



Rechnernetze I, WS 2004/2005

Fenstermechanismen zur Fehlerkontrolle (Ergänzung zum Vorlesungsskript)

Andreas Schorr



QoS Group

Stop-and-Wait

- **Sender schickt ein nummeriertes Paket und wartet dann auf ein Ack (Acknowledge), bevor er das nächste Paket überträgt.**
 - **Empfänger akzeptiert nur fehlerfreie Pakete mit der erwarteten Sequenznummer. Bei jedem Paketempfang sendet er ein Ack für das zuletzt fehlerfrei und in richtiger Reihenfolge empfangene Paket.**
 - **Sender überträgt letztes Paket erneut, wenn ein Timeout abgelaufen ist.**
- **Benötigt Puffer der Größe 1 auf Senderseite.**
- **Paketnummerierung modulo 2.**

Stop-and-Wait mit Nack

Wie Stop-and-Wait aber:

- Empfänger schickt Nack (Negative Acknowledge) für fehlerhafte Pakete.
 - Sender überträgt letztes Paket erneut, wenn
 - a) ein Timeout abgelaufen ist.
 - b) Nack für das letzte Paket angekommen ist.
- Bessere Effizienz als Stop-and-Wait ohne Nack, da bei fehlerhafter Übertragung bereits nach Empfang des Nack weitergesendet werden kann, auch wenn der Timeout noch nicht abgelaufen ist.
- Benötigt Puffer der Größe 1 auf Senderseite.
- Paketnummerierung modulo 2.

Go-Back-N

- Sender benutzt ein Sendefenster der Größe $w > 1$.
 - Sender schickt w Pakete, ohne auf ein Ack zu warten.
 - Empfänger akzeptiert nur fehlerfreie und in richtiger Reihenfolge empfangene Pakete. Bei jedem Paketempfang sendet er ein Ack für das zuletzt korrekt und in richtiger Reihenfolge empfangene Paket.
 - Sender überträgt Paket n erneut (und auch alle folgenden bereits geschickten Pakete), wenn Timeout für Paket n abgelaufen ist.
 - Wenn Ack für Paket n beim Sender ankommt, schiebt dieser das Sendefenster auf die Position $n+1$.
-
- Benötigt Puffer der Größe w auf Senderseite.
 - Paketnummerierung modulo $w+1$.

Go-Back-N mit Nack

Wie Go-Back-N aber:

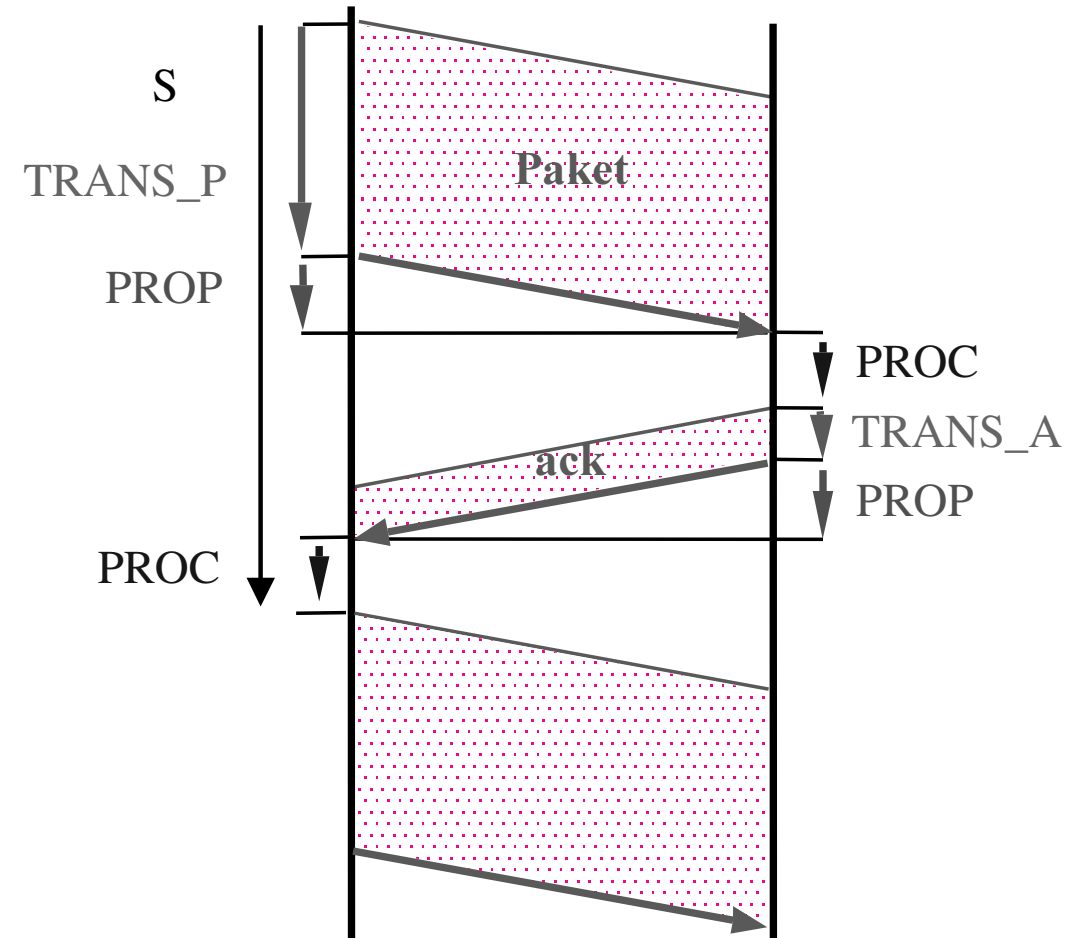
- Empfänger schickt Nack (Reject) für fehlerhafte Pakete.
 - Sender überträgt Paket n erneut (und auch alle folgenden bereits geschickten Pakete), wenn
 - a) Timeout für Paket n abgelaufen ist.
 - b) Nack für Paket n angekommen ist.
-
- Bessere Effizienz als Go-Back-N ohne Nack, da bei fehlerhafter Übertragung bereits nach Empfang des Nack weitergesendet werden kann, auch wenn der Timeout noch nicht abgelaufen ist.
 - Benötigt Puffer der Größe w auf Senderseite.
 - Paketnummerierung modulo $w+1$.

Selective Reject (oder: Selective Repeat)

- Verwendet Sende- und Empfangsfenster der Größe $w > 1$.
 - Sender schickt w Pakete, ohne auf ein Ack zu warten.
 - Empfänger akzeptiert jedes fehlerfreie Paket innerhalb des Empfangsfensters und bestätigt mit Ack. Bei fehlerhaften Paketen wird ein Nack geschickt.
 - Falls ein empfangenes Paket die kleinste Sequenznummer im Empfangsfenster hat, verschiebt der Empfänger das Fenster bis zur nächsten noch nicht empfangenen Sequenznummer.
 - Sender überträgt nur Paket n erneut, wenn
 - a) Timeout für Paket n abgelaufen ist.
 - b) Nack für Paket n angekommen ist.
 - Wenn Ack für Paket n beim Sender ankommt, markiert dieser die Position n als korrekt empfangen. Falls n die kleinste Sequenznummer im Sendefenster ist, schiebt der Sender das Fenster auf die nächste unbestätigte Position.
- Benötigt Puffer der Größe w auf Sender- und Empfängerseite.
- Paketnummerierung modulo $2w$.

Effizienz von Stop-and-Wait

- **TRANS_P:**
Übertragungszeit eines Datenpaketes = $\text{Bits pro Paket} / \text{Übertragungsrate}$
- **PROP:**
Signallaufzeit
- **TRANS_A:**
Übertragungszeit eines Ack
- **PROC:**
Verarbeitungszeit einer Nachricht (typischerweise vernachlässigbar)
- **S:**
mittlere Zeit zwischen 2 Paketübertragungen
- **Effizienz eines Protokolls:**
Verhältnis zwischen eigentlicher Übertragungszeit und Gesamtzeit = $\text{TRANS_P} / S$



