Leistungsnachweis in

Rechnerkommunikation

Sommersemester 2011
04. Oktober 2011

Name: _____
Matrikelnummer: _____
Geburtsdatum: _____
Studienfach: _____
Fachsemester: _____
Extra Nachweis: ◯ benoteter Schein ◯ unbenoteter Schein

- 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 16 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!

Ich habe die Hinweise auf dieser Seite zur Kenntnis genommen und alle 16 Blätter der Klausur empfangen:

_____
Unterschrift

Bewertung:

1	2	3	4	Σ

1 Allgemeine Fragen (22 Punkte gesamt)

1.1 Leitungskodierung (4 Punkte)

Vervollständigen Sie Abbildung 1.1. Für die Non-Return-To-Zero (NRZ) und für die Manchester-Kodierung tragen Sie die Bitsequenz ein. Bei der differentiellen Manchester-Kodierung (Diff-Manchester) ist der Signalverlauf zu vervollständigen.

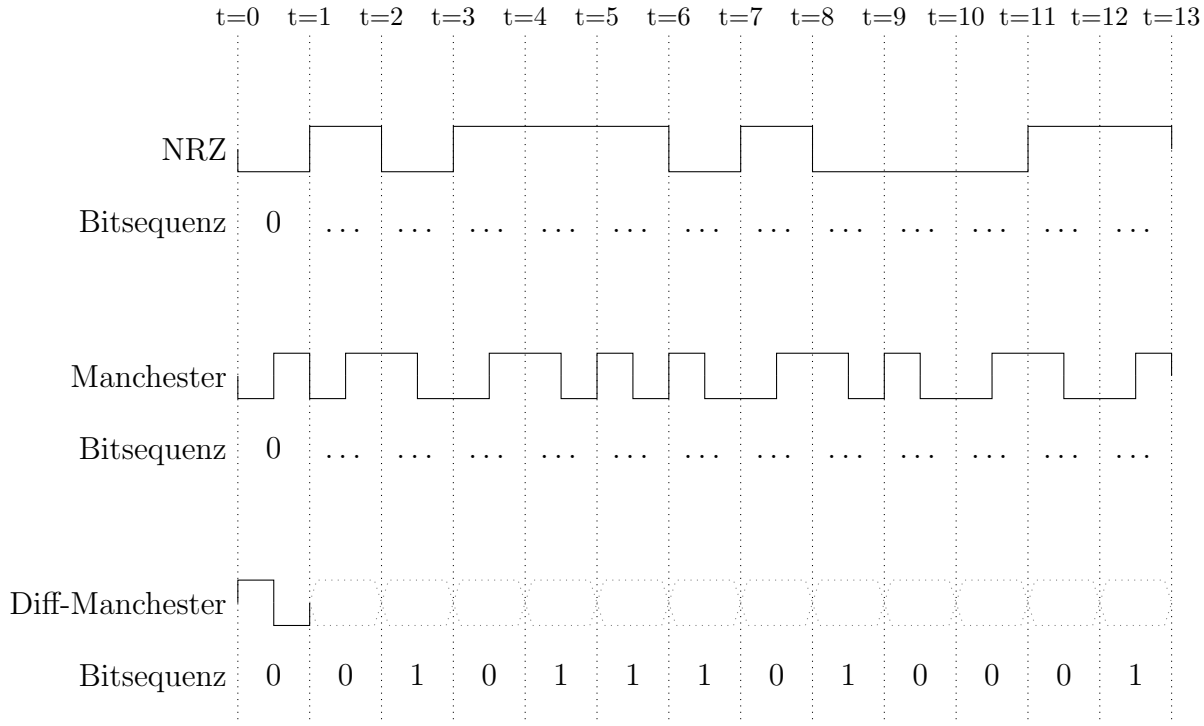

Abbildung 1: Leitungskodierungen

1.2 ALOHA vs. CSMA (3 Punkte)

Unter welchen Bedingungen bietet ALOHA einen höheren Durchsatz als ein CSMA-basiertes Verfahren (grobe ganzzahlige Abschätzung)? Nennen Sie ein Anwendungsbeispiel.

