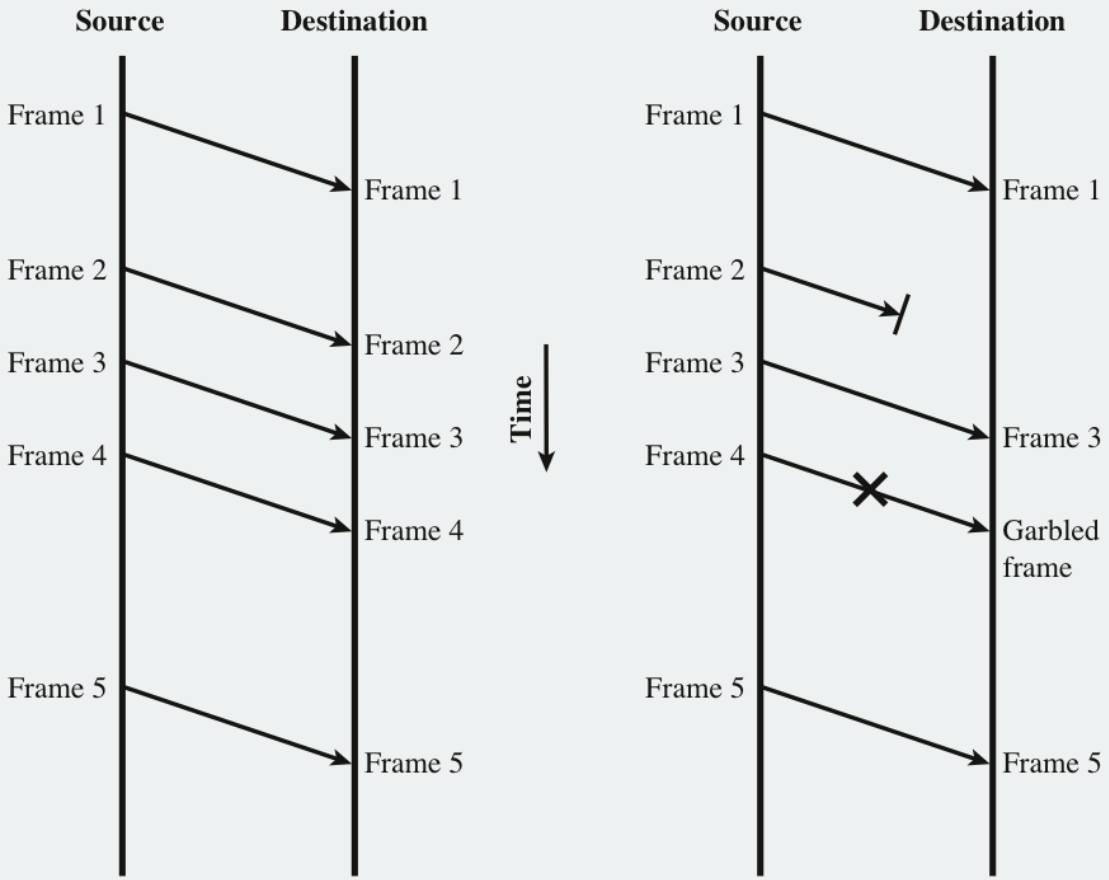
Communication and Networking
Error Control Basics

D. Richard Brown III

(selected figures from Stallings Data and Computer Communications 10th edition)

Lost and Damaged Frames and ACKs

1. Lost frame
   - 1.1 Network did not deliver frame
   - 1.2 Frame was so damaged that it was not even recognized as a frame
2. Damaged frame
   - 2.1 Frame recognized but failed parity, checksum, or CRC
3. Lost ACK
4. Damaged ACK



(a) Error-free transmission

(b) Transmission with losses and errors

Figure 7.1 Model of Frame Transmission

Automatic Repeat and Request (ARQ)

Three common versions of ARQ:

1. Stop-and-wait ARQ
2. Go-back-N ARQ
3. Selective-reject ARQ

All employ some combination of:

1. Error detection and discarding frames with errors
2. Positive acknowledgement (like flow control)
3. Negative acknowledgment and retransmission
4. Timeout and retransmission

The net effect of ARQ is to turn an unreliable data link into a reliable one (but in a different way that forward error correction).

Error Control Timing Notation

Notation (similar to what we saw earlier for flow control):

t_{prop} : propagation time of data link (assumed to be the same in both directions)
t_{frame} : frame transmission time
t_{proc} : processing time at TX and RX to react to frame or ACK
t_{ack} : acknowledgement transmission time
t_{timeout} : ACK timeout interval
n : number of frames

Link utilization:

$$U = \frac{\text{time spent transmitting data in } n \text{ frames}}{\text{total time required to send } n \text{ frames and receive ACKs}} \leq 1.$$

