Capitolo 5 (Chapter 3) Trasporto – Prima parte

Domande di ripasso

R1. Supponete che il livello di rete fornisca il seguente servizio: Il livello di rete nell'host sorgente accetta messaggi di dimensione massima pari a 1200 byte e un indirizzo di rete dell'host destinazione. Il livello di rete in questo modo garantisce la consegna del messaggio al livello di transport dell'host destinazione. Supponete infine che numerosi processi applicativi possano essere in esecuzione sia sull'host sorgente che sull'host destinazione.

a. Progettate il protocollo di trasporto più semplice possibile che consegni i dato applicativi al processo desiderato dell'host destinazione. Assumete che il sistema operativo assegni un numero di porta di 4 byte a ciascun processo applicativo in esecuzione.
b. Modificate il protocollo in modo che fornisca un "numero di porta di ritorno" al processo destinazione.
c. Nei vostri protocolli, il livello di trasporto deve esistere anche sui router e deve "fare qualcosa"?

R2. Considerate un pianeta in cui ognuno appartiene ad una famiglia di sei persone, ogni famiglia vive in una casa di sua proprietà, ogni casa ha un indirizzo univoco, e ogni persona nella casa ha un nome univoco all'interno della casa. Supponete che questo pianeta abbia un servizio di posta che consegna lettere da una casa sorgente ad una casa destinazione. Questo servizio di posta richiede che (1) la lettera sia posta in una busta, e (2) l'indirizzo della casa destinazione e nient'altro sia chiaramente scritto sulla busta. Supponete che ogni famiglia abbia un componente delegato che raccoglie le lettere da spedire e distribuisce le lettere ricevute ai componenti della famiglia. Nel testo della lettera il mittente non necessariamente pone informazioni che permettono di risalire al destinatario della lettera.

(a) Descrivete un protocollo che i delegati possono utilizzare per consegnare le lettere ad altri delegati, in modo che i delegato ricevente possa consegnare la lettera al destinatario corretto nella casa.
(b) Nel vostro protocollo il servizio postale avrà mai bisogno di aprire le buste ed esaminare il contenuto per poter portare a termine il proprio lavoro?

R3. Consideriamo una connessione TCP tra gli Host A e B. Supponiamo che i segmenti TCP in viaggio tra A e B abbiano numero di porta di origine x e numero di porta di destinazione y. Quali sono i numeri di porta di origine e destinazione per i segmenti che viaggiano dall'Host B ad A?

R4. Descrivete perché lo sviluppatore di una applicazione potrebbe scegliere di fare uso di UDP anziché di TCP.

R5. Perché nell'odierna Internet il traffico voce e video viene spesso invialo su TCP piuttosto che su UDP? Suggerimento: la risposta che stiamo cercando non ha nulla a che vedere con il meccanismo di controllo di congestione di TCP.

R6. È possibile che un' applicazione che fa uso di UDP ottenga un trasferimento dati affidabile? Come?

R7. Supponete che un processo in un host C abbia una socket UDP con un numero di porta 6789. Supponete che entrambi gli host A e B mandino ciascuno un datagramma UDP all'host C con numero di porta di destinazione 6789. Entrambi questi datagrammi saranno diretti alla stessa socket dell'host C? Se sì, come il processo sull'host C saprà che questi due datagrammi hanno origine da due host diversi?

R8. Supponete che un web server sia in esecuzione sull'host C sulla porta 80. Supponete che questo web server usi le connessioni persistenti e stia al momento ricevendo richieste da due diversi host, A e B. Tutte le richieste vengono inviate attraverso la stessa socket sull'host C? Se vengono fatte passare attraverso socket diverse, entrambe hanno la porta 80? Discutete e spiegate questa situazione.

R9. Nei nostri protocolli rdt, perché abbiamo bisogno di introdurre i numeri di sequenza?

R1O. Nei nostri protocolli rdt, perché abbiamo bisogno di introdurre i timer?

R11. Supponete che il ritardo end-to-end tra il mittente e il destinatario sia costante e noto al mittente. Sarebbe ancora necessario un timer nel protocollo rdt 3.0, assumendo che i pacchetti possano andare persi? Motivate la risposta.

