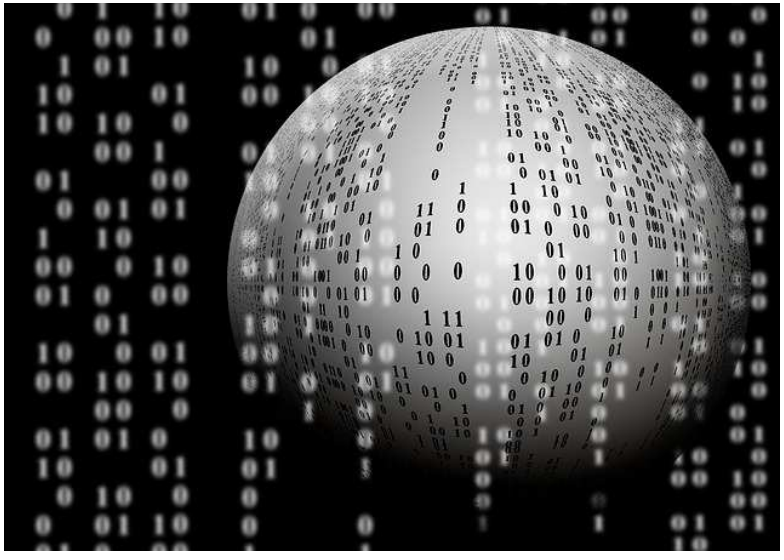
Kommunikationssysteme (Lösungsvorschläge)

Prof. Dr. Henning Pagnia

DHBW Mannheim



Aufgabe 1: (Kollisionen)

Ein Nachteil bei Broadcast-Netzen ist die Verschwendung von Netzkapazität, wenn mehrere Hosts gleichzeitig auf den Kanal zugreifen wollen. Stellen Sie sich als einfaches Beispiel vor, dass die Zeit in einzelne Zeitschlitze (*Slots*) unterteilt wird, und jeder der N Hosts eines Netzes mit einer mittl. Wahrscheinlichkeit p während eines Slots versucht, auf den Kanal zuzugreifen.

Welcher Anteil an Slots wird im Mittel durch Kollisionen verschwendet?
Stellen Sie die Situation grafisch dar!

Zu Aufgabe 1: (Kollisionen)

Prob(Kollision innerhalb eines Slots)

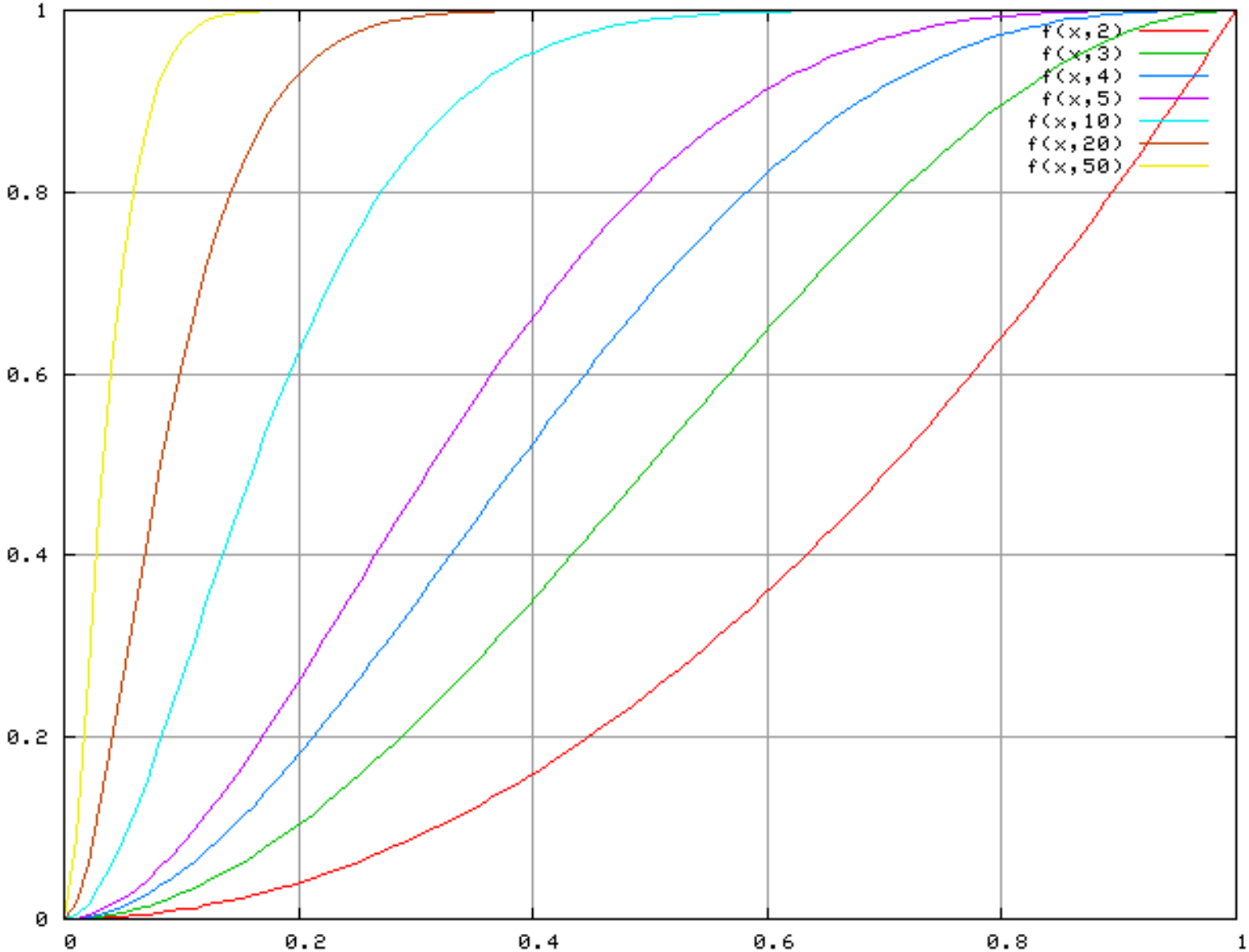
= Prob(In dem Slot versuchen genau zwei oder mehr Station zu senden)