Stop-and-Wait ARQ

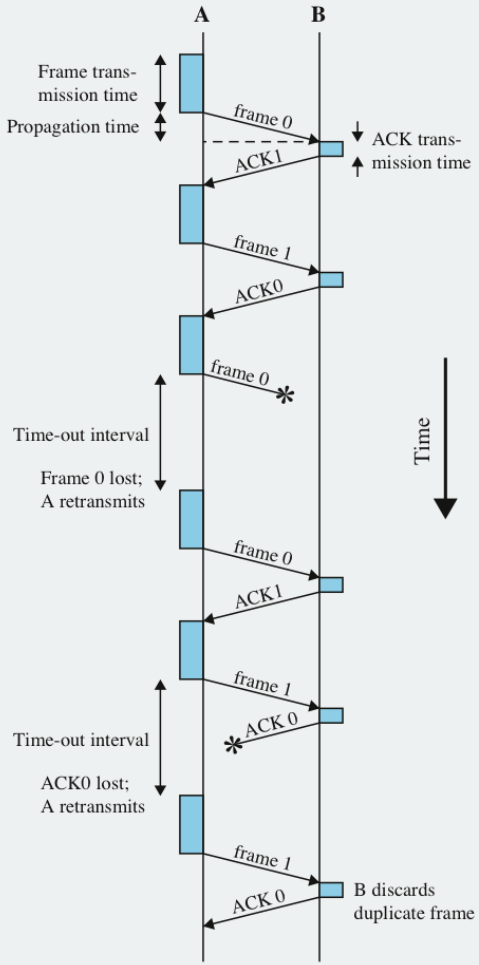
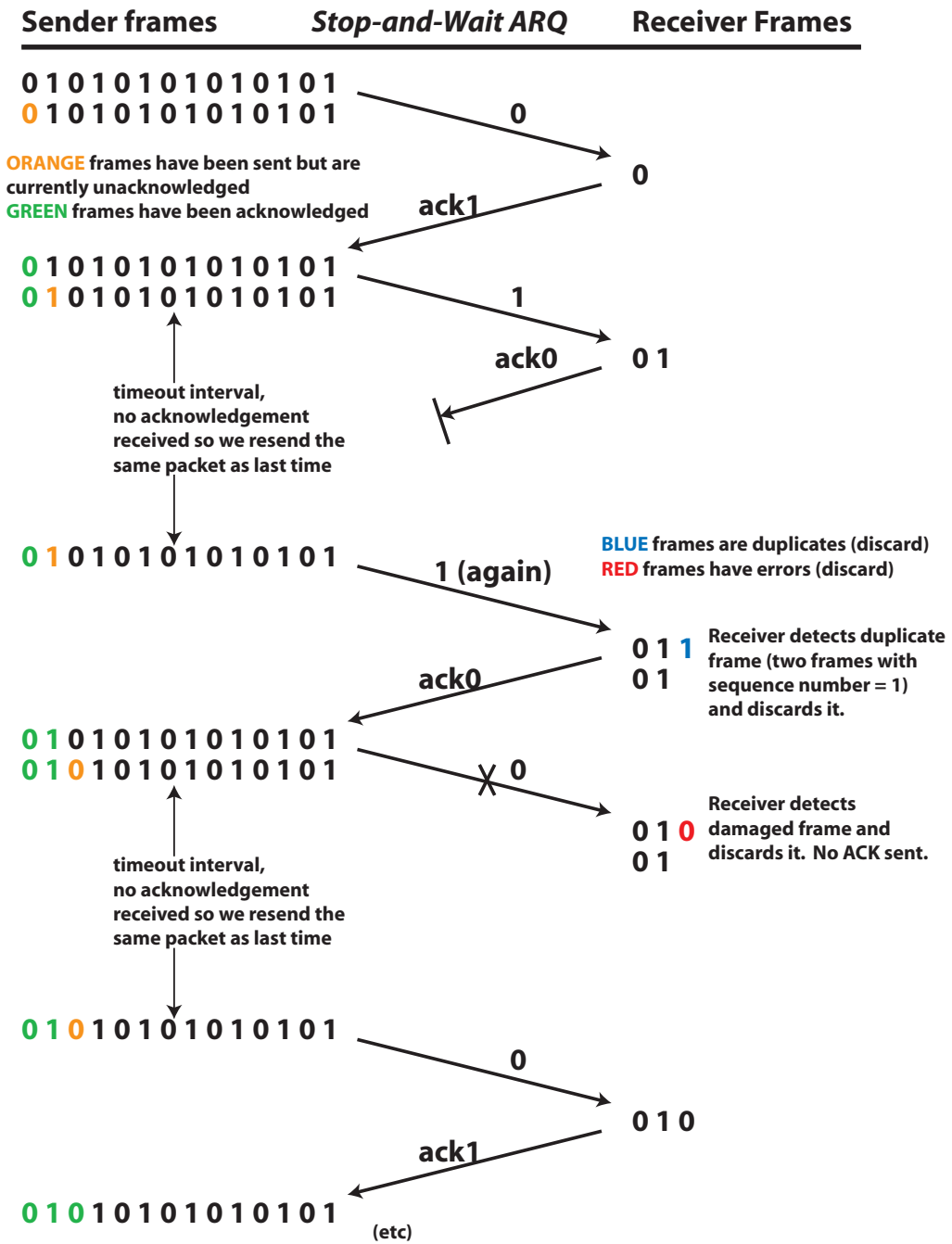


Figure 7.5 Stop-and-Wait ARQ

Steps:

1. A transmits frame
2. A waits for ACK
3. If frame is damaged, B discards it.
4. A resends frame if ACK timeout period is exceeded.
5. In the case of a lost/damaged ACK:
   - A resends frame
   - Receiver has two copies of the frame
   - Can use ACK0/ACK1 to determine if frame is new or old


Sender frames
Stop-and-Wait ARQ
Receiver Frames
0 1 0 1 0 1 0 1 0 1 0 1 0 1
0 1 0 1 0 1 0 1 0 1 0 1 0 1
0
ORANGE frames have been sent but are currently unacknowledged
GREEN frames have been acknowledged
0
ack1
0 1 0 1 0 1 0 1 0 1 0 1 0 1
0 1 0 1 0 1 0 1 0 1 0 1 0 1
1
ack0
0 1
timeout interval, no acknowledgement received so we resend the same packet as last time
0 1 0 1 0 1 0 1 0 1 0 1 0 1
1 (again)
BLUE frames are duplicates (discard)
RED frames have errors (discard)
0 1 1
0 1
Receiver detects duplicate frame (two frames with sequence number = 1) and discards it.
ack0
0 1 0 1 0 1 0 1 0 1 0 1 0 1
0 1 0 1 0 1 0 1 0 1 0 1 0 1
0
0 1 0
0 1
Receiver detects damaged frame and discards it. No ACK sent.
timeout interval, no acknowledgement received so we resend the same packet as last time
0 1 0 1 0 1 0 1 0 1 0 1 0 1
0
0 1 0
ack1
0 1 0 1 0 1 0 1 0 1 0 1 0 1
(etc)

Stop-and-Wait ARQ Timing Analysis

In the previous example, the total time to transmit three frames was

$$
\begin{aligned}
T &= T_1 + T_2 + T_3 \\
&= \underbrace{t_{\text{frame}} + t_{\text{prop}} + t_{\text{proc}} + t_{\text{ack}} + t_{\text{prop}} + t_{\text{proc}}}_{\text{first frame}} + \\
&\quad \underbrace{t_{\text{frame}} + t_{\text{timeout}} + t_{\text{frame}} + t_{\text{prop}} + t_{\text{proc}} + t_{\text{ack}} + t_{\text{prop}} + t_{\text{proc}}}_{\text{second frame}} \\
&\quad \underbrace{t_{\text{frame}} + t_{\text{timeout}} + t_{\text{frame}} + t_{\text{prop}} + t_{\text{proc}} + t_{\text{ack}} + t_{\text{prop}} + t_{\text{proc}}}_{\text{third frame}} \\
&= 3T_f + 2(t_{\text{timeout}} + t_{\text{frame}})
\end{aligned}
$$

where $T_f = t_{\text{frame}} + 2t_{\text{prop}} + 2t_{\text{proc}} + t_{\text{ack}}$ is the time to successfully transmit a frame and receive its ACK.