1.3 ISO OSI-Schichtenmodell (4 Punkte)

Welche Informationen werden für die TCP oder UDP Prüfsummenberechnung verwendet? Warum verstößt sie damit gegen die Schichtenkapselung? Nennen Sie die Motivation (Vorteil) hinter diesem Ansatz.

1.4 Subnetze und klassenlose Adressierung (5 Punkte)

Kann der Computer in Tabelle 1 direkt mit dem Webserver (auf Netzwerkschicht) kommunizieren oder benötigt er hierfür einen Router? Nennen Sie die Subnetzadresse in dem sich der Computer befindet. Wie viele Teilnehmer kann es maximal enthalten? (Achtung: alle Host-Bits zu „0“ oder zu „1“ haben Sonderrollen.)

Router	IP:	10.0.1.1/22
	MAC:	00:00:00:00:00:F1
Computer	IP:	10.0.1.2/22
	MAC:	00:00:00:00:00:F2
Webserver	IP:	10.0.3.4/22
	MAC:	00:00:00:00:00:F3

Tabelle 1: Netzwerkinformationen

1.5 Ethernet Rahmen (6 Punkte)

Ein Computer fordert eine Webseite von einem Server an („GET /index.html“; TCP-Port 80). Gehen Sie davon aus, dass bereits alle benötigten Informationen (Netzwerk-Adressen) vorliegen. Zeichnen Sie in Abbildung 2 schematisch den Aufbau des vom Computer zu versendenden Rahmens in dem der HTTP-Request enthalten ist (= drittes Paket des 3-Wege-Handshakes) ein. Markieren Sie eindeutig die einzelnen Header und tragen Sie die wichtigen Informationen ein. Nutzen Sie hierfür auch die in Tabelle 1 gelisteten Adressen.