R12. Provate la applet relativa al Go-back-N sul sito web del testo.
(a) Fate mandare alla sorgente 5 pacchetti e poi fermate l'animazione prima che qualcuno dei 5 pacchetti raggiunga le destinazione. Poi buttate via il primo pacchetto e fate ripartire l'animazione. Descrivete che cosa accade.
(b) Ripetete l'esperimento, ma ora lasciate che il primo pacchetto raggiunga la destinazione e buttate via il primo acknowledgement. Descrivete di nuovo che cosa accade.
(c) Infine provate a mandare 6 pacchetti. Che cosa accade?

R13. Ripetete R12, ma con l'applet relativa a Selective Repeat. In che modo Selective Repeat e Go-back-N sono diversi?

Pl. Supponiamo che il Client A dia inizio a una sessione Telnet con il Server S e che, quasi nello stesso istante, anche il Client B avvii una sessione Telnet con S. Fornite possibili numeri di porta di origine e destinazione per:
(a) i segmenti trasmessi da A a S
(b) i segmenti trasmessi da B a S
(c) i segmenti trasmessi da S ad A
(d) i segmenti trasmessi da S a B.
(e) Se A e B sono host diversi, è possibile che il numero di porta di origine dei segmenti da A a S coincida con quello dei segmenti da B a S?
(f) Che cosa avviene se si tratta dello stesso host?

P2. Relativamente alla Figura 3.5, quali sono i valori delle porte origine e destinazione nei segmenti che fluiscono dal server ai processi dei client? Quali sono gli indirizzi IP nei datagrammi a livello di rete che trasferiscono i segmenti a livello di trasporto?

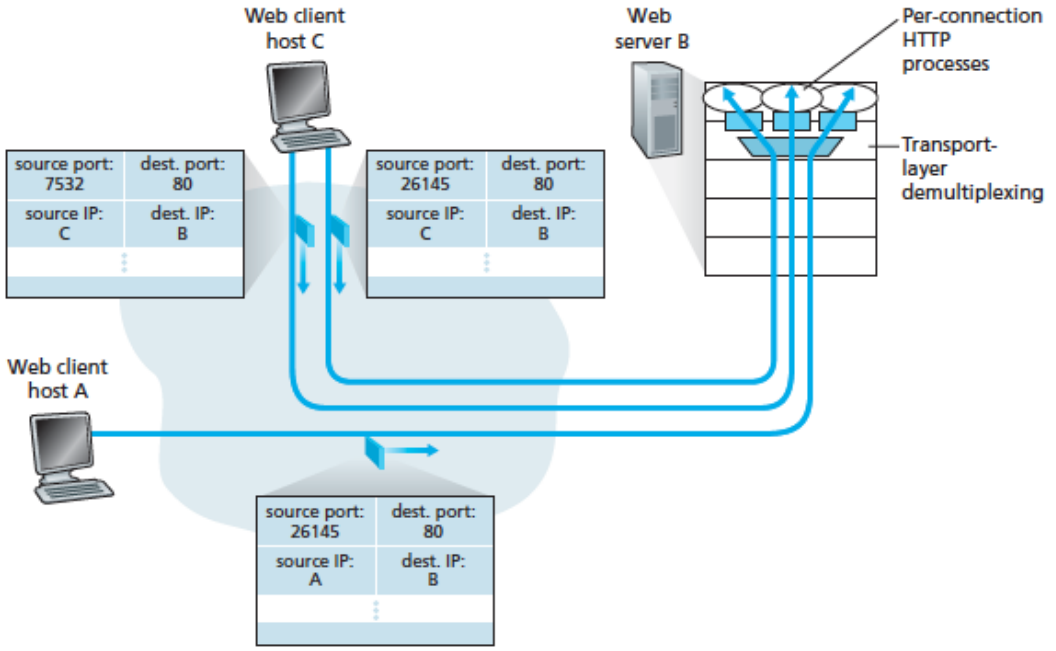

Figure 3.5 ♦ Two clients, using the same destination port number (80) to communicate with the same Web server application

P3. UDP e TCP utilizzano il complemento a l per calcolare il checksum. Supponiamo di avere i seguenti tre byte: 01010011, 01100110 e 01110100. Qual è il complemento a l della loro somma? Notiamo che, sebbene UDP e TCP usino checksum da 16 bit, in questo problema consideriamo addendi a 8 bit. Mostrate tutto il procedimento. Perché UDP considera il complemento a l della somma, e non semplicemente la somma? Con lo schema del complemento a l , come vengono rilevati gli errori dal destinatario? È possibile che un errore su l bit non venga rilevato? E che cosa avviene per gli errori su 2 bit?