Hence, for stop-and-wait ARQ each lost/damaged frame or ACK adds $t_{\text{timeout}} + t_{\text{frame}}$ to the time required to successfully transmit a frame.

Stop-and-Wait ARQ: Average Number of Transmissions

Let P denote the probability of successfully transmitting a frame and receiving the ACK. The probability that it takes k attempts to successfully send a frame and receive the ACK is equal to the probability that the first $k-1$ attempts were unsuccessful and the k^{th} attempt was successful:

$$\text{Prob}[\text{frame+ACK successfully received after } k \text{ attempts}] = P^{k-1}(1-P).$$

The **average** number of transmissions required to successfully send a frame is then

$$\text{E}[\text{transmissions}] = \sum_{k=1}^{\infty} kP^{k-1}(1-P) = (1-P)\sum_{k=1}^{\infty} kP^{k-1}$$

Note that, for $|\alpha| < 1$, we have

$$\sum_{k=0}^{\infty} \alpha^k = \frac{1}{1-\alpha}$$

and it follows that

$$\frac{d}{d\alpha}\sum_{k=0}^{\infty}\alpha^k = \frac{d}{d\alpha}\left(1+\alpha+\alpha^2+\ldots\right) = 0+1+2\alpha+\cdots = \sum_{k=1}^{\infty} k\alpha^{k-1} = \frac{d}{d\alpha}\left(\frac{1}{1-\alpha}\right) = \frac{1}{(1-\alpha)^2}.$$

Hence $\text{E}[\text{transmissions}] = (1-P)\frac{1}{(1-P)^2} = \frac{1}{1-P}$.

Stop-and-Wait ARQ: Average Link Utilization

If we select t_{timeout} such that

$$t_{\text{timeout}} \approx t_{\text{prop}} + t_{\text{proc}} + t_{\text{ack}} + t_{\text{prop}} + t_{\text{proc}}$$

then $t_{\text{timeout}} + t_{\text{frame}} \approx T_f = t_{\text{frame}} + 2t_{\text{prop}} + 2t_{\text{proc}} + t_{\text{ack}}$. In other words, the total time of each unsuccessfully transmitted frame is approximately the same as each successfully transmitted frame.

The average time per frame is then

$$\text{E[time per frame]} = T_f \cdot \text{E[transmissions]} = \frac{t_{\text{frame}} + 2t_{\text{prop}} + 2t_{\text{proc}} + t_{\text{ack}}}{1 - P}$$

and the average link utilization is

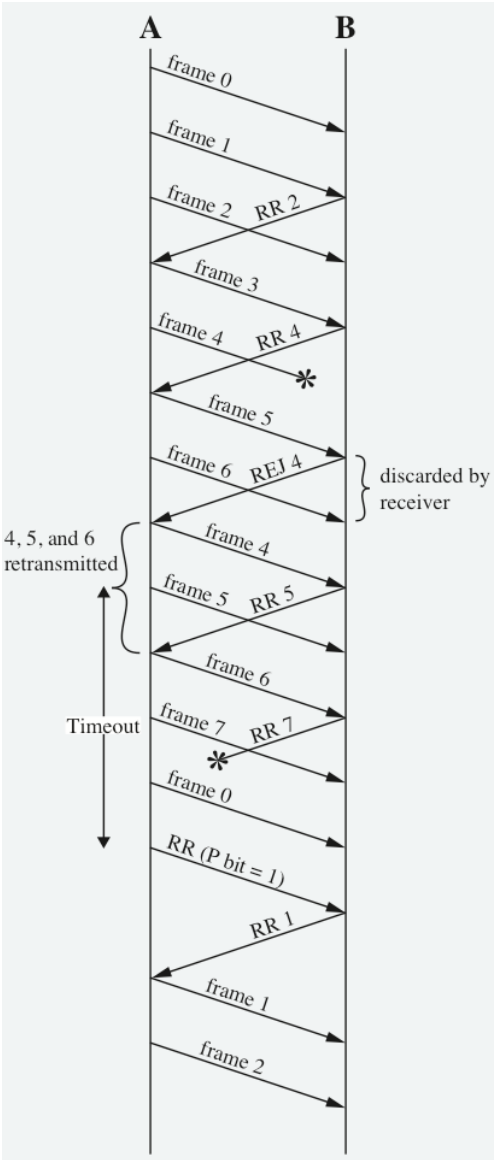
$$\text{E}[U] = \frac{t_{\text{frame}}(1 - P)}{t_{\text{frame}} + t_{\text{frame}} + 2t_{\text{prop}} + 2t_{\text{proc}} + t_{\text{ack}}}.$$

If we further assume that $t_{\text{ack}} \ll t_{\text{frame}}$ and $t_{\text{proc}} \approx 0$, we can express the average link utilization more simply as

$$\text{E}[U] = \frac{t_{\text{frame}}(1 - P)}{t_{\text{frame}} + 2t_{\text{prop}}} = \frac{1 - P}{1 + 2a}$$

where $a = \frac{t_{\text{prop}}}{t_{\text{frame}}}$. Note this agrees with our earlier flow control results with $P = 0$.