= 1− Prob(Es versucht keine ODER genau eine Station zu senden)

= 1 − [Prob(Keine Station sendet) +
Prob(GENAU eine Station sendet UND alle anderen senden nicht)]

$$= 1 - [1 \cdot \boldsymbol{p}^0 \cdot (1 - \boldsymbol{p})^{\boldsymbol{N}} + \boldsymbol{N} \cdot \boldsymbol{p}^1 \cdot (1 - \boldsymbol{p})^{\boldsymbol{N}-1}]$$

Zu Aufgabe 1: (Kollisionen) (Forts.)



(Grafik erstellt mit `gnuplot`)

Aufgabe 2: (Kollisionen)

Überlegen Sie, wie sich die Effizienz des Netzes aus Aufgabe 1 steigern lässt!

Zu Aufgabe 2: (Kollisionen)

Wir setzen keine Hubs sondern Switches ein. Somit treten (i. Allg.) keine Kollisionen auf.
Zudem steht für jede paarweise Verbindung die volle Datenrate zur Verfügung.

Aufgabe 3: (Übertragungsrate)

Wie hoch ist die maximale Übertragungsrate, wenn ein binäres Signal über einen 3-kHz-Kanal mit einem Rauschabstand von 20 dB gesendet wird?

Zu Aufgabe 3: (Übertragungsrate)

(a) nach Shannon

$$\boldsymbol{TR} \leq \boldsymbol{B} \cdot \log_2(1 + \boldsymbol{S}/\boldsymbol{N})$$

Also: $\boldsymbol{TR} \leq 3000 \text{ Hz} \cdot \log_2(1 + 10^{20/10}) = 19,97 \text{ Kbps}$

(b) nach Nyquist

$$\boldsymbol{TR} \leq 2\boldsymbol{B} \cdot \log_2 \boldsymbol{V}$$

Also: $\boldsymbol{TR} \leq 2 \cdot 3000 \text{ Hz} \cdot \log_2 2 = 6 \text{ Kbps}$

$\Rightarrow$ beide Grenzen müssen eingehalten werden, daher gilt: $\boldsymbol{TR} \leq 6$ Kbps !

Aufgabe 4: (Übertragungsdauer)

Ein historischer Fernkopierer digitalisierte Bilder in 300 dpi mit je 4 Bit pro Pixel.
Wie lange dauerte die Übertragung eines 8x10 Zoll großen Bildes über einen ISDN Datenkanal mit einer Datenrate von 64 kbps?
Modernere Faxe schaffen das schneller. Wie machen die das?

Zu Aufgabe 4: (Übertragungsdauer)

Hinweise

1 Zoll = 1 Inch; dpi: *dots per inch*; 1 Dot = 1 Pixel

Rechnung

Anzahl Bits = (8 inch ·300 dpi) · (10 inch · 300 dpi) · 4 Bit = $28,8 \cdot 10^6$ Bit

Ein ISDN Datenkanal (B-Kanal) schaffte 64 Kbps:

$$\boldsymbol{T} = \frac{28,8 \cdot 10^6 \ \boldsymbol{Bit}}{64 \cdot 10^3 \ \boldsymbol{bps}} = 450 \ \boldsymbol{s} = 7,5 \ \boldsymbol{min}$$

Modernere Faxe übertragen meist nur s/w und komprimieren zusätzlich. Darüberhinaus ist ein Fax oft zu 90 % weiß (insg. ca. Faktor 20).

Aufgabe 5: (Datenrate)

In Japan und in den USA werden in der Telefontechnologie so genannte T1-Träger mit einer Bandbreite von 1,544 Mbps verwendet. Wie hoch ist die Kapazität eines 100 km langen Kabels, d. h. wieviele Bits passen in das Kabel?

Hinweis:
Die Ausbreitungsgeschwindigkeit in Kupfer- und Glasfaserkabeln beträgt ca. $\frac{2}{3}c_0$.

Zu Aufgabe 5: (Datenrate)

Bestimmen der Latenz T_L sowie der Kabelkapazität C:

$$v = \frac{s}{t} \Rightarrow T_L = \frac{100\ km}{200\,000\ km/s} = \frac{1}{2000}\ s$$

$$C = \frac{1}{2000}\ s \cdot 1,544 \cdot 10^6\ bps = 772\ Bit$$

($\Rightarrow$ Ausdehnung eines Bit: $100\ km/772\ Bit \approx 130\ m/Bit$)

Aufgabe 6: (Signalkodierung)

Skizzieren Sie für den Bitstrom 0001110101

(a) die Manchester-Kodierung.

