# Tutoriumsblatt 4 mit Musterlösung

## Aufgabe 4.1: Schrittgeschwindigkeit und Ausbreitungsgeschwindigkeit

Zwei Parameter, welche die Dauer einer Datenübertragung beeinflussen, sind die Schrittgeschwindigkeit und die Ausbreitungsgeschwindigkeit.

Die Schrittgeschwindigkeit bestimmt, wie viele Signale pro Sekunde auf ein Medium gegeben werden können. Aus diesem Parameter ergibt sich die Datenrate, mit der übertragen werden kann.

Die Ausbreitungsgeschwindigkeit bestimmt, mit welcher Geschwindigkeit sich ein Signal auf dem Medium ausbreitet. Aus diesem Parameter ergibt sich die Latenz, d.h. die Verzögerung, mit der Daten (Signale) über eine Leitung zugestellt werden.

Überlegen Sie: *welchen dieser beiden Parameter halten Sie für kritischer für die Gesamtdauer einer Datenübertragung*?

## Lösung 4.1

Die Aufgabe ist bewußt sehr allgemein gehalten! Es hängt immer vom konkreten Szenario ab, welcher der Parameter kritischer ist. Nehmt diese URL her:

https://www.ccs-labs.org/teaching/rn/animations/propagation/

Auf dieser Seite kann man mal mit der Datenrate, der Entfernung und der Paketgröße spielen und sehen, dass je nach Kombination das erste Bit schon beim Empfänger ankommt, bevor das letzte Bit abgesendet wurde bzw. alles abgesendet wurde, lange bevor das erste Bit ankommt.

Es gibt also eigentlich keine Antwort auf die hier gestellt Frage. Wir wollen nur, dass der Einfluß beider Parameter klar wird:

Die Schrittgeschwindigkeit bestimmt, innerhalb welcher Zeit eine bestimmte Datenmenge versendet werden kann (= Sendedauer), die Ausbreitungsgeschwindigkeit bestimmt, innerhalb welcher Zeit ein Signal übertragen wird (= Latenz). Die Gesamtdauer einer Datenübertragung (auf Schicht 1) ist die Summe beider Werte: die Zeitdauer zur Versendung aller Bits + die Laufzeit eines Bits hin zum Empfänger.

## Aufgabe 4.2: Flusskontrolle und Fehlerbehandlung

Es gibt verschiedene Verfahren, um Übertragungswiederholungen und Flusskontrolle umzusetzen. Welche davon verwendet werden, beinflusst direkt die Leistungsfähigkeit eines Protokolls der Sicherungsschicht.

a) Im Folgenden soll das Verfahren *Stop-and-Wait* benutzt werden um eine 2 MByte große Datei zu übertragen. Allerdings dürfen in einem Rahmen maximal 750 Byte übetragen werden. Die Datenrate des verwendeten Mediums betrage 6 MBit/s. Die Latenz zwischen Sender und Empfänger der Datei betrage in beide Richtungen 20 Millisekunden.

*Wie lange dauert die Übertragung der Datei (vom Beginn der Übertragung bis hin zum Zeitpunkt, zu dem der Sender sich sicher sein kann, dass die Übertragung der Datei korrekt abgeschlossen wurde)?*

*Hinweise*: Es treten keine Übertragungsfehler auf. Vernachlässigen Sie die Verarbeitungszeiten auf Sender- und Empfängerseite. Gehen Sie also z.B. davon aus, dass eine Bestätigung ohne weitere Verzögerung direkt nach Empfang eines vollständigen Rahmens gesendet werden kann. Vernachlässigen Sie weiterhin die Kontrollinformationen in den Headern in beiden Richtungen, d.h. Sie können die 750 Byte innerhalb eines Rahmens vollständig für Nutzdaten verwenden sowie die Sendedauer der Bestätigungen ignorieren.