P4. (a) Supponete di avere i seguenti 2 byte: 01011100 e 01100101. Qual è il complemento a l della somma di questi 2 byte?
(b) Supponete di avere i seguenti 2 byte: 11011010 e 01100101. Qual è il complemento a 1 della somma di questi 2 byte?
(c) Per i byte nella domanda (a), fornite un esempio dove un bit è cambiato in ciascuno dei 2 byte e il complemento a 1 non cambia.

P5. Supponete che un ricevente UDP calcoli il checksum Internet per il segmento UDP ricevuto e trovi che corrisponda al valore trasportato nel campo checksum. Può il ricevente essere assolutamente certo che non vi siano stati errori sui bit? Spiegate perché.

P6. Considerate le nostre motivazioni per la correzione del protocollo rdt2. 1. Mostrate che quando il destinatario, presentato nella Figura 3.57, opera con il mittente presentato nella Figura 3.11, può portare se stesso e il mittente in uno

stato di stallo, in cui entrambi sono in attesa di un· evento che non si verificherà mai.

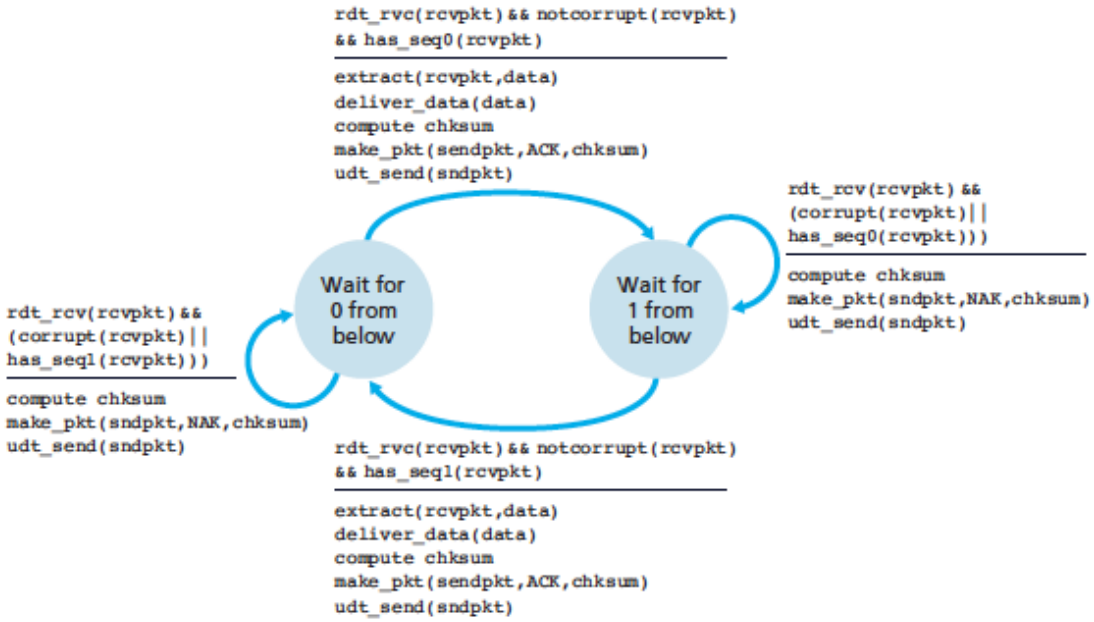

Figure 3.57 ♦ An incorrect receiver for protocol `rdt 2.1`

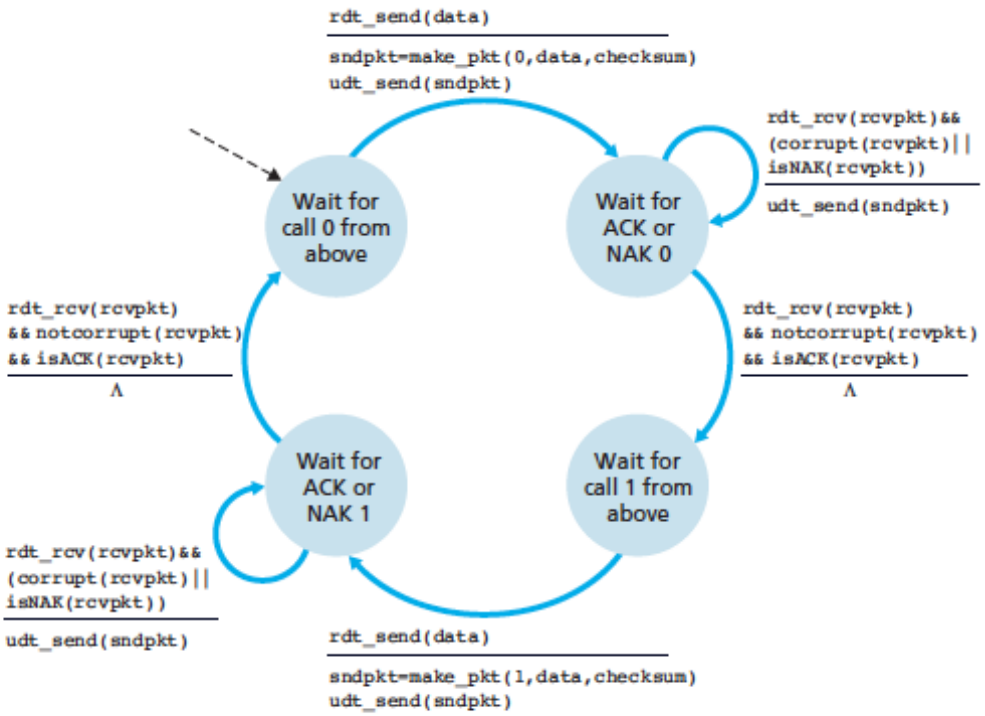

Figure 3.11 ♦ `rdt2.1` sender

P7. Nel protocollo rdt3. 0, i pacchetti ACK che fluiscono tra destinatario e mittente non hanno numeri di sequenza (sebbene abbiano un campo ACK contenente il numero di sequenza del pacchetto che stanno riscontrando). Perché i pacchetti ACK non richiedono numeri di sequenza?

P8. Disegnate l'automa a stati finiti del lato ricevente del protocollo rdt3. 0.