(b) die differenzielle Manchester-Kodierung.
(Annahme: Der Signalpegel ist unmittelbar vor Beginn der Übertragung auf LOW)

Zu Aufgabe 6: (Signalkodierung)

(a) Manchester-Kodierung

0	0	0	1	1	1	0	1	0	1
LH	LH	LH	HL	HL	HL	LH	HL	LH	HL

(b) diff. Manchester-Kodierung (zu Beginn: Low)

	0	0	0	1	1	1	0	1	0	1
L	HL	HL	HL	LH	HL	LH	LH	HL	HL	LH

Aufgabe 7: (Fehlerkontrolle)

Eine Nachricht auf einer oberen Schicht ist in 10 Frames unterteilt, die jeweils eine Chance von 80% haben, unbeschädigt anzukommen. Wie oft muss die Nachricht durchschnittlich übertragen werden, um vollständig und fehlerfrei anzukommen, wenn in der Sicherungsschicht keine Fehlerbehandlung vorgenommen wird?

Zu Aufgabe 7: (Fehlerkontrolle)

Prob(ein Frame wird fehlerfrei übertragen) $= 0.8$
$\Rightarrow$ Prob(Alle 10 Frames werden fehlerfrei übertragen) $= 0.8^{10} = \boldsymbol{p}$

Sei $\boldsymbol{X}$ die Zufallsvariable, welche die Anzahl der aufeinanderfolgenden Sendeversuche bis zum Erfolg beschreibt.

$$\boldsymbol{E}(\boldsymbol{X}) = \sum_{\boldsymbol{i}=1}^{\infty} \boldsymbol{i} \cdot (1-\boldsymbol{p})^{\boldsymbol{i}-1} \cdot \boldsymbol{p}$$

$$= \boldsymbol{p} \cdot \sum_{\boldsymbol{i}=1}^{\infty} \boldsymbol{i} \cdot (1-\boldsymbol{p})^{\boldsymbol{i}-1} = \boldsymbol{p} \cdot \frac{1}{(1-(1-\boldsymbol{p}))^2} = \frac{1}{\boldsymbol{p}} = 9.3$$

($\boldsymbol{i}$ gibt an, wie oft bei der Übertragung ein Fehler aufgetreten ist.)

Aufgabe 8: (Frame-Bildung)

(a) Wie sieht der folgende Text nach erfolgtem Character-Stuffing und angebrachter Anfangs- und Ende-Markierung aus:
Q STX E DLE DLE D

(b) Wie sieht die folgende Bitfolge nach erfolgtem Bit-Stuffing aus:
0111101111101111110

Zu Aufgabe 8: (Frame-Bildung)

(a) Character-Stuffing

DLE STX Q STX E DLE DLE DLE DLE D DLE ETX

(b) Bit-Stuffing

01111110 01111011111 0 011111 0 10 01111110

Aufgabe 9: (Fehlerkontrolle)

Welchen Rest erhält man, wenn man $x^7 + x^5 + 1$ durch das generierende Polynom $x^3 + 1$ teilt?

Zu Aufgabe 9: (Fehlerkontrolle)

Polynomdivision

$(x^7 + x^5 + 1)/(x^3 + 1) = x^4 + x^2 - x \quad \text{Rest } (-x^2 + x + 1)$

in GF(2)

10100001 / 1001 = 10110 Rest 111

Aufgabe 10: (CRC)

Berechnen Sie für alle 2-Bit-Frames die Codewörter nach dem CRC-Algorithmus (auf Folie 58) für das generierende Polynom
$\boldsymbol{G}(\boldsymbol{x}) = \boldsymbol{x}^3 + \boldsymbol{x}^2 + 1$.

Welche Hamming-Distanz hat der Code? Wieviele Fehler können sicher erkannt werden?

Zu Aufgabe 10: (CRC)

Es gibt vier Frames mit 2 Bit Nutzdaten: 00, 01, 10, 11.

Mit 1101 als Generator berechnen wir:

00 000 (Rest aus 00000 / 1101 anhängen)

01 101

10 111

11 010

Die Hammingdistanz des Codes ist 3, daher können bis zu 2 Bitfehler sicher erkannt werden.

Aufgabe 11: (CRC Implementierung)

Implementieren Sie eine Java-Klasse CRC mit der Methode
public static int[] checksum(int[] frame, int[] generator),
welche die CRC-Prüfsumme als int-Array zurückliefert.
Definieren Sie die die Arrays z. B. mittels int frame[] = {1,1,1,1};

Zu Aufgabe 11: (CRC Implementierung)

```
public static int[] computeCRC (int[] f, int[] generator){
    // make a copy of frame f and extend it with zeros
    int frame[] = new int[f.length+generator.length-1];
    for (int i=0; i < f.length+generator.length-1; i++)
        if (i < f.length)
            frame[i] = f[i];
        else
            frame[i] = 0;

    // compute checksum
    for (int j = 0; j <= (frame.length-generator.length); j++)
        if (frame[j] != 0) // else result = 0 -> continue
            for (int i = 0; i<generator.length; i++)
                frame[i+j] = frame[i+j] ^ generator[i]; // XOR
    // return checksum
    int check[] = new int[generator.length - 1];
    for (int i = 0; i < check.length; i++)
            check[i] = frame[frame.length-generator.length+1+i];
    return check;
}//computeCRC
```