b) Oft kommt es vor, dass im praktischen Einsatz eines Protokolls nicht die volle Kapazität einer Leitung ausgenutzt werden kann. Das Verhältnis von erreichter zu maximal erreichbarer Datenrate auf einem Medium wird als Auslastung oder auch als Effizienz bezeichnet. *Welche Auslastung wird in der Situation aus a) erreicht?*

c) Statt Stop-and-Wait wird nun ein Sliding-Window-Verfahren mit Go-Back-N zur Übertragungswiederholung verwendet. Sie überlegen, ob Sie eine Fenstergröße von 4 Rahmen oder besser eine Fenstergröße von 160 Rahmen verwenden sollten. *Berechnen Sie daher für beide Fenstergrößen jeweils die Auslastung.*

## Lösung 4.2

### 2.a)

Das Versenden von 750 Byte Daten dauert bei 6 MBit/s eine nicht vernachlässigbare Zeit. Zunächst rechnen wir Byte in Bit um:

$$750\,\text{Byte} = 8 \cdot 750\,\text{Bit} = 6000\,\text{Bit}.$$

Damit ergibt sich die Sendedauer $T_S$:

$$T_S = \frac{6000\,\text{Bit}}{6\,\text{MBit/s}} = \frac{6 \cdot 10^3\,\text{Bit}}{6 \cdot 10^6\,\text{Bit/s}} = 0,001\,\text{s} = 1\,\text{ms}.$$

Nachdem wir 750 Byte gesendet haben, müssen wir eine RTT warten. Diese ist das Doppelte der Latenz:

$$T_{RTT} = 40\,\text{ms}.$$

Damit dauert es

$$T_S + T_{RTT} = 41\,\text{ms},$$

bis wir nach Stop-and-Wait den nächsten Rahmen versenden können.

Wenn wir 2 MByte in 750 Byte-Stücke aufteilen, kommen wir auf 2667 Rahmen, die gesendet werden müssen, um die ganze Datei zu Übertragen.

Damit dauert die Übertragung insgesamt

$$2667\,\text{Rahmen} \cdot 41\,\text{ms} = 109,347\,\text{Sekunden}.$$

Wenn man es genau nimmt, ist es etwas weniger, da Rahmen 2667 weniger als 750 Byte enthält. Wir brauchen also hier weniger als 1 ms.

**2.b)**

Wir übertragen 2 MByte in 109,347 Sekunden. Bei voller Nutzung von 6 MBit/s hätte man in 109,347 Sekunden allerdings

$$109,347\,\text{s} \cdot 6\,\text{MBit/s} \approx 656\,\text{MBit} = 82\,\text{MByte}$$

übertragen können.

Damit ergibt sich folgende Effizienz:

$$\frac{2\,\text{MByte}}{82\,\text{MByte}} \approx 0,0244 (= 2,44\%).$$

*Alternative Berechnung:* Man kann die Effizienz auch über das Verhältnis von Sendedauer zu gesamter Dauer bis zur Ankunft einer Quittung berechnen:

$$\frac{T_S}{T_S + T_{RTT}} = \frac{1\,\text{ms}}{41\,\text{ms}} \approx 2,44\,\%.$$

**2.c)**

Aus Aufgabenteil a) wissen wir, dass 2667 Rahmen zu übertragen sind und dass es 41 ms dauert, bis der Sender die Quittung (Empfangsbestätigung) des Empfängers für einen gesendeten Rahmen erhält.

**Fenstergröße 4:**

Bei einer Fenstergröße von 4 können wir vier Rahmen sofort hintereinander senden. Wir erhalten die Quittung für den ersten gesendeten Rahmen nach 41 ms. Dann können wir sofort den fünften Rahmen senden. Direkt nachdem wir den fünften Rahmen gesendet haben, erhalten wir die Quittung für den zweiten Rahmen und können den nächsten Rahmen senden. Genauso verfahren wir jeweils beim Erhalt der Quittung für den dritten und vierten Rahmen. Die Quittung für den fünften Rahmen empfangen wir 41 ms nach dem Versenden des Rahmens, d.h. nach 82 ms senden wir den neunten Rahmen usw.