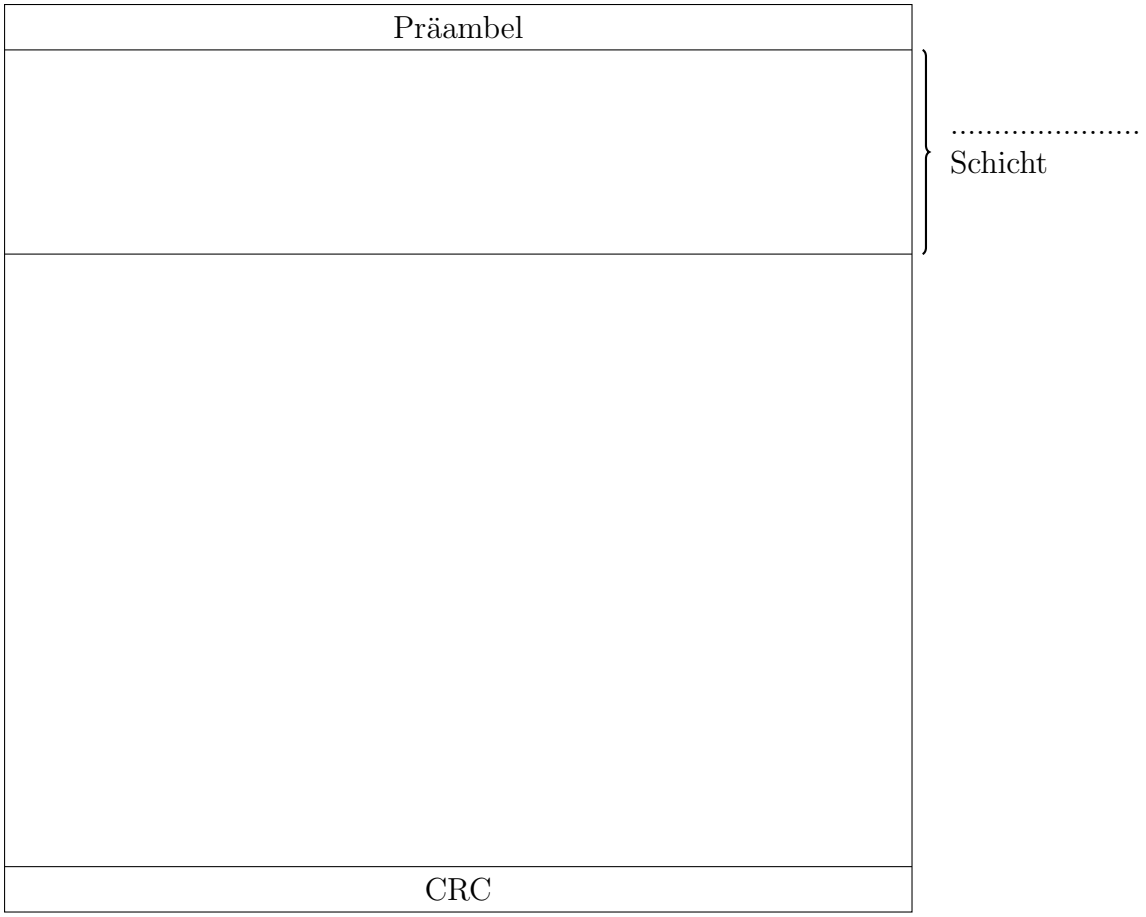

Abbildung 2: Ethernet Paket

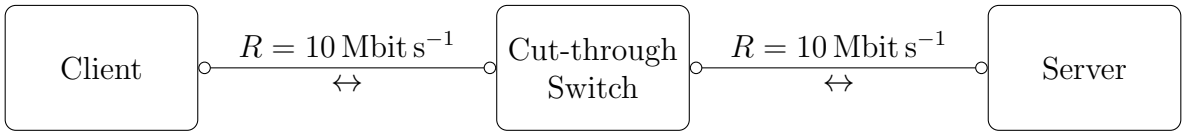

Abbildung 3: Netzwerktopologie

Sicherungsschicht Header	$h_2 = 25\,\mathrm{B}$
Netzwerkschicht Header	$h_3 = 50\,\mathrm{B}$
Transportschicht Header	$h_4 = 50\,\mathrm{B}$
Nutzdaten	$L = 1125\,\mathrm{B}$
Ausbreitungsverzögerung pro Link	$d_{prop} = 10\,\mu\mathrm{s}$

Tabelle 2: Zusätzliche Konstanten

2 Übertragungslatenzen (24 Punkte gesamt)

Gegeben sei die Netzwerktopologie gemäß Abbildung 3. Zwei Standard Ethernet (10Base-T) fähige Computer sind über einen sogenannten *Cut-through*-Switch miteinander verbunden. Im Unterschied zu einem klassischen *Store-and-Forward*-Switch leitet dieser die Pakete sobald wie möglich (= sobald der Empfänger bekannt ist) weiter, ohne den kompletten Empfang des Pakets abzuwarten.

Zusätzlich gelten die Konstanten gemäß Tabelle 2. Alle anderen relevanten Größen können zu 0 gesetzt werden. Gehen Sie vereinfachend davon aus, dass die einzelnen Header immer vollständig empfangen werden müssen. D.h. der Switch kann während er einen Header (h_2, h_3, h_4) empfängt keine Weiterleitungsentscheidung treffen.

2.1 Funktionsweise (2 Punkte)

Auf welcher Schicht arbeitet ein Switch? Welche minimale Information wird für die Weiterleitung des Pakets benötigt?

2.2 Charakerisierung (4 Punkte)

Nennen Sie je einen Vor- und einen Nachteil von *Cut-through* switching im Vergleich zum *Store-and-Forward* switching.

2.3 Round Trip Time (4 Punkte)

Berechnen Sie die Round Trip Time (RTT) nur unter Berücksichtigung der in Tabelle 2 gelisteten Informationen für ein idealisiertes Paket der Gesamtgröße 0 B zwischen beiden Computern in Abbildung 3. Geben Sie ebenfalls eine Formel an.

2.4 Paketaustausch (5 Punkte)

Tragen Sie einen Paketverlauf für das *Cut-through* Verfahren in Abbildung 4 ein. Die Zusammensetzung des Paket (einzelne Header und Nutzdaten) muss klar erkennbar sein.