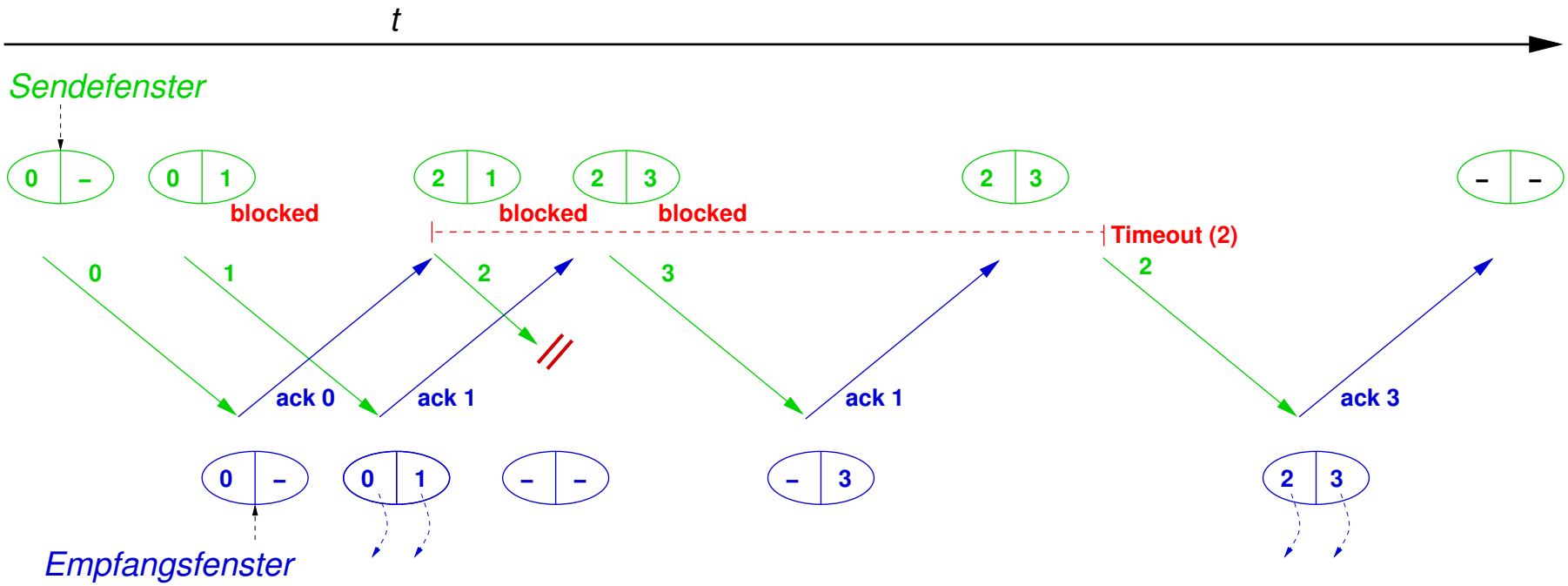
Aufgabe 12: (Flusskontrolle)

Skizzieren Sie den Ablauf der Übertragung von vier Frames mit 2-Bit-Sequenznummern nach dem *Selective Repeat*–Protokoll, wobei das dritte Frame zunächst fehlerhaft übertragen wird.
Nehmen Sie an, dass der Sender sehr schnell ist, Bestätigungen als Sammelbestätigungen gesendet werden und Timeout-Zeiten sowie RTT lang sind.
Hinweis: Berücksichtigen Sie die korrekten Fenstergrößen!

Zu Aufgabe 12: (Flusskontrolle)

Fenstergröße bei *Selective Repeat*: $\boldsymbol{F} = 2^{\boldsymbol{n}-1} \Rightarrow$ für $\boldsymbol{n} = 2$ gilt: $\boldsymbol{F} = 2$

Verlauf der Kommunikation:



Aufgabe 13: (ALOHA)

(a) Eine Gruppe von N Stationen teilt sich einen reinen ALOHA-Kanal mit 56 Kbps. Jede Station gibt im Durchschnitt alle 100 Sekunden einen 1.000 Bit-Frame aus, selbst dann wenn der vorherige noch nicht vollständig übertragen wurde (Pufferung).

Welchen Wert sollte N hier sinvollerweise nicht überschreiten?

(b) 10.000 Terminals eines Flugreservierungssystems konkurrieren um die Benutzung eines einzigen Slotted-ALOHA-Kanals. Im Durchschnitt stellt jede Station 18 Anfragen pro Stunde. Ein Slot ist 125 μs lang.

Berechnen Sie die mittlere Kanalbelastung sowie den Durchsatz!

Zu Aufgabe 13: (ALOHA)

(a)

Der Kanal darf durch die N Stationen nicht überlastet werden,
d. h. $S \leq 0.184$ (vgl. Formeln und Grafiken auf den Folien).
Jede Station sendet alle 100 Sekunden 1000 Bits, d. h. mit 10 *bps*

$$N \leq \frac{56\,000\ \textit{bps} \cdot 0.184}{10\ \textit{bps}} = 1030,4$$

D. h. es sollten max. 1030 Stationen angeschlossen werden.

(b)

Berechnung des Gesamtanfragevolumens:
$10\,000 \cdot 10$ Anfragen / Stunde $= 50$ Anfragen / Sekunde
Ein Slot dauert 125 $\mu s = 125 \cdot 10^{-6} s$ $\Rightarrow$ es gibt 8000 Slots / Sekunde.
$\Rightarrow$ Kanalbelastung: $G = \frac{50}{8000} = 0.00625$ Anfragen pro Slot

($\Rightarrow$ Durchsatz: $S = G \cdot e^{-G} = 0.006211$)
Der Kanal ist nur gering belastet; es gibt nur wenige Kollisionen.

Aufgabe 14: (Slotted-ALOHA)

Messungen an einem Slotted-ALOHA-Kanal mit unendlich vielen Benutzern zeigen, dass 10% aller Slots leer sind.

(a) Wie hoch ist die Kanalbelastung $\boldsymbol{G}$?

(b) Wie hoch ist der Durchsatz $\boldsymbol{S}$?

(c) Ist der Kanal unterlastet oder überlastet?