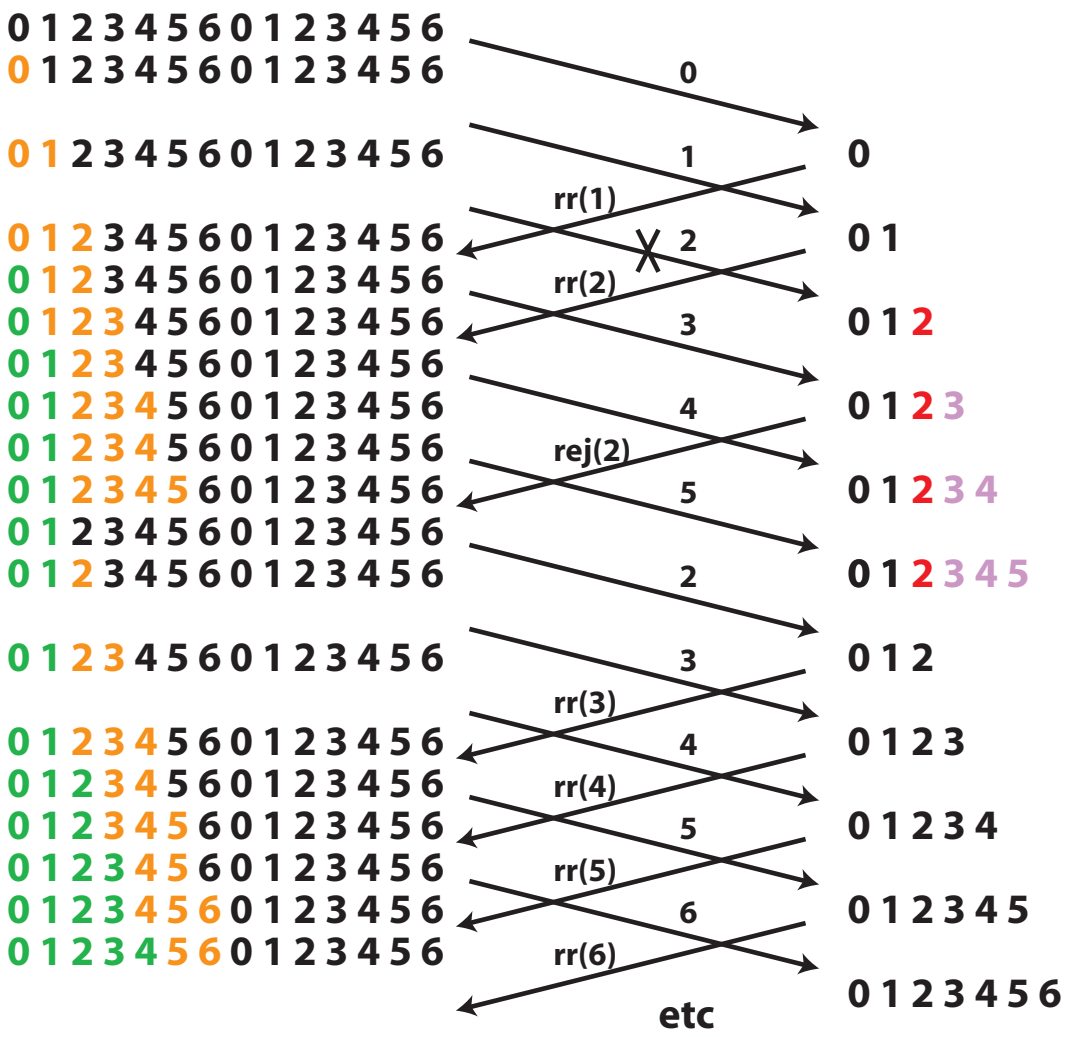
Go-Back-N ARQ



Basic idea:

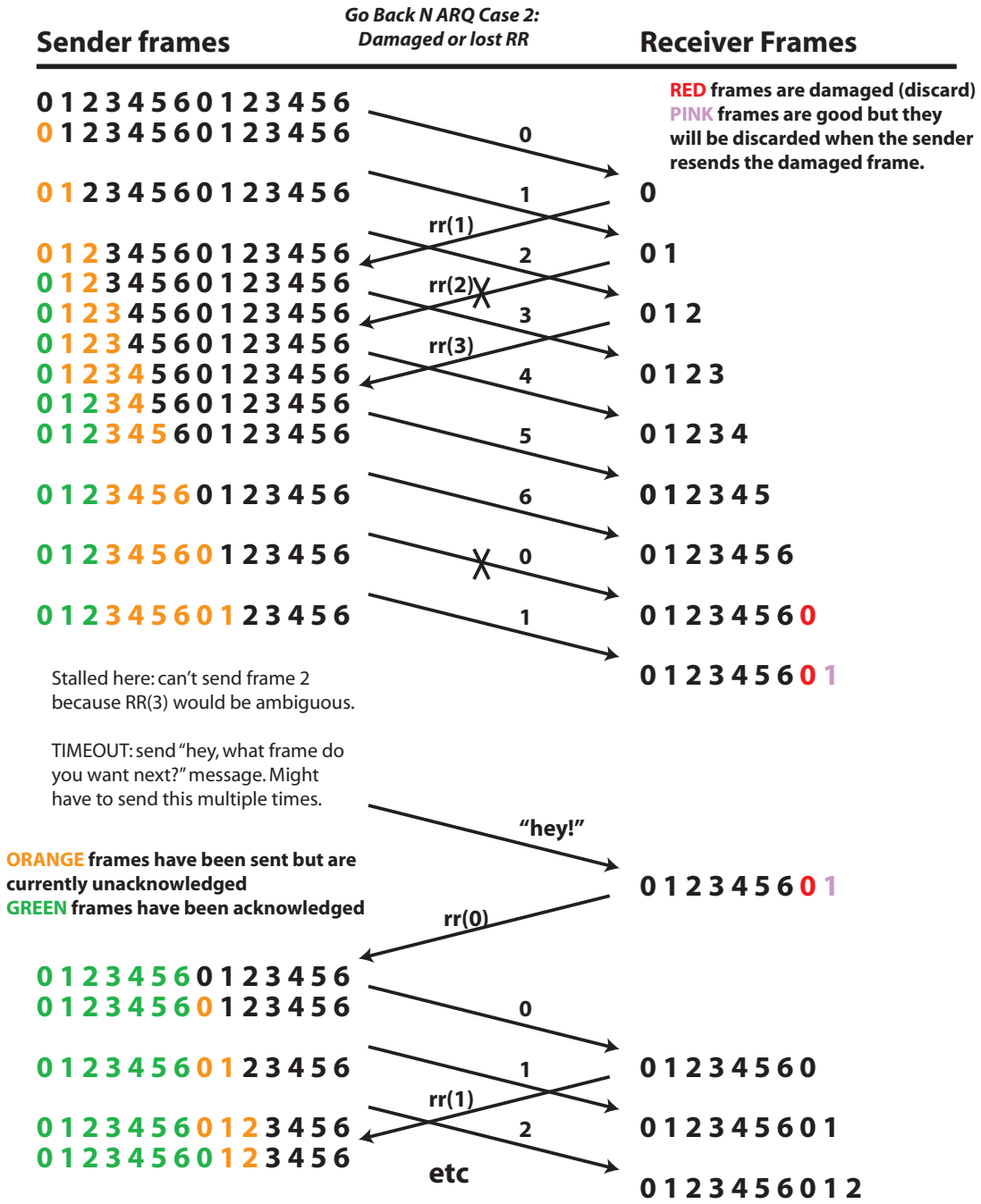
1. Based on sliding-window flow control.
2. Usual RR N acknowledgements for good frames.
3. REJ N message means "send all frames again, starting from frame N"
4. After sending REJ N message, receiver discards all frames $N, N+1, \ldots$ even if some were successfully received.
5. Timeouts still present to handle lost ACKs.
6. Most commonly used method for error control.