P9. Presentate una traccia del funzionamento del protocollo rdt3. 0, quando vengono alterati pacchetti di dati o di acknowledgement. La descrizione dovrebbe essere simile a quella della Figura 3.16.

P10. Considerate un canale che può perdere pacchetti, ma che presenta un ritardo massimo noto. Modificate il protocollo rdt2. 1 per includere timeout e ritrasmissioni del mittente. Argomentate in modo informale la correttezza del vostro protocollo (su tale canale).

P11. Considerate il ricevente rdt2. 2 della Figura 3.14 e la creazione di un nuovo pacchetto nell'auto-transizione (la transizione da uno stato a sé stesso) negli stati "attesa di chiamata O dal basso" e "attesa di chiamata 1 dal basso": `sndpkt=make_pkt(ACK, 0, checksum)` e `sndpkt=make_pkt(ACK, 0, checksum)`. Il protocollo funzionerebbe correttamente se questa azione venisse rimossa dall'auto-transizione "attesa di chiamata 1 dal basso"? Giustificate la risposta. E se rimossa dall' auto-transizione "attesa di chiamata O dal basso"? Suggerimento: In quest'ultimo caso considerate che cosa accadrebbe se il primo pacchetto da mittente a ricevente fosse corrotto.

P12. Il lato mittente di rdt3. 0 ignora, cioè non esegue alcuna azione, sia i pacchetti con errori sia i pacchetti di acknowledgement con un valore errato nel campo acknum. Supponiamo che, in queste circostanze, rdt3. 0 abbia semplicemente ritrasmesso il pacchetto di dati corrente. Il protocollo funzionerebbe ancora? Suggerimento: ricordate che, se ci sono soltanto errori da un bit, non si verificano perdite di pacchetti, ma possono esserci timeout prematuri. Considerate quante volte viene inviato l'n-esimo pacchetto, quando n tende all'infinito.