Effizienz von Stop-and-Wait

- **TRANS_A und PROC seien vernachlässigbar $\rightarrow S = TRANS_P + 2 * PROP$**
- **Annahme: Es treten keine Fehler auf.**
 - **Effizienz $U = TRANS_P / S = TRANS_P / (TRANS_P + 2 * PROP)$**
 - **Beispiel 1: Pakete der Länge 1500 Byte, 100 Mbps, Ausbreitungsgeschwindigkeit $2 * 10^8$ m/s, Leitungslänge 10 km**
 $\rightarrow U = 0.54$
 - **Beispiel 2: Pakete der Länge 1500 Byte, 56 Kbps, Ausbreitungsgeschwindigkeit $2 * 10^8$ m/s, Leitungslänge 100 km**
 $\rightarrow U = 0.99$
- **Im Fehlerfall:**
 - **Timeout = $2 * PROP$**
 - **Wahrscheinlichkeit, dass Paketübertragung wiederholt werden muss = p**
 - **Effizienz $U = ((1 - p) * TRANS_P) / (TRANS_P + 2 * PROP)$**
- **Durchsatz = Übertragungsrate * Effizienz**
 - **Beispiel: Durchsatz $e(100 \text{ Mbit}, 0.54) = 100 \text{ Mbps} * 0.54 = 54 \text{ Mbps}$**

Effizienz von Go-Back-N

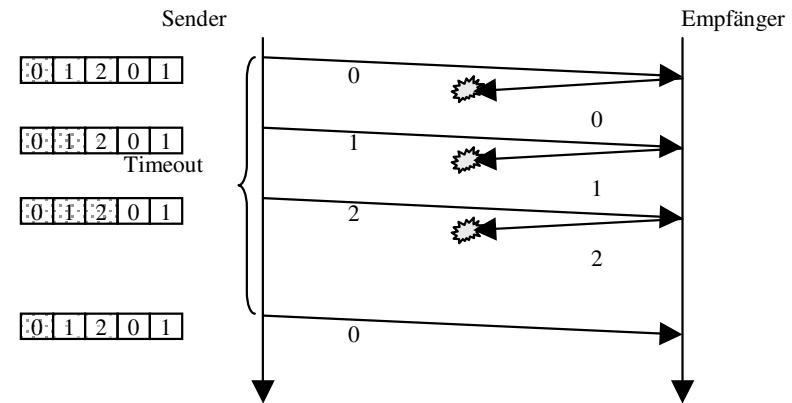
- Effizienz abhängig von Fenstergröße w . Falls immer bereits ein Ack angekommen ist, bevor das Senderfenster voll ist, dann kann der Sender immer senden.
- Zeit zum Senden eines ganzen Fensters: $TRANS_P * w$
- Wartezeit auf erstes Ack: $TRANS_P + 2 * PROP$
- \rightarrow Sender kann immer senden, falls $TRANS_P + 2 * PROP \leq TRANS_P * w$
 $\Leftrightarrow w \geq 1 + 2 * (PROP / TRANS_P)$
- Annahme: Es treten keine Fehler auf.
 - $U = 1,$ falls $w \geq 1 + 2 * (PROP / TRANS_P)$
 - $U = w / (1 + 2 * (PROP / TRANS_P)),$ falls $w < 1 + 2 * (PROP / TRANS_P)$
- Im Fehlerfall:
 - Wahrscheinlichkeit, dass Paketübertragung wiederholt werden muss = p
 - $U = (1 - p) / (1 + 2 * p * (PROP / TRANS_P)),$ falls $w \geq 1 + 2 * (PROP / TRANS_P)$
 - $U = ((1 - p) * w) / ((2 * (PROP / TRANS_P) + 1) * (1 - p + wp)),$
 falls $w < 1 + 2 * (PROP / TRANS_P)$

Effizienz von Selective Reject

- Im fehlerfreien Fall wie bei Go-Back-N.
- Im Fehlerfall:
 - Wahrscheinlichkeit, dass Paketübertragung wiederholt werden muss = p
 - $U = 1 - p$, falls $w \geq 1 + 2 * (PROP / TRANS_P)$
 - $U = ((1 - p) * w) / (2 * (PROP / TRANS_P) + 1)$, falls $w < 1 + 2 * (PROP / TRANS_P)$

Sequenznummernraum

Go-Back-N, $w=3$
 normal: modulo $w+1$
 hier: modulo w



Selective-Reject, $w=3$
 normal: modulo $2w$
 hier: modulo $2w-1$