Go Back N ARQ Case 1: Damaged frame



ORANGE frames have been sent but are currently unacknowledged
GREEN frames have been acknowledged

RED frames are damaged (discard)
PINK frames are good but they will be discarded when the sender resends the damaged frame.


Go Back N ARQ Case 2:
Damaged or lost RR
Sender frames
Receiver Frames
RED frames are damaged (discard)
PINK frames are good but they will be discarded when the sender resends the damaged frame.
0 1 2 3 4 5 6 0 1 2 3 4 5 6
0 1 2 3 4 5 6 0 1 2 3 4 5 6
0
1
0 1 2 3 4 5 6 0 1 2 3 4 5 6
rr(1)
2
0 1
0 1 2 3 4 5 6 0 1 2 3 4 5 6
rr(2)
3
0 1 2
0 1 2 3 4 5 6 0 1 2 3 4 5 6
rr(3)
4
0 1 2 3
0 1 2 3 4 5 6 0 1 2 3 4 5 6
5
0 1 2 3 4
0 1 2 3 4 5 6 0 1 2 3 4 5 6
6
0 1 2 3 4 5
0 1 2 3 4 5 6 0 1 2 3 4 5 6
0
0 1 2 3 4 5 6
0 1 2 3 4 5 6 0 1 2 3 4 5 6
1
0 1 2 3 4 5 6 0
0 1 2 3 4 5 6 0 1 2 3 4 5 6
Stalled here: can't send frame 2 because RR(3) would be ambiguous.
TIMEOUT: send "hey, what frame do you want next?" message. Might have to send this multiple times.
"hey!"
0 1 2 3 4 5 6 0 1
ORANGE frames have been sent but are currently unacknowledged
GREEN frames have been acknowledged
rr(0)
0 1 2 3 4 5 6 0 1 2 3 4 5 6
0 1 2 3 4 5 6 0 1 2 3 4 5 6
0
0 1 2 3 4 5 6 0
1
0 1 2 3 4 5 6 0 1 2 3 4 5 6
rr(1)
0 1 2 3 4 5 6 0 1
2
0 1 2 3 4 5 6 0 1 2 3 4 5 6
0 1 2 3 4 5 6 0 1 2 3 4 5 6
0 1 2 3 4 5 6 0 1 2
etc

Go-Back-N ARQ: Transmitter Timeout

If, for any reason, the transmitter does not receive a RR indicating frame i was successfully received (recall that RRs are cumulative), it will timeout. There are two options:

- Option 1:
  - Transmitter just sends frame i again
- Option 2:
  - Transmitter sends special "hey" message to receiver asking what frame the receiver wants next
  - This message may need to be repeated until the receiver responds
  - The receiver responds with RR i
  - Transmitter resumes transmission starting from frame i

Go-Back-N ARQ: Damaged/Lost Frames/RRs/REJs

If frame i is received correctly, the receiver typically transmits RR $i+1$ (RRs are cumulative and some can be skipped).

If frame i is damaged (or lost), the receiver simply discards that frame and does nothing.

- If frame $i+1$ (or any frame with a sequence number greater than i) is subsequently received, the receiver sends a REJ i message.
- If no more frames are received, the transmitter's frame i timeout occurs

A damaged or lost RR $i+1$ can result in:

- no effect if a subsequent RR is received before the sliding window stalls and the transmitter's frame i timeout occurs
- sliding window stall but no timeout if a subsequent RR is received
- transmitter's frame i timeout occurs

A damaged or lost REJ results in timeout.

Go-Back-N ARQ Timing Analysis Example

Suppose we have a scenario with

$$t_{\text{frame}} = 1 \text{ ms}$$
$$t_{\text{prop}} = 2 \text{ ms}$$
$$t_{\text{timeout}} = 10 \text{ ms}$$
$$W = 5$$

and negligible ACK and processing times. Also assume the receiver sends RRs for all correctly received frames. Suppose we transmit 10 frames and the 4th frame is lost. How much time does it take to transmit all of the frames and receive all of the ACKs?

Note that $W \geq 2a + 1$ so we have 100% link utilization. So if there were no errors, the total time would be

$$T = 10 \cdot t_{\text{frame}} + 2t_{\text{prop}} = 14 \text{ ms}$$

Go-Back-N ARQ Timing Analysis Example (continued)

What happens when the 4th frame is lost?