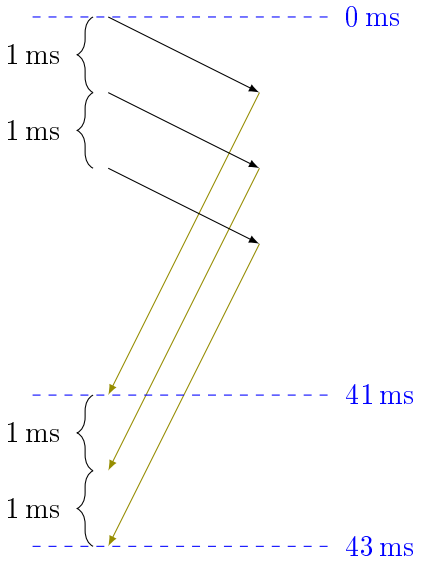
Also brauchen wir zum Versenden aller Rahmen

$$\left\lceil \frac{n}{m} \right\rceil = \left\lceil \frac{2667}{4} \right\rceil \cdot 41\,\text{ms} = 27,347\,\text{s},$$

wobei $n$ die Anzahl der zu sendenden Rahmen und $m$ die Fenstergröße ist.

Das kann man als Näherung gerne so stehen lassen.

Wenn man es ganz exakt machen will, muss man allerdings berücksichtigen, dass man beim letzten Block noch auf alle Quittungen warten muss. Bei vorheriger Rechnung muss man nämlich eigentlich berücksichtigen, dass der Empfänger einen Rahmen nach dem anderen bekommt und genauso quittiert – der Sender bekommt also auch eine Quittung nach der anderen. Die 41 ms geben die Zeit bis zum Empfang der ersten Quittung an, was oben ausreichend ist, da der Sender beim Empfang der ersten Quittung schon anfangen kann, den nächsten Rahmen zu senden. Die zweite Quittung kommt so rechtzeitig, dass der Sender direkt den nächsten Rahmen senden kann. Drum kann man die Rechnung oben so einfach gestalten. Beim letzten Block an Rahmen, der versendet wird, muss man aber genau genommen noch berücksichtigen, wie viel länger als 41 ms es dauert, die Quittungen für die letzten (beiden) Rahmen zu erhalten. Im Bild ist das beispielhaft dargestellt. Hier werden drei Rahmen gesendet.




Das korrekte Ergebnis ist also

$$\begin{aligned}&\left\lceil \tfrac{n}{m} \right\rceil \cdot T_{\text{ACK}} + [(n-1) \mod m] \cdot T_S \\ &= \left\lceil \tfrac{2667}{4} \right\rceil \cdot 41\,\text{ms} + [(2667-1) \mod 4] \cdot 1\,\text{ms} \\ &= 27,347\,\text{s} + 0,002\,\text{s} = 27,349\,\text{s}\end{aligned}$$

dabei ist $T_{\text{ACK}}$ die Zeit, bis wir die Quittung zu einem gesendeten Rahmen erhalten, und $T_S$ die Sendedauer, die benötigt wird, um einen Rahmen auf die Leitung zu setzen.

In der Zeit, die wir für das Senden von 2 MByte benötigt haben, hätten wir

$$27,349\,\text{s} \cdot 6\,\frac{\text{MBit}}{\text{s}} = 164.094.000\,\text{Bit} \approx 20,5\,\text{MByte}$$

senden können. Also haben wir eine Auslastung von

$$\frac{2\,\text{MByte}}{20,5\,\text{MByte}} \approx 0,0976 \approx 9,8\,\%.$$

**Fenstergröße von 160:**

Hier können wir 160 ms am Stück senden. Die mögliche Sendedauer ist größer als die Zeit bis zum Erhalt der Quittungen. Daher können wir durchgehend senden. Die Auslastung liegt also bei 100 %.

Es würde reichen, 41 ms am Stück zu senden. Was bewirkt die Überdimensionierung auf das Vierfache des Wertes? Hier in der Rechnung nicht viel, aber in der Praxis, wo Übertragungsfehler vorkommen können, kann es sein, dass wir viele Rahmen geschickt haben, bevor wir die (negative) Quittung erhalten, mittels Go-Back-N noch einmal den ersten Rahmen zu senden – so dass wir das Netz lange blockiert haben, obwohl es nicht nötig war. Also: zu große Flusskontrollfenster können das Netz unnötig belasten. Darauf kommen wir noch mal bei TCP zurück, wo das Fenster durch Empfänger und Netz quasi gemeinsam bestimmt wird.

