildung 4.

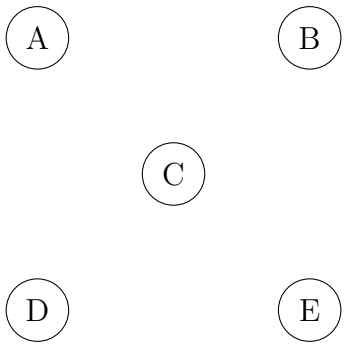

Abbildung 4: Minimal aufgespannter Baum

Wurde in beiden Teilaufgaben (Dijkstra-Verfahren und Forward-Search-Algorithmus) der gleiche Baum aufgespannt? Begründen Sie Ihre Antwort.

Zusatzblatt

Zusatzblatt

Arbeitskopie: Socket-Programmierung

Programmieren Sie einen vereinfachten POP3-Client in JAVA, der eine Verbindung zu einem POP3-Server aufbauen, sich authentifizieren und die Anzahl der vorhandenen E-Mails überprüfen kann. Leiten Sie hierzu die Klasse `Pop3` von `Base` ab und implementieren Sie die abstrakten Methoden. Halten Sie sich an das in Abbildung 5 gezeigte Verhalten. Gehen Sie davon aus, dass die Methoden `init()` und `test()` von außen aufgerufen werden, d.h., Sie benötigen kein `main()`. Nehmen Sie außerdem an, dass `BufferdReader in` und `BufferdWriter out` bei `authentificate();` und `countMails();` geöffnet sind. Exceptions müssen berücksichtigt, aber nicht behandelt werden. Berücksichtigen Sie die Konstanten, Variablen, Methoden und Kommentare von `Base`:

```
import java.io.*;     import java.net.*;

abstract class Base {
   abstract void init() throws Exception; // siehe Statechart!
   abstract void test() throws Exception; // siehe Statechart!

   // Authentifizierung beim POP3-Server. erfolgreich --> true, sonst false
   abstract boolean authentificate() throws Exception;
   // Frägt die Anzahl der E-Mails beim POP3-Server ab
   abstract int countMails() throws Exception;

   // Verwenden Sie diese Konstanten in den dahinter stehenden Methoden;
   final static String POP3_SERVER = "192.168.5.122";// für init();
   final static String USER        = "user";         // für authentificate();
   final static String PASSWORD    = "geheim";       // für authentificate();

   // Verwenden Sie diese Variablen für die Kommunikation mit dem POP3-Server
   Socket socket;         // für init();
   BufferedReader in;     // für init(); authentificate(); countMails();
   BufferedWriter out;    // für init(); authentificate(); countMails();

   // Verwenden Sie diese Variablen und Methoden wie im Statechart
   boolean authOK = false;      // Authentifizierung erfolgreich? ja --> true
   void print(int num){}        // Ausgabe einer Zahl
   void end() {}                // Beenden des POP3-Clients
}
```

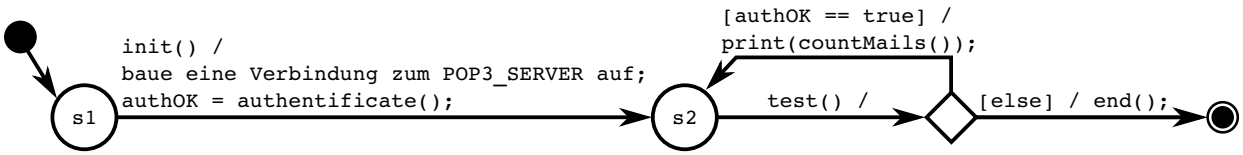

Abbildung 5: Vollständiges Statechart eines vereinfachten POP3-Clients