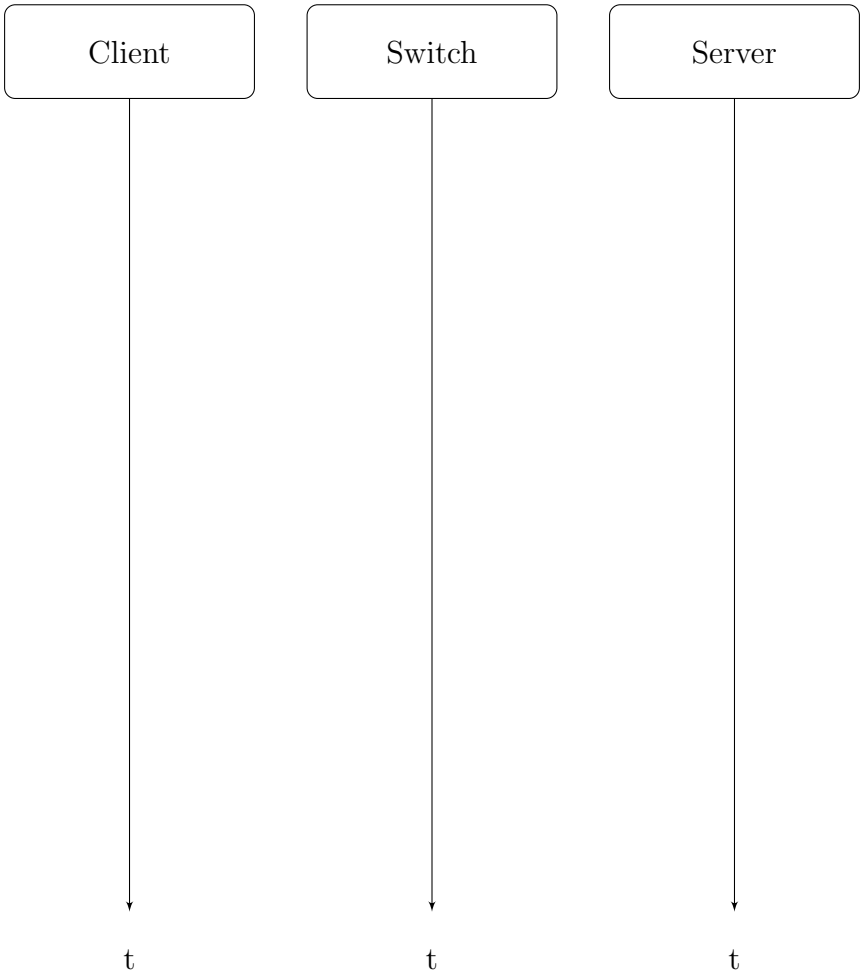

Abbildung 4: Ablauf Diagramm

2.5 Cut-through und Store-and-Forward im Vergleich (9 Punkte)

Wie lange dauert es L Nutzdaten (= ein Paket) zwischen den Computern zu übertragen (einfacher Weg)? Geben Sie jeweils eine allgemeine Formel für das *Cut-through* (d_{ct}) und für das *Store-and-Forward* (d_{sf}) Verfahren unter Berücksichtigung einer variablen Anzahl von Links E an. Um wie viel Prozent ist hier (Abbildung 3, $E = 2$) welches Verfahren schneller?

3 POP3-Server-Programmierung (30 Punkte)

Programmieren Sie einen vereinfachten POP3-Server in JAVA, der auf eine Verbindung von einem POP3-Client wartet, die Authentifizierung des Benutzers überprüft und Anfragen (STAT, LIST, QUIT) des Clients beantworten kann. Jede Antwort des Servers beginnt mit +OK oder –ERR, worauf weitere Informationen bis zur Steuerzeichensequenz „$\backslash r \backslash n$“ folgen können. Bei LIST gibt es eine mehrzeilige Antwort, deren Ende durch eine Zeile, die nur einen Punkt enthält, markiert wird.