Zudem braucht der Sender einen Buffer, der der Größe des Fenster entspricht. Wir verschwenden also auch Bufferspeicher, wenn wir das Fenster zu groß wählen.

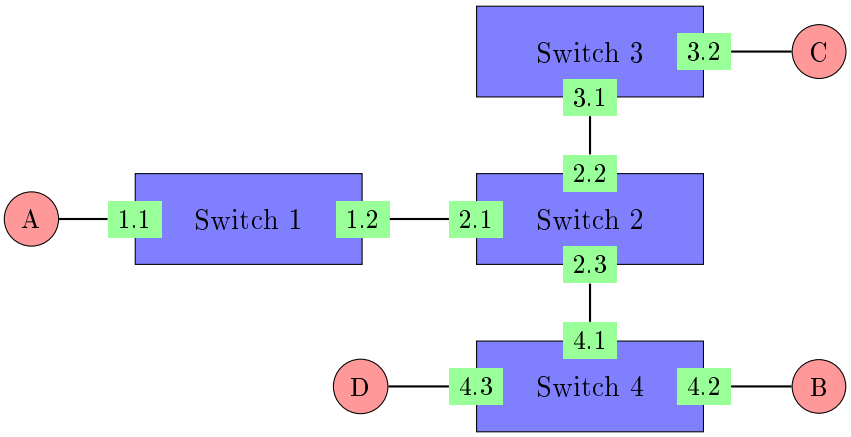
Hier ist Flusskontrolle mit Go-Back-N zur Fehlerbehandlung visualisiert: http://www.ccs-labs.org/teaching/rn/animations/gbn_sr/.

Aber Vorsicht – die Animationen sind nicht ganz korrekt. Nach einem Übertragungsfehler werden alle Rahmen innerhalb des Fensters gleichzeitig neu gesendet statt schön nacheinander.

## Aufgabe 4.3: Hubs und Switches

a) Welche Vorteile haben Switches gegenüber Hubs?

b) Switches lernen die Adressen angeschlossener Rechner. Betrachten Sie folgendes Netzwerk:




Die Weiterleitungstabellen der Switches seien leer. Jetzt werden nacheinander folgende Rahmen gesendet:

1. $A$ sendet an $C$.
2. $C$ sendet an $A$.
3. $B$ sendet an $C$.

Wie sehen die Weiterleitungstabellen der Switches 1 bis 4 nach Versenden dieser Rahmen aus? *Geben Sie für jeden der drei versendeten Rahmen an, welche Switches und Hosts durchlaufen werden und welche Einträge die Switches in ihren Weiterleitungstabellen erzeugen.* Die Einträge altern nicht, d.h., sie bleiben ewig bestehen.

c) *Welche Funktionen könnten moderne Switches noch bieten?*

## Lösung 4.3

a) Hubs unterbrechen die Kollisionsdomäne nicht, d.h. empfangene Signale werden auf allen anderen Ports gebroadcastet. Also darf nur eine Station gleichzeitig senden. Daher wird ein Medienzugriffsverfahren wie z.B. CSMA/CD benötigt. Switches hingegen trennen die Kollisionsdomänen. Wenn Switches über entsprechend großen Puffer verfügen, gibt es keine Kollisionen mehr. Mehrere Station können gleichzeitig senden bzw. empfangen. An Switches können Rechner mit verschiedenen Datenraten angeschlossen werden –die langsamste Station bremst die anderen nicht aus.

b) 1. Es liegen noch keine Adressinformationen vor. $A$ sendet an $C$. Switch 1 broadcastet den Rahmen auf Port 1.2. Switch 2 broadcastet den Rahmen auf Port 2.2 und 2.3. Switch 3 broadcastet den Rahmen auf Port 3.2. Switch 4 broadcastet den Rahmen auf Port 4.2. Jeder Switch merkt sich jeweils die Absenderadresse des Rahmens ($A$) zusammen mit dem Port, über den der Rahmen empfangen wurde.