P13. Considerate il protocollo rdt 3.0. Tracciate un grafico che mostri che, se la connessione di rete tra mittente e destinatario può riordinare i messaggi (ossia, i due messaggi che si propagano nel mezzo tra mittente e destinatario possono essere riordinati), allora il protocollo con alternanza di 1 bit non funziona correttamente. Accertatevi di aver identificato con chiarezza in quale direzione il protocollo non funziona. Il vostro diagramma dovrebbe presentare il mittente alla sinistra e il destinatario alla destra, con l'asse temporale che scende lungo la pagina, mostrando lo scambio di messaggi di dati (D) e acknowledgement (A). Accertatevi di indicare il numero di sequenza dei segmenti di dati e acknowledgement.

P14. Considerate un protocollo di trasferimento dati affidabile che usa soltanto acknowledgement negativi. Ipotizzate che il mittente trasmetta dati poco frequentemente. In questo caso sarebbe preferibile un protocollo dotato soltanto di NAK, rispetto a un protocollo che utilizza ACK? Perché? Supponete ora che il mittente abbia molti dati da inviare e che la connessione end-to-end presenti poche perdite. In questo secondo caso sarebbe preferibile un protocollo dotato soltanto di NAK o un protocollo che utilizza ACK? Perché?

PI5. Considerate l'esempio della Figura 3.17. Quanto dovrebbe essere grande l'ampiezza della finestra affinché l'utilizzo del canale superi il 98%? Supponete che il pacchetto sia di 1500 byte, comprensivi di intestazione e dati.

P16. Supponete che un'applicazione usi come protocollo di livello di trasporto rdt 3.0. Poiché il protocollo stop-and-wait presenta un basso utilizzo del canale (mostrato nell'esempio in cui si passa da un lato all'altro del continente), i progettisti dell'applicazione lasciano che il ricevente continui a inviare indietro più di due acknowledgement alternati ACK O e ACK l anche se i dati corrispondenti non sono arrivati al ricevente. Questa progettazione dell'applicazione incrementa l'utilizzo del canale? Perché? Ci sono problemi potenziali in questo approccio? Spiegatelo.

P17. Considerate due entità di rete, A e B, connesse da un canale perfettamente bidirezionale (cioè ogni messaggio inviato verrà ricevuto correttamente; il canale non danneggerà, perderà o riordinerà i pacchetti). A e B devono consegnarsi dei messaggi vicendevolmente e in modo alternato: prima, A deve consegnare un messaggio a B, quindi B deve consegnare un messaggio ad A, poi A deve consegnare un messaggio a B e via cosi. Se un' entità si trova in uno stato in cui non debba tentare di consegnare un messaggio all'altra, e avviene un evento come una chiamata `rdt_send (data)` da sopra che tenta di passargli dati per trasmetterli, tale chiamata può essere semplicemente ignorata attraverso una chiamata a `rdt_unable_ to_send (data)`, che informa il livello più alto di non essere attualmente in grado di inviare dati. Nota: questa assunzione semplificata è fatta in modo che non dobbiate preoccuparvi di gestire un buffer per i dati. Disegnate una FSM per dare una specifica di questo protocollo (una FSM per A e una per B). Notate che qui non dovete preoccuparvi di un meccanismo affidabile; il punto principale di questo problema è creare una FSM che rifletta il comportamento sincronizzato delle due entità. Dovreste usare i seguenti eventi e azioni che hanno lo stesso significato che avevano nel protocollo rdt l . 0 nella Figura 3.9:
`rdt_send(data), packet=make_pkt(data),udt_send(packet), rdt_rcv(packet), extract(packet, data),deliver_data(data).`
Assicuratevi che il vostro protocollo rifletta l'alternanza stretta delle trasmissioni tra A e B e che indichiate gli stati iniziali di A e B nelle descrizioni tramite FSM.