- When the receiver receives the 4th frame, it discards it (no response)
- When the receiver receives the 5th frame, it sends a REJ
- When the transmitter receives the REJ, it starts transmission again from the 4th frame

The receiver receives the 5th frame at time

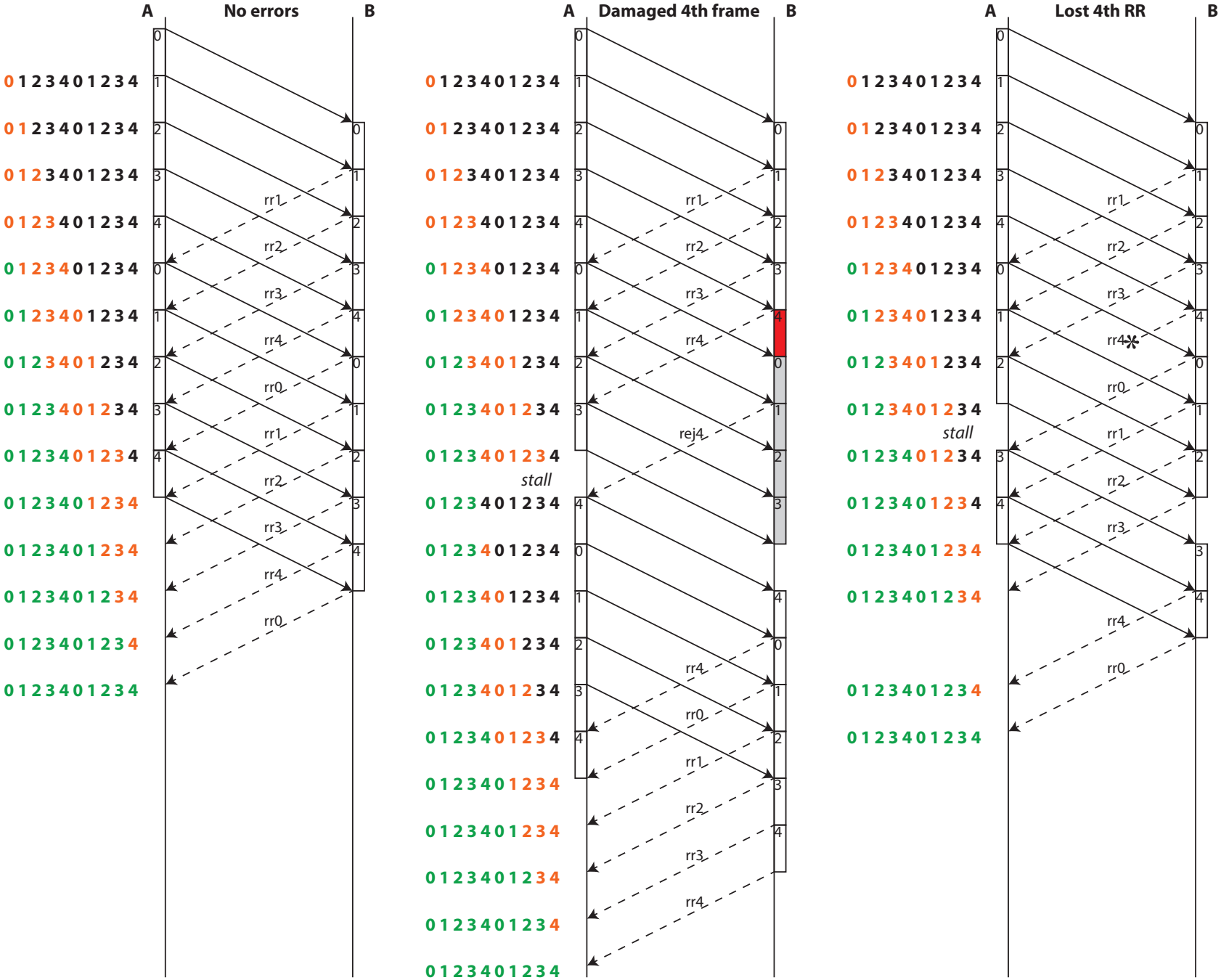
$$t_5 = 5t_{\text{frame}} + t_{\text{prop}} = 7 \text{ ms}$$

The transmitter receives the REJ at time

$$t_{\text{rej}} = t_5 + t_{\text{prop}} = 9 \text{ ms}$$

The receiver then transmits the 4th through 10th frames (7 frames total) without error. Hence, the total time is

$$T = t_{\text{rej}} + 7t_{\text{frame}} + 2t_{\text{prop}} = 20 \text{ ms}$$


A
No errors
B
rr1
rr2
rr3
rr4
rr0
A
Damaged 4th frame
B
rr1
rr2
rr3
rr4
rej4
stall
rr4
rr0
rr1
rr2
rr3
rr4
A
Lost 4th RR
B
rr1
rr2
rr3
rr4*
rr0
stall
rr1
rr2
rr3
rr4
rr0

Go-Back-N ARQ: Average Link Utilization

To determine the average link utilization, we assume that the only types of errors are damaged frames and that the window is long enough so that stalling doesn't occur. In this case, if we denote ℓ as the number of times frame i is received in error, we can express the total number of frames transmitted to successfully receive frame i as

$$f(\ell) = 1 + (\ell - 1)K$$

where $K \approx \min\{W, 2a + 1\}$ is the number of frames that must be transmitted again if frame i is damaged.