Realisieren Sie das in Abbildung 5 gezeigte Verhalten in der *run()*-Methode der Klasse *Pop3Server*. Es ist dabei anzunehmen, dass gleichzeitig nur ein Client bedient wird und dass dieser nur die im Statechart spezifizierten Befehle in der angegebenen Reihenfolge sendet. Leiten Sie die Klasse *Pop3Server* von *Base* ab und nehmen Sie an, dass die Methoden für die Kommunikation mit dem Client, Authentifizierung und Ausführung von POP3-Befehlen in *Base* bereits implementiert sind. Bei Fehler werfen die Methoden eine *POP3Exception* bzw. eine *IOException*. Die *IOExceptions* müssen dabei nicht behandelt werden und führen zum Abbruch des Programms. Die *POP3Exceptions* müssen dagegen abgefangen und mit einer –ERR Nachricht beantwortet werden. Beachten Sie den unten begefügten Beispielablauf in Listing 1 sowie die Kommentare von *Base* in Listing 2.

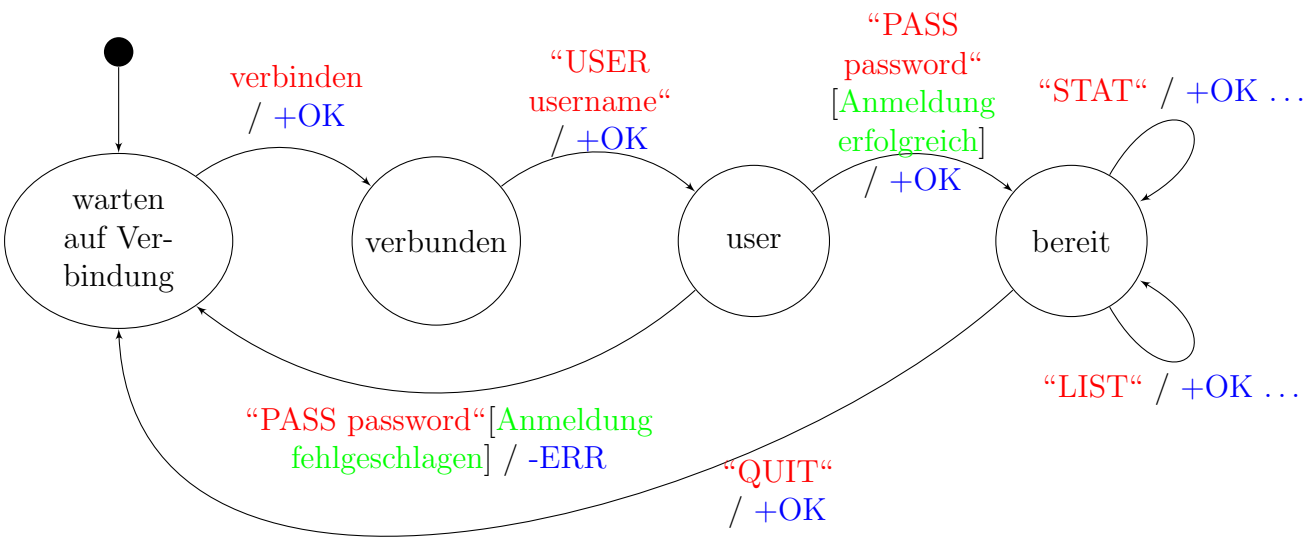

Abbildung 5: POP3-Serververhalten

Listing 1: Beispielablauf POP3-Session

```
S: (wartet auf Verbindung)
C: (oeffnet eine Verbindung)
S: +OK example.com POP3-Server
C: USER username
S: +OK Please enter password
C: PASS passwort_in_klartext
S: +OK mailbox locked and ready
C: STAT
S: +OK 2 436
C: LIST
S: +OK mailbox has 2 messages
S: 1 232
S: 2 204
S: .
C: QUIT
S: +OK bye
S: (schliesst die Verbindung)
```