Zu Aufgabe 14: (Slotted-ALOHA)

Bei Slotted ALOHA gilt: $P_0 = e^{-G}$

(a)

Gegeben $P_0 = 0.1 \Rightarrow G = -\ln 0.1$
$\Rightarrow G = 2,303$

(b)

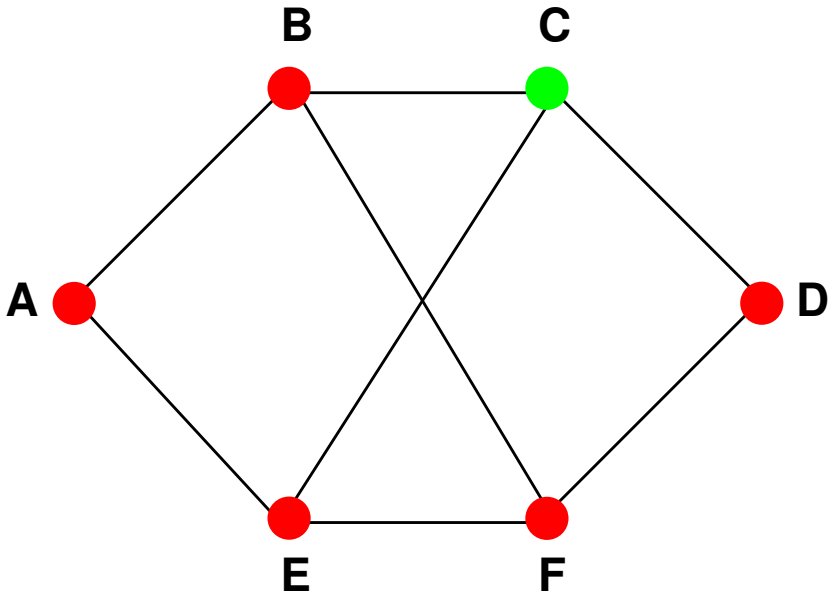
$S = G \cdot P_0 = 0,23$

(c)

Überlast, da $G > 1$

Aufgabe 15: (DVR)

Für das folgende Teilnetz wird *Distance Vector Routing* eingesetzt.



Router ***C*** bekommt von seinen Nachbarn die folgenden Vektoren geschickt:
von ***B***: (5,0,8,12,6,2), von ***D***: (16,12,6,0,9,10) und von ***E*** (7,6,3,9,0,4).
Die gemessenen Verzögerungen sind zu ***B***: 6, zu ***D***: 3 und zu ***E***: 5.

Wie sieht die neue Routing-Tabelle von ***C*** aus? Geben Sie für die ausgehenden Leitungen sowohl die Nachbar-Router an als auch die Distanzvektoren!

Zu Aufgabe 15: (DVR)

C berechnet die folgenden Werte:

Zu	via ***B***	via ***D***	via ***E***	Distanz	Route
A	11	19	12	11	***B***
B	6	15	11	6	***B***
C	14	9	8	0	–
D	18	3	14	3	***D***
E	12	12	5	5	***E***
F	8	13	9	8	***B***

Die Werte in der Spalte *Route* dienen fortan als neue Routingtabelle, die in der Spalte *Distanz* werden als neuer Distanzvektor in der nächsten Runde an alle Nachbarn gesendet.

Aufgabe 16: (Implementierung Dijkstra-Algorithmus)

Implementieren Sie den Dijkstra-Algorithmus in Java!

Zu Aufgabe 16: (Implementierung Dijkstra-Algorithmus)

```
public class Dijkstra {
   static final int MAX = Integer.MAX_VALUE;
   static final int DIM = 8;
   static int[][] C = {
      {0,2,-1,-1,  -1,-1,6,-1},  //A
      {2,0,7, -1,   2,-1,-1,-1}, //B
      {-1,7,0,3,   -1,3,-1,-1},  //C
      {-1,-1,3,0,  -1,-1,-1,2},  //D
      {-1,2,-1,-1,  0,2,1,-1},   //E
      {-1,-1,3,-1,  2,0,-1,2},   //F
      {6,-1,-1,-1,  1,-1,0,4},   //G
      {-1,-1,-1,2,  -1,2,4,0}    //H
   };
   static int[] E = {1,1,1,1,1,1,1,1};
   static int s = 0;
   static int cost[] = {0,MAX,MAX,MAX,MAX,MAX,MAX,MAX};
   static int pre[] = {-1,-1,-1,-1, -1,-1,-1,-1};
   static String bez[] = {"A","B","C","D","E","F","G","H"};

   public static void main (String[] args) {
      printC();
      int a = s;
      while (notEmptyE()) {
         E[a]=0;
         System.out.println("Startknoten:␣" + name(a)+"\\\\");
         for (int j=0; j<DIM; j++)
            if ((E[j] != 0) && (C[a][j] > 0))
               if ((cost[a] + C[a][j]) < cost[j]) {
                  cost[j] = cost[a] + C[a][j];
                  pre[j] = a;
               }
         a = minE();
      } // while
      printCost();
      printPre();
   } // main
```

Dijkstra.java. (Forts.)

```
static boolean notEmptyE() {
   for (int i = 0; i<DIM; i++)
      if (E[i] != 0)
         return true;
   return false;
}

static int minE() {
   int node = -1;
   int min = MAX;
   for (int i = 0; i<DIM; i++)
      if (E[i] != 0 && cost[i] < min) {
         min = cost[i];
         node = i;
      }
   return node;
}

static String name(int i) {
   if (i>=0 && i<DIM)
      return bez[i];
   return "ERROR";
}
static String cost(int i) {
   if (cost[i] == MAX) return "X"; // unendlich
   return ""+cost[i];
}

static void printCost() {
   System.out.print("cost␣=␣(");
   for (int i=0; i<DIM-1; i++)
      System.out.print(cost(i)+ ",␣");
   System.out.println(cost(DIM-1)+ ")");
}
```