The average number of frames that must be transmitted for each frame is then

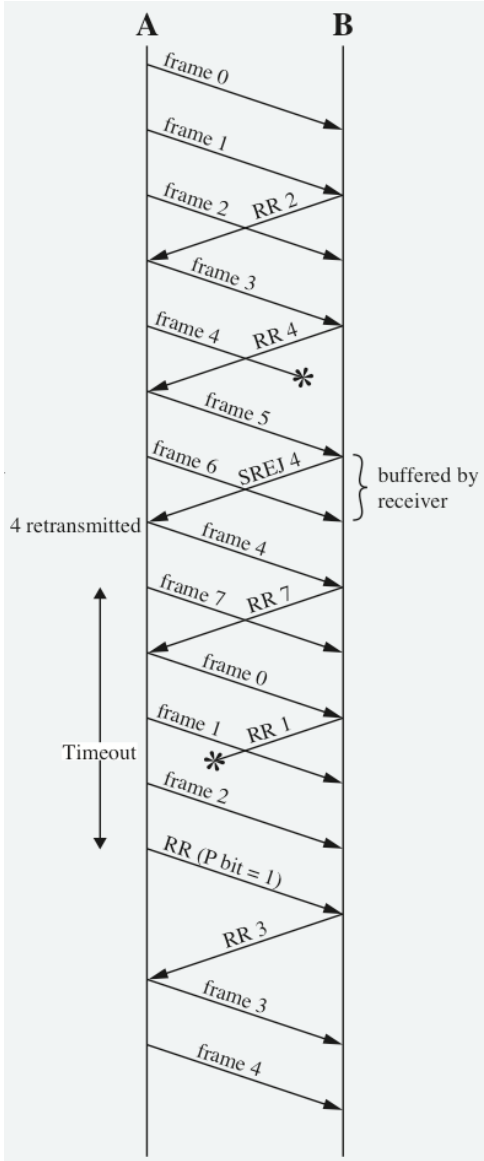
$$\mathrm{E}[\text{frames transmitted per successful frame}] = \sum_{\ell=1}^{\infty} f(\ell) P^{\ell-1}(1-P) = \frac{1 - P + KP}{1 - P}$$

Further assuming $t_{\text{ack}} \ll t_{\text{frame}}$ and $t_{\text{proc}} \approx 0$, the average link utilization is then

$$\mathrm{E}[U] = \begin{cases} \frac{1-P}{1+2aP} & W \geq 2a + 1 \\ \frac{W(1-P)}{(2a+1)(1-P+WP)} & W < 2a + 1 \end{cases}$$

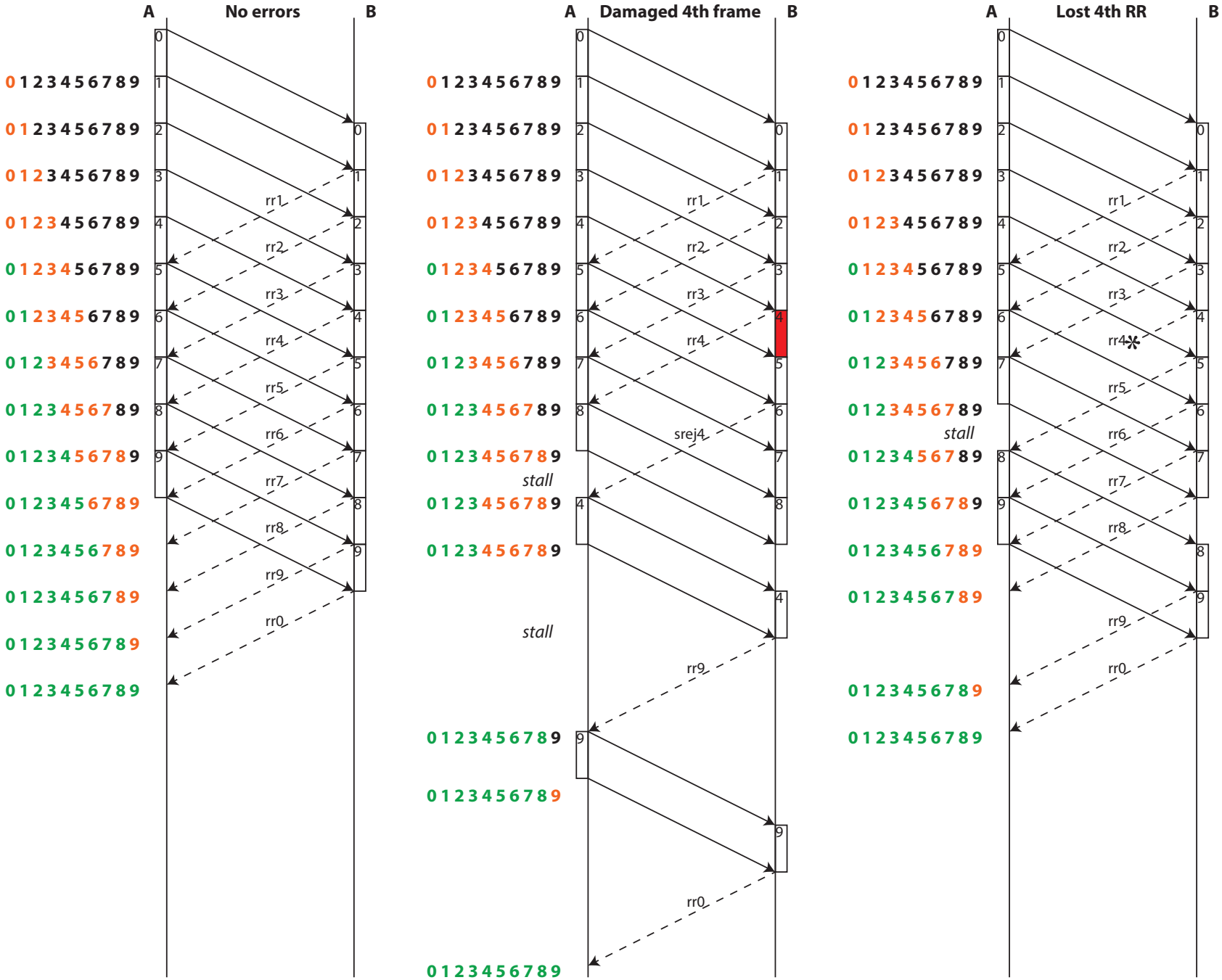
where $a = \frac{t_{\text{prop}}}{t_{\text{frame}}}$. Note this agrees with our earlier flow control results with $P = 0$.

Selective-Reject ARQ



Basic idea:

1. Similar to go-back-N except the only frames that are retransmitted are those that receive a negative acknowledgement SREJ or timeout.
2. Usual RR N acknowledgements for good frames.
3. SREJ N message means "send just frame N again"
4. Timeouts still present to handle lost ACKs.
5. Can be more efficient than go-back-N but also more complicated.
6. Window size should be no larger than half the range of the sequence numbers to avoid ambiguity.