Listing 2: Abstrakte Klasse Base

```
1  import java.io.IOException;
2
3  public abstract class Base {
4
5          final String NL = "\r\n";
6          final String OK = "+OK ";
7          final String ERR = "-ERR ";
8
9          /* Wartet auf eingehende Verbindung vom POP3-Client */
10         void acceptConnection() throws IOException {...}
11
12         /* Trennt die Verbindung zum POP3-Client */
13         void disconnect() throws IOException {...}
14
15         /* Wartet bis ein Befehl vom verbundenen POP3-Client
16          * emfangen wurde und liefert die emfangene Befehlzeile
17          * als String zurueck */
18         String receive() throws IOException {...}
19
20         /* Sendet eine Antwortnachricht an den POP3-Client. */
21         void send(String answer) throws IOException {...}
22
23         /* Authentifizierung beim POP3-Server.
24          * Liefert bei Erfolg true, sonst false zurueck. */
25         boolean authentificate(String user,String password){...}
26
27         /* Liefert die Anzahl und die Gesamtgroesse der E-Mails
28          * in der Mailbox des aktuell autorisierten Benutzers.
29          * Beispiel: "2 436"    */
30         String stat() throws POP3Exception {...}
31
32         /* Liefert eine Liste, bestehend aus der ID und
33          * der Groesse (in Bytes) jeder Nachricht im Postfach.
34          * Beispiel: "1 232", "2 204"   */
35         String[] list() throws POP3Exception {...}
36
37         /* Muss in POP3Server-Klasse implementiert werden */
38         abstract public void run() throws IOException;
39 }
```

public class POP3Server

4 Routingverfahren (24 Punkte gesamt)

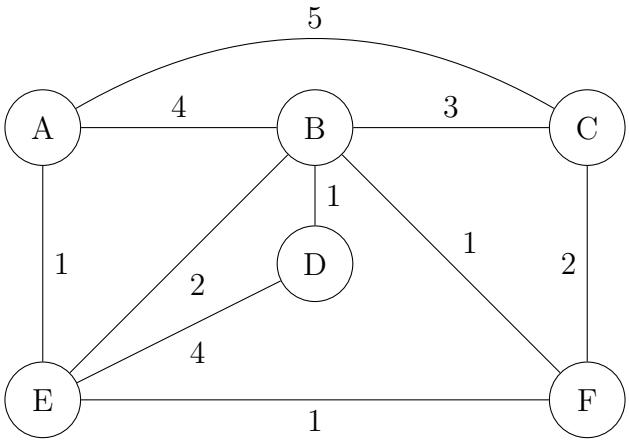

Abbildung 6: Netzwerk

4.1 Dijkstra-Verfahren (10 Punkte)

Abbildung 6 zeigt eine Netzwerktopologie mit sechs Knoten. Führen Sie aus Sicht des Knotens A das Dijkstra-Verfahren in der Tabelle 3 durch, um minimale Pfade zwischen den Knoten zu erhalten. Verwenden Sie bitte die gleiche Notation wie in der Vorlesung.

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

Tabelle 3: Dijkstra-Verfahren für Knoten A

4.2 Forward-Search-Algorithmus (12 Punkte)

Ausgehend aus der Netzwerktopologie in Abbildung 6 verfolgen Sie in Tabelle 4 den Ablauf des Forward-Search-Algorithmus für den Knoten C. Verwenden Sie bitte die gleiche Notation wie in der Vorlesung.

S	bestätigte Liste	vorläufige Liste
1		
2		
3		
4		
5		
6		
7		

Tabelle 4: Forward-Search-Algorithmus für Knoten C

4.3 Minimal aufspannender Baum (2 Punkte)

Zeichnen Sie auf Basis von Tabelle 3 (Dijkstra-Verfahren für Knoten A) die daraus resultierende minimale Baumstruktur in Abbildung 7:

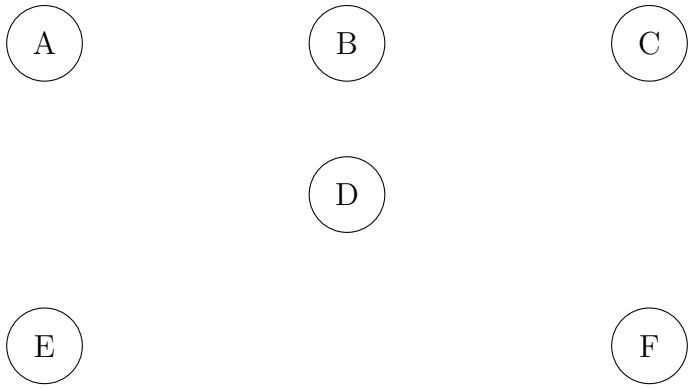

Abbildung 7: Netzwerk

Zusatzblatt

Zusatzblatt