   Folgende Switches werden durchlaufen: 1, 2, 3, 4. Alle Hosts (A, B, C, D) bekommen die Daten.

   Weiterleitungstabellen der Switches:

   - Switch 1: ($A$, Port 1.1)
   - Switch 2: ($A$, Port 2.1)
   - Switch 3: ($A$, Port 3.1)
   - Switch 4: ($A$, Port 4.1)

2. $C$ sendet an $A$. Da die Switches sich vorher gemerkt haben auf welchem Port sie Rahmen an $A$ weiterleiten müssen, kann das Paket direkt zugestellt werden. Alle Switches, die den Rahmen weiterleiten, merken sich die Absenderadresse des Rahmens ($C$) zusammen mit dem Quellport, über den der Rahmen empfangen wurde.

   Folgende Switches werden durchlaufen: 3, 2, 1. Nur Host A erhält die Daten. Weiterleitungstabellen der Switches:

   - Switch 1: ($A$, Port 1.1), ($C$, Port 1.2)
   - Switch 2: ($A$, Port 2.1), ($C$, Port 2.2)
   - Switch 3: ($A$, Port 3.1), ($C$, Port 3.2)
   - Switch 4: ($A$, Port 4.1)

3. $B$ sendet an $C$. Switch 4 hat noch keinen Eintrag für $C$ in seiner Weiterleitungstabelle. Daher broadcastet Switch 4 den Rahmen. Switch 2 hat schon einen Eintrag und kann den Rahmen gezielt auf Port 2.2 weiterleiten. Switch 3 kann den Rahmen ebenfalls gezielt auf Port 3.2 weiterleiten. Alle Switches, die den Rahmen weiterleiten, merken sich die Absenderadresse des Rahmens ($B$) zusammen mit dem Quellport, über den der Rahmen empfangen wurde.

   Folgende Switches werden durchlaufen: 4, 2, 3. Neben C erhält auch D die Daten.

   Weiterleitungstabellen der Switches:

   - Switch 1: ($A$, Port 1.1), ($C$, Port 1.2)
   - Switch 2: ($A$, Port 2.1), ($C$, Port 2.2), ($B$, Port 2.3)
   - Switch 3: ($A$, Port 3.1), ($C$, Port 3.2), ($B$, Port 3.1)
   - Switch 4: ($A$, Port 4.1), ($B$, Port 4.2)

   Anhand des Ablaufs kann man schön sehen: Adressinformationen werden quasi ins ganze Netz geflutet. Ist eine Adresse einmal bekannt, wird Verkehr aber gefiltert, Kollisionsdomänen werden dadurch verkleinert. Hat man nur einen einzelnen Rechner in einem Segment, ist die Kollisionsdomäne auch auf diesen beschränkt, Kollisionen treten gar nicht mehr auf. (Falls die Switches Rahmen auch buffern können und daher vermeiden, dass Daten, die über zwei Ports eingehen und an den gleichen Port weitergeleitet werden sollen, im Switch kollidieren.)

   Aber: da bei bekannter Adresse kein Broadcast mehr stattfindet, erhält in diesem Beispiel Switch 4 keine Adressinformation zu $C$. Daher muss dieser Switch den dritten Rahmen ($B$ an $C$) broadcasten. (Gut, in unserem Beispiel ist das langweilig, da eh nur ein Port existiert. Aber es geht ums Prinzip.)

c) Switches werden heute oft als „Datensammler“ eingesetzt. Z.B. werden Daten wie Bandwidth Usage, Collision Rates, oder Traffic Types gesammelt. Diese werden von den Switches zur weiteren Analyse exportiert. Moderne Switches enthalten sogar Firewalls, welche auf der Vermittlungsschicht operieren oder implementieren sogar noch höhere Schichten, wenn die Pakete genauer analysiert werden sollen.

Wenn wir von „Switches“ reden, meinen wir reine Layer-2-Switche zur Datenweiterleitung. All die Zusatzfunktionalität berücksichtigen wir nicht, da sie zu anderen Schichten gehört oder spezielle Anwendungen ermöglicht.