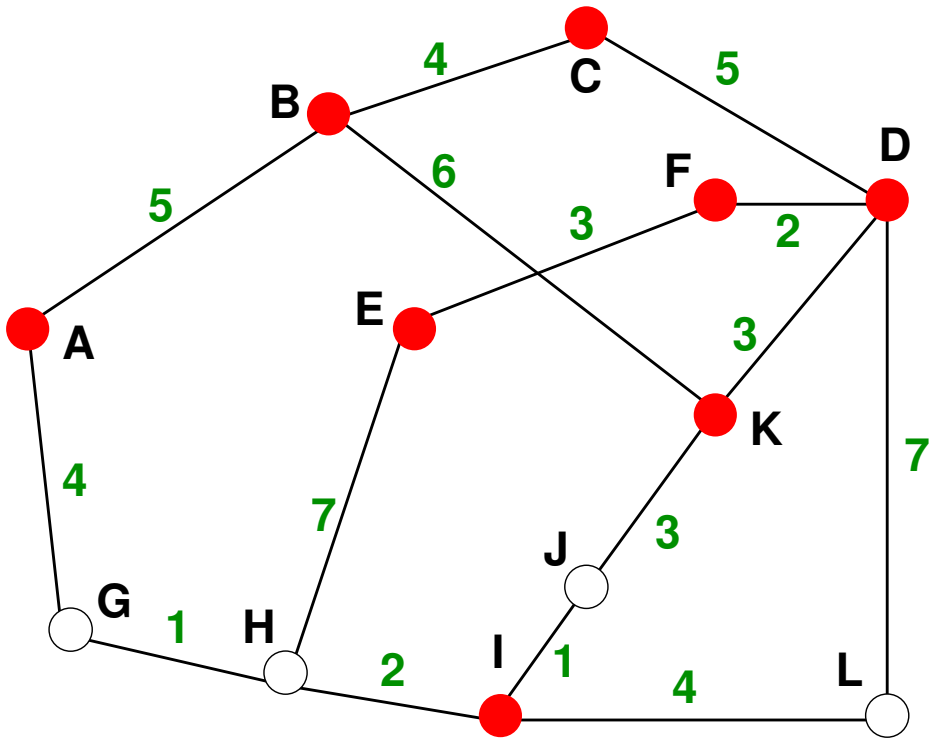
```
static void printPre() {
System.out.print("pre␣=␣(");
   for (int i=0; i<DIM-1; i++)
      System.out.print(name(pre[i])+ ",␣");
   System.out.println(name(pre[DIM-1])+ ")");
}

static void printC() {
   System.out.println("C␣=␣");
   for (int i=0; i<DIM; i++) {
      System.out.print("(␣");
      for (int j=0; j<DIM-1; j++) {
         if (C[i][j]<0)
            System.out.print("-,␣");
         else
            System.out.print(C[i][j]+ ",␣");
      }
      if (C[i][DIM-1]<0)
         System.out.println("-␣)");
      else
         System.out.println(C[i][DIM-1]+ "␣)");
   } // for
} // printC

} // class Dijkstra
```

Aufgabe 17: (Minimaler Spannbaum)

Berechnen Sie mit dem Dijkstra-Algorithmus – unter Angabe aller notwendigen Zwischenschritte – den minimalen Multicast-Spannbaum für Router ***C*** in dem folgenden Teilnetz, wobei Gruppenmitglieder nur über die Router ***A***, ***B***, ***C***, ***D***, ***E***, ***F***, ***I***, ***K*** angeschlossen sind.



Wie wird dieser Spannbaum verwendet?

Zu Aufgabe 17: (Minimaler Spannbaum)