P18. Nel generico protocollo SR studiato nel Paragrafo 3.4.4 il mittente trasmette un messaggio non appena questo è disponibile (se si trova nella finestra) senza aspettare un acknowledgement. Supponiamo: (1) di volere che il protocollo SR invii due messaggi alla volta (in altre parole, il mittente trasferirà una coppia di messaggi e invierà i due successivi soltanto quando è certo che la prima coppia sia stata ricevuta correttamente); (2) che il canale possa perdere messaggi senza però alterarli o cambiame l'ordine. Progettate un protocollo uni direzionale di controllo dell'errore per il trasferimento affidabile dei messaggi. Disegnate le FSM di mittente e destinatario. Descrivete il formato dei pacchetti scambiati tra mittente e destinatario e viceversa. Se le vostre chiamate di procedura sono diverse da quelle del Paragrafo 3.4 (per esempio: `udt_send (),`

`start_timer()`, `rdt_rcv()` e così via), dichiarate le loro azioni. Fornite un esempio (una linea temporale di mittente e destinatario) che mostri come il vostro protocollo reagisce alla perdita di pacchetti.

P19. Considerate uno scenario in cui l' Host A voglia simultaneamente inviare messaggi agli Host B e C. A è connesso a B e C tramite un canale broadcast: i pacchetti inviati da A sono trasportati sia a B sia a C. Supponete che il canale che collega i tre host possa perdere e alterare i messaggi (per esempio, un messaggio inviato da A potrebbe essere correttamente ricevuto da B, ma non da C). Progettate un protocollo dì controllo d'errore di tipo stop-and-wait per trasferire in modo affidabile i pacchetti da A a B e C, di modo che A non prenda nuovi dati dal livello superiore finché non sia certo che B e C abbiano correttamente ricevuto il pacchetto corrente. Fomite le FSM per A e C. Suggerimento: la FSM di B dovrebbe essere essenzialmente la stessa di C. Inoltre, fomite una descrizione del formato o dei formati di pacchetto utilizzati.

P20. Considerate uno scenario nel quale l'Host A e l'Host B vogliano inviare messaggi a un Host C. Gli Host A e C sono collegati da un canale che può perdere e alterare (ma non riordinare) i messaggi. Gli Host B e C sono collegati da un altro canale (indipendente da quello che collega A e C) con le stesse caratteristiche. Il livello di trasporto all'Host C si dovrebbe alternare nell'invio di messaggi da A e B al livello superiore (cioè, dovrebbe prima consegnare i dati dei pacchetti da A, poi quelli dei pacchetti da B e così via). Progettate un protocollo per il controllo degli errori tipo stop-and-wait, per un trasferimento affidabile dei pacchetti da A, B e C, con la consegna alternata di C, descritta sopra. Fornite la descrizione delle FSM di A e C e fomite anche una descrizione dei formati dei pacchetti usati. Suggerimento: l'automa di B dovrebbe essere essenzialmente uguale a quello di A.

P21. Supponete di avere due entità di rete, A e B. I messaggi di dati di B verranno inviati ad A secondo le seguenti convenzioni.
(a) Quando A riceve una richiesta dal livello superiore per ottenere il prossimo messaggio di dati (D) da B, deve inviare un messaggio di richiesta (R) a B sul canale tra A e B.
(b) B può inviare un messaggio di dati (D) ad A sul canale tra B e A soltanto quando riceve un messaggio R.
(c) A dovrebbe consegnare esattamente una copia di ciascun messaggio D al livello superiore.
(d) Sul canale tra A e B i messaggi R possono essere perduti. ma non alterati.
(e) I messaggi D, una volta inviati, vengono sempre consegnati correttamente.
(f) Il ritardo lungo i canali non è noto ed è variabile.
Progettate (fornendo una descrizione con una FSM) un protocollo che incorpora i meccanismi appropriati per compensare le possibili perdite del canale tra A e B e che implementa il trasferimento di messaggi al livello superiore presso l'entità A, come discusso precedentemente. Utilizzate solo i meccanismi assolutamente necessari.

P22. Considerate il protocollo GBN con un' ampiezza della finestra mittente pari a 4 e intervallo di numeri di sequenza da O a 1024. Supponete che all'istante t il

successivo pacchetto nella sequenza atteso dal destinatario abbia numero di sequenza k. Ipotizzate che il mezzo non effettui riordino dei messaggi. Rispondete alle seguenti domande.
(a) Quali sono i possibili numeri di sequenza all'interno della finestra del mittente all' istante t? Giustificate la vostra risposta.
(b) Quali sono tutti i possibili valori del campo ACK nei possibili messaggi che si propagano all'indietro verso il mittente all'istante t? Giustificate la vostra risposta.