A
No errors
B
rr1
rr2
rr3
rr4
rr5
rr6
rr7
rr8
rr9
rr0
A
Damaged 4th frame
B
rr1
rr2
rr3
rr4
srej4
stall
stall
rr9
rr0
A
Lost 4th RR
B
rr1
rr2
rr3
rr4*
stall
rr5
rr6
rr7
rr8
rr9
rr0

Selective Reject ARQ: Average Link Utilization

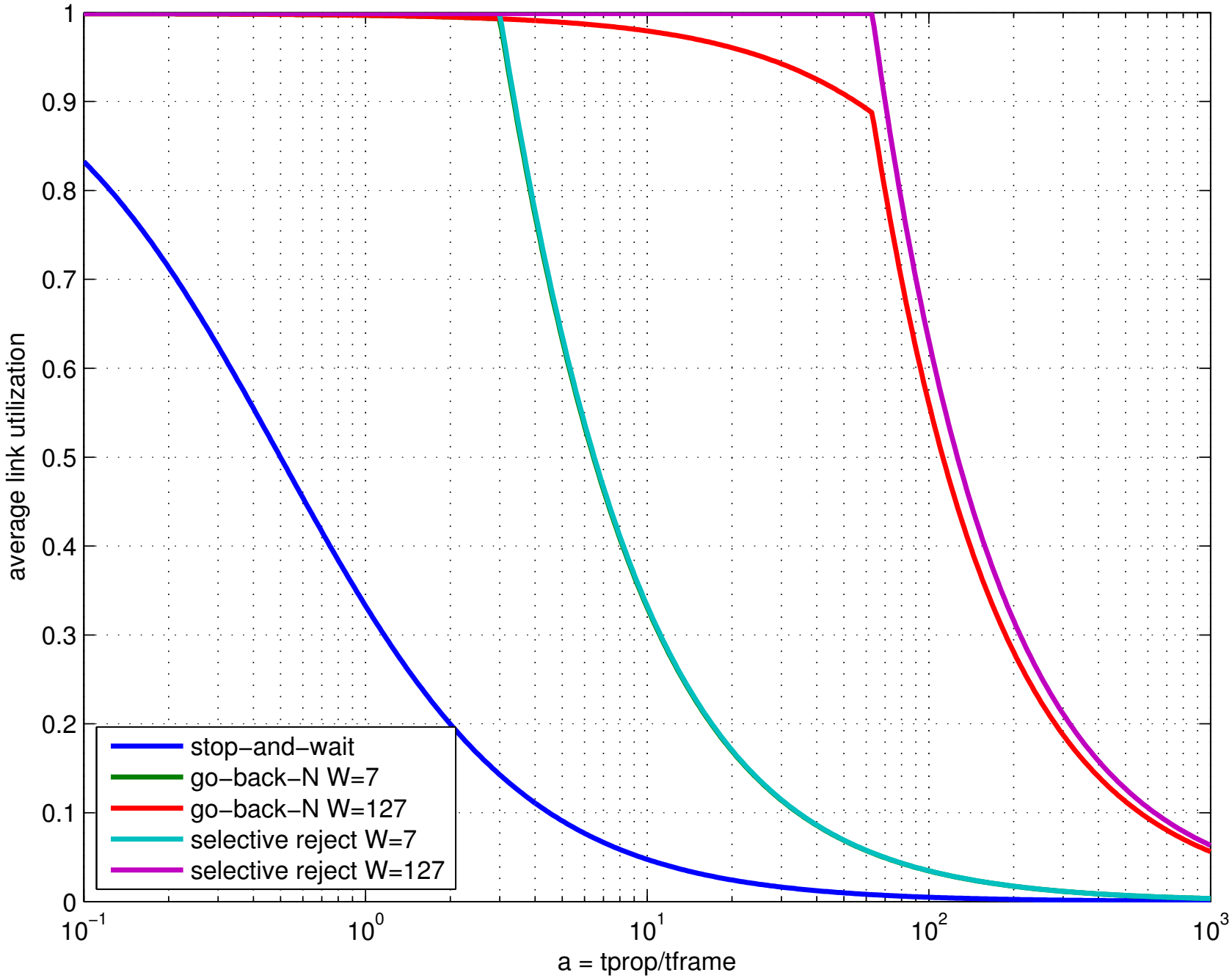
To determine the average link utilization, we assume that the only types of errors are damaged frames and that the window is long enough so that stalling doesn't occur. In this case, the analysis is similar to the stop-and-wait case.
The **average** number of transmissions required to successfully send a frame is

$$\mathrm{E}[\text{transmissions}] = \sum_{k=1}^{\infty} kP^{k-1}(1-P) = (1-P)\sum_{k=1}^{\infty} kP^{k-1} = \frac{1}{1-P}$$

Further assuming $t_{\text{ack}} \ll t_{\text{frame}}$ and $t_{\text{proc}} \approx 0$, the average link utilization is then

$$\mathrm{E}[U] = \begin{cases} 1-P & W \geq 2a+1 \\ \frac{W(1-P)}{2a+1} & W < 2a+1 \end{cases}$$

where $a = \frac{t_{\text{prop}}}{t_{\text{frame}}}$. Note this agrees with our earlier flow control results with $P = 0$.


average link utilization
a = tprop/tframe
stop–and–wait
go–back–N W=7
go–back–N W=127
selective reject W=7
selective reject W=127
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
10^{-1}
10^{0}
10^{1}
10^{2}
10^{3}

Final Remarks

- Primary purpose of error control: make channel reliable
- How? Receiver transmits ACK/RR/REJ/SREJs to sender
- Stop-and-wait ARQ
  - Simple but low link utilization, especially if propagation delays are large
- Go-back-N ARQ (based on sliding-window)
  - REJ i message causes transmitter to retransmit all frames $i, i+1, \ldots$
  - Better link utilization
  - More complicated to analyze
  - Most common ARQ
- Selective-Reject ARQ (based on sliding-window)
  - SREJ i message causes transmitter to retransmit just frame i and then resume normal sequence
  - Best link utilization
  - More complicated to implement
- Timing analysis can be tricky. Use timing diagrams when possible.