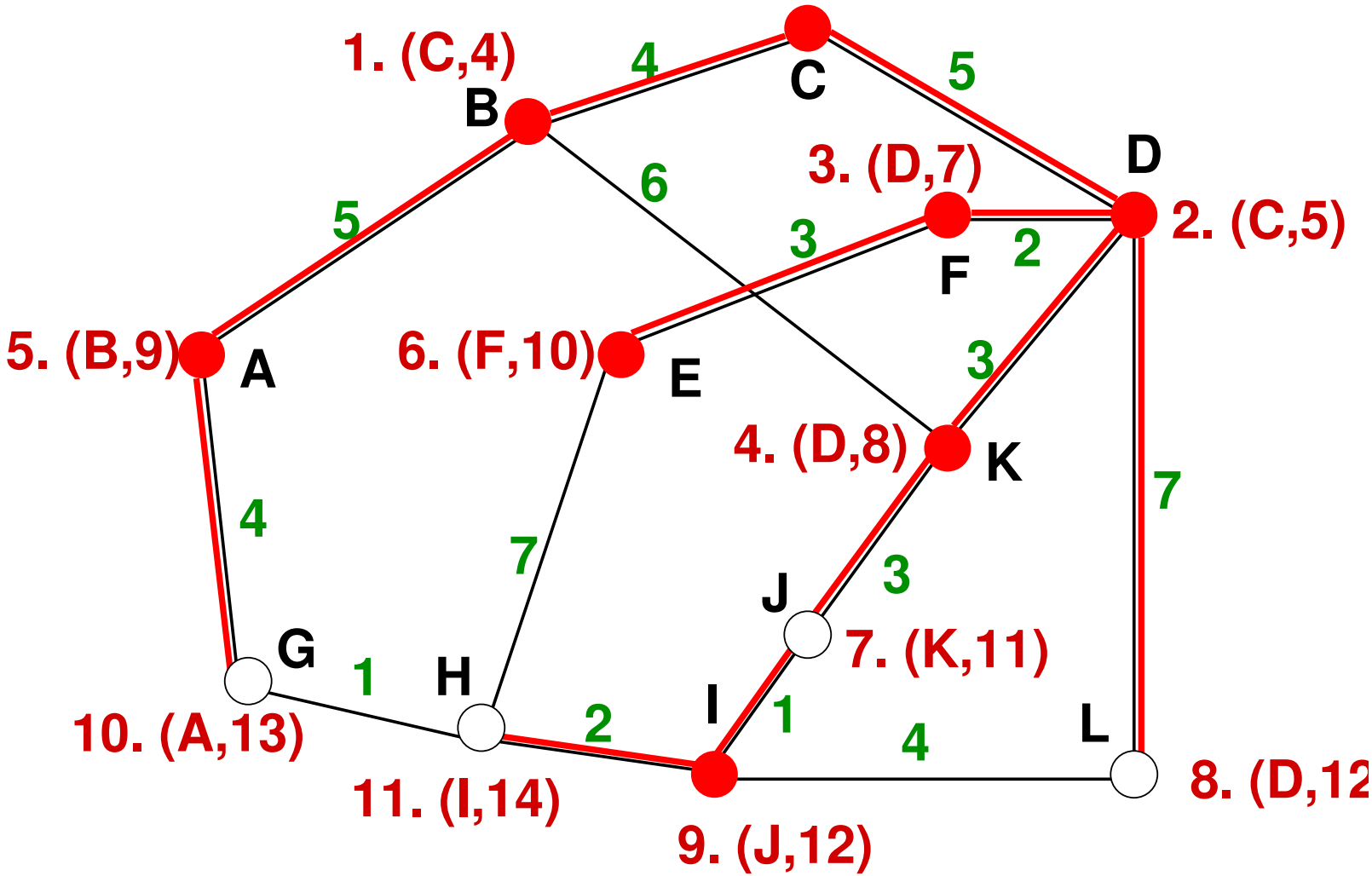
Zunächst mit dem Dijkstra den vollständigen minimalen Spannbaum berechnen:

(1) a=C; B(4,C); D(5,C)
(2) a=B; A(9,B); K(10,B)
(3) a=D; F(7,D); K(8,D); L(12,D)
(4) a=F; E(10,F)
(5) a=K; J(11,K)
(6) a=A; G(13,A)
(7) a=E; H(17,E)
(8) a=J; I(12,J)
(9) a=L
(10) a=I; H(14,I)
(11) a=G
(12) a=H

$\Rightarrow$ pre = (B,C,-,C,F,D,A,I,J,D,D)

Anschließend entsprechend der Multicast-Gruppe verkürzen:
$\Rightarrow$ Knoten ***G***, ***H*** und ***L*** entfernen (***J*** muss bleiben)

Zu Aufgabe 17: (Minimaler Spannbaum)



Aufgabe 18: (Netzmaske)

In einem Netz der Klasse ***B*** im Internet gibt es ein Teilnetz mit der Teilnetzmaske

`255.255.240.0`

Wieviele verschiedene Rechneradressen werden maximal unterstützt?

Zu Aufgabe 18: (Netzmaske)

Die Teilnetzmaske in Binärdarstellung lautet:

11111111.11111111.11110000.00000000

(bzw. "/20")

Damit lassen sich theoretisch $2^{12} = 4096$ Adressen bilden.

Die Rechneradressen mit lauter Nullen (Netzadresse) bzw. lauter Einsen (Broadcast-Adresse) können jedoch nicht vergeben werden.
$\Rightarrow$ Demnach werden insgesamt 4094 Rechneradressen unterstützt.

Aufgabe 19: (TCP)

Das Feld *Window size* im TCP-Header begrenzt leider die Datenrate.

(i) Wieso?

(ii) Wie hoch ist die effektive Datenrate von TCP maximal bei einer angenommenen Round-Trip-Time von 0.5 Sekunden?

(iii) Wie kann dieser Wert erhöht werden?

Zu Aufgabe 19: (TCP)

(i)

Wenn das Empfangsfenster voll ist, muss der Sender in jedem Fall die Übertragung unterbrechen und warten. Dadurch sinkt die Effizienz, falls das Fenster nicht groß genug sein kann. Die max. Größe des Fensters wird daher durch die Anzahl der Bits des Feldes *Window Size* bestimmt (16 Bit) $\Rightarrow$ max. Fenstergröße $= 2^{16}$ Byte

(ii)

Max. Datenrate $= 2^{16}$ / RTT $= 2^{17} = 131\,072$ Byte / s ≈ 1 Mbps
(Formel vgl. RFC 1323)

(iii)

Das Sendefenster kann mittels des Header-Feldes *Options* vergrößert und damit die maximale Datenrate nahezu beliebig erhöht werden.