P23. Considerate i protocolli GBN e SR. Supponete che lo spazio dei numeri di sequenza abbia dimensione k. Qual è, per i due protocolli, la finestra mittente più grande che evita il verificarsi di problemi come quelli presentati nella Figura 3.27?

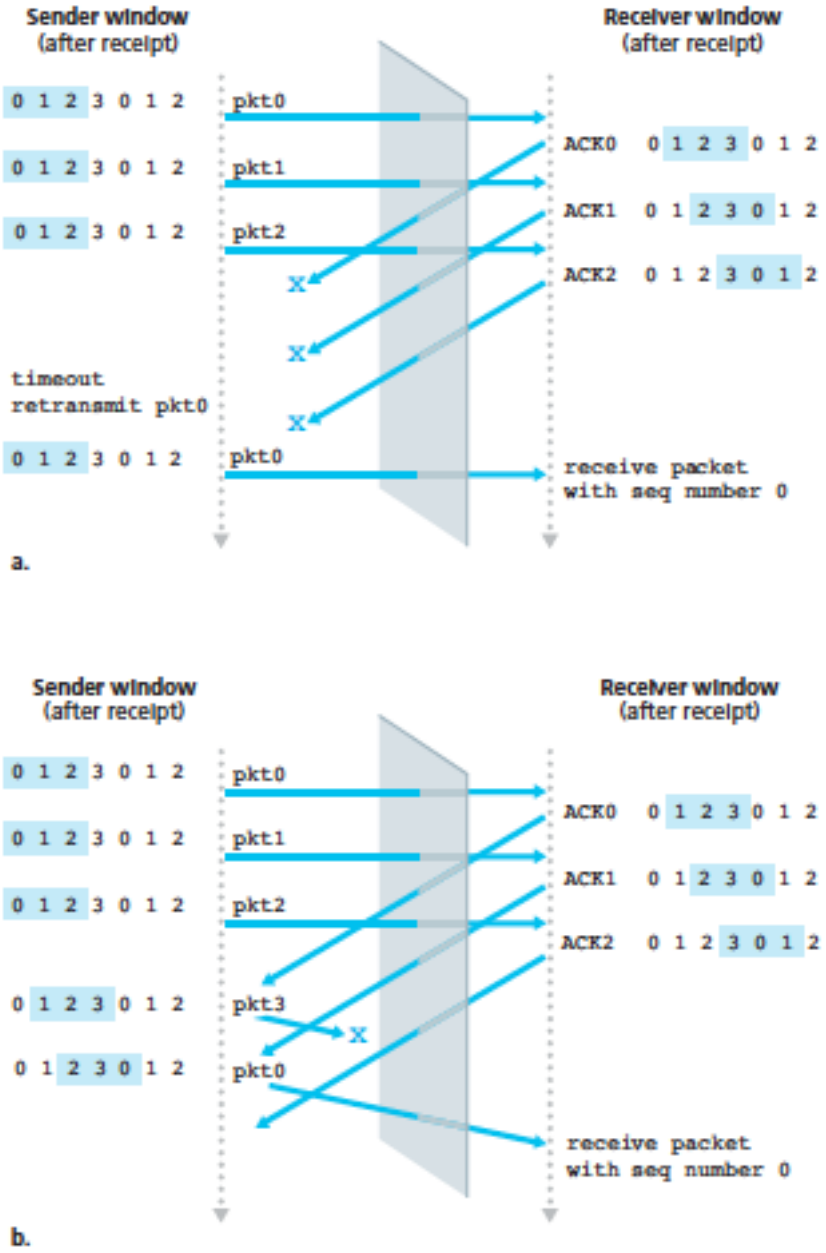

Figure 3.27 ♦ SR receiver dilemma with too-large windows: A new packet or a retransmission?

P24. Rispondete alle seguenti domande, giustificando brevemente le vostre risposte.
(a) Nel protocollo SR, il mittente può ricevere un ACK relativo a un pacchetto che ricade al di fuori della sua finestra corrente?
(h) Nel protocollo GBN, il mittente può ricevere un ACK relativo a un pacchetto che ricade al di fuori della sua finestra corrente?
(c) Il protocollo stop-and-wait è uguale al protocollo SR con ampiezza di finestra mittente e destinatario pari a l?
(d) Il protocollo stop-and-wait è uguale al protocollo GBN con ampiezza di finestra mittente e destinatario pari a 1?