Aufgabe 20: (Huffman)

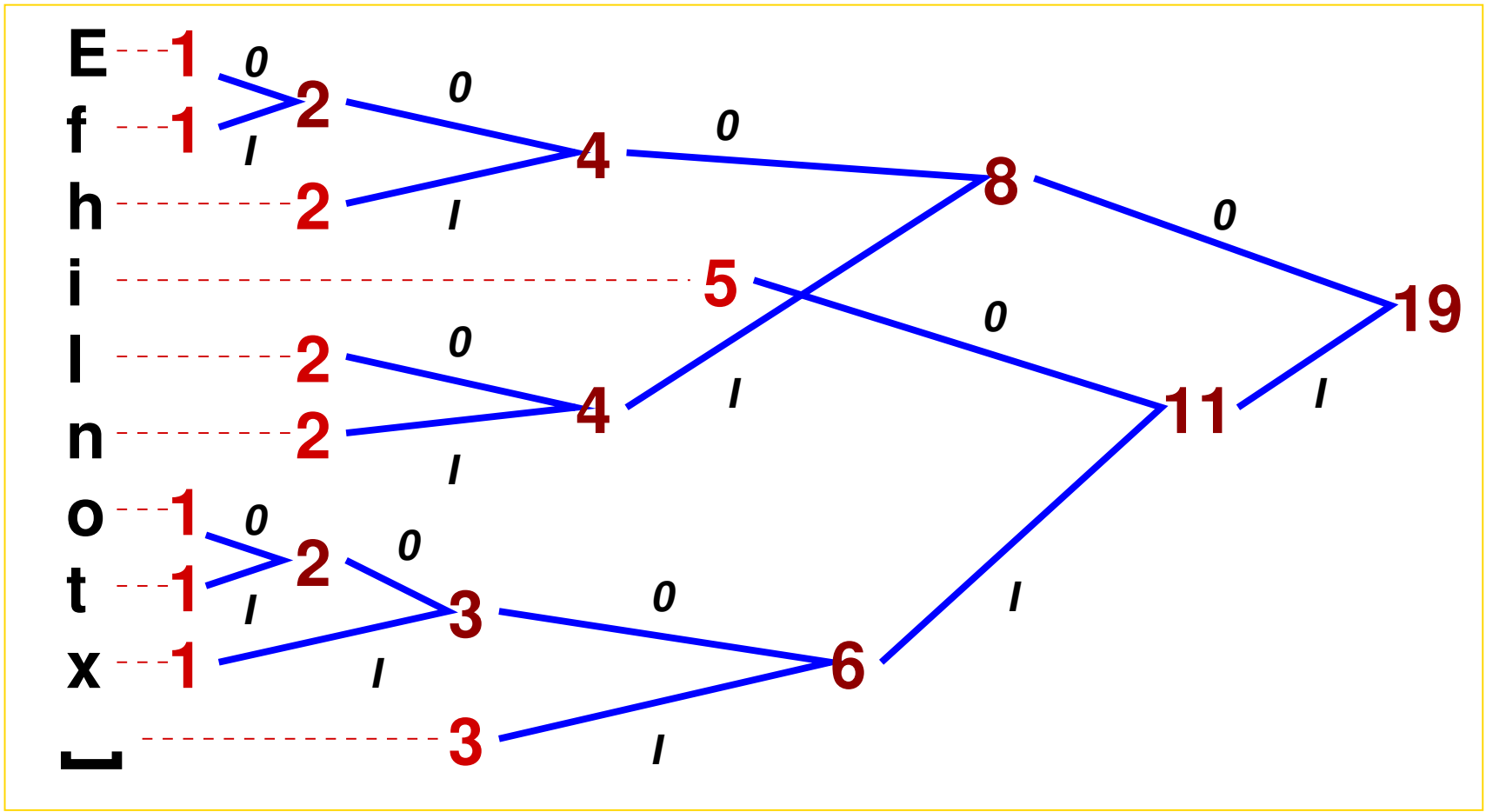
Führen Sie eine Huffman-Kodierung für den folgenden Text durch, indem Sie die einzelnen Buchstaben und die Leerzeichen als Symbole wählen:

```
Ex nihilo nihil fit
```

(Aus Nichts kann man nichts machen)

Zu Aufgabe 20: (Huffman)

(1) Huffman- Baum konstruieren (es gibt verschiedene korrekte Lösungen), z. B.:



Zu Aufgabe 20: (Huffman) (Forts.)

(2) Code-Liste erstellen:

Symbol	*Kodierung*	*Häufigkeit*
E	0000	1
f	0001	1
h	001	2
i	10	5
l	010	2

Symbol	*Kodierung*	*Häufigkeit*
n	011	2
o	11000	1
t	11001	1
x	1101	1
␣	111	3

(3) Text umkodieren:

0000 1101 111 011 10 001 10 010 11000 111 011
10 001 10 010 111 0001 10 11001

Für den reinen Text benötigen wir 59 Bits in der komprimierten Darstellung.
Damit erhalten wir eine mittlere Codewortlänge von 3,105 ($= 59 / 19$).

Aufgabe 21: (WWW)

Nehmen Sie an, es gibt im World-Wide-Web 10 Millionen Dokumente, die im Mittel 10 Hyperlinks besitzen. Wie lange dauert es (mindestens) das gesamte Web zu indexieren, wenn das Holen eines Dokumentes durchschnittl. 100 ms dauert, bei

(a) rein sequenziellen Zugriffen

(b) einem *Multi-Threaded*-Client mit beliebig hoher Parallelität und Datenrate?

Zu Aufgabe 21: (WWW)

(a) sequenziell

Das Laden aller Dokumente dauert $10^7 \cdot 100\ \boldsymbol{ms} \approx 11,5$ Tage.

(b) mit Multi-Threading

Jedes Dokument enthält zehn (verschiedene) Links, die anschließend parallel geladen werden können:
1 Dok. $\rightarrow$ 10 Dok. $\rightarrow$ 100 Dok. $\rightarrow$ ... $\rightarrow$ 10^7 Dok.

Also dauert es $8 \cdot 100\ \boldsymbol{ms} = 0,8\boldsymbol{s}$
(bei beliebig hoher Parallelität und Leistungsfähigkeit)