RETI DI CALCOLATORI – 05.06.2013

Per ottenere una valutazione sufficiente dell'intera prova è necessario ottenere una valutazione sufficiente della prima parte.

Prima parte (10 punti)

Q1. Supponiamo che un router A trasmetta un pacchetto di 256 byte a un router B direttamente collegato ad A. Supponiamo che la velocità di propagazione sia di 2×10^8 m/s e che la velocità di trasmissione sia di 10 Mbps. Indicare –giustificando la risposta- quanto deve essere la distanza tra A e B affinché il primo bit del pacchetto arrivi a B quando l'ultimo bit parte da A.

Q2. Supponiamo che il TCP di un host A abbia stabilito una connessione con un suo pari e che, trovandosi nello stato di congestion avoidance, dopo avere ricevuto un riscontro R non duplicato aggiorni la dimensione della sua finestra di congestione a $\frac{10}{3}$ MSS. Indicare –giustificando la risposta– quale era la dimensione della finestra di congestione del TCP di A subito prima di ricevere R.

Q3. Un nodo RIP R, dopo aver calcolato che la sua distanza per la destinazione Z è n+2, riceve da un suo vicino V un advertisement A che pubblicizza una distanza n per Z. Indicare –giustificando la risposta– in quali casi R *non* aggiorna la sua distanza per Z dopo avere ricevuto A.

Q4. Un anello Chord utilizza identificatori a 7 bit ed è formato dai nodi con identificatori 3, 13, 33, 34, 58, 61, 62, 73, 81, 93, 100, 115, 120. Indicare quali sono i finger del nodo 73.

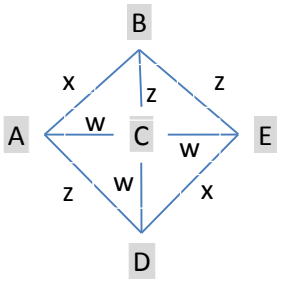
Seconda parte

E1 (5 punti). Una rete utilizza una variante del protocollo *Go-Back-N* per ridurre il traffico di rete utilizzando la tecnica del piggybacking. In particolare il receiver di un host non invia un riscontro positivo subito dopo aver ricevuto correttamente il segmento atteso, ma, dopo aver preparato il riscontro, segnala al sender dello stesso host la possibilità di utilizzare un riscontro per inviare dati in piggybacking. Per fare ciò, il receiver assegna il valore *true* alla variabile booleana *AckToSend* condivisa con il sender. Il receiver controlla quindi il valore di *AckToSend* dopo un tempo *TPB*. Se tale valore è ancora *true* allora ciò indica che il sender non ha inviato il riscontro col piggybacking e quindi il receiver provvede ad inviarlo, in caso contrario il receiver non invia niente. In entrambi i casi il receiver passa ad attendere il successivo segmento. Descrivere, utilizzando un automa a stati finiti, il comportamento del receiver del protocollo.

E2 (4 punti). Descrivere con uno pseudo-codice il comportamento di un server DNS che risponde solo a richieste di tipo "A" relative al server Web www.myserver.com, restituendo uno dei K indirizzi IP associati a tale server Web in modo da distribuire il carico, e ignora invece ogni altra richiesta che riceve. Supporre di avere a disposizione le operazioni:

getDNSrequest()	//per ricevere una richiesta DNS
setDNSAnswer(m,t)	//per inserire una tupla t nel campo *answer* di una richiesta DNS m
sendDNSAnswer(m)	//per inviare una risposta DNS m
getQTYPE(m)	//per ricavare il valore del campo QYTPE di una richiesta DNS m
getQNAME(m)	//per ricavare il valore del campo QNAME di una richiesta DNS m

E3 (5 punti). Si consideri la rete a lato, in cui i nodi utilizzano l'algoritmo distance vector con poisoned reverse, e in cui i valori di x, w, z sono interi positvi. (a) Determinare quali vincoli devono valere tra i valori di x, w e z affinché, quando la rete ha raggiunto lo stato di quiescenza, nessuno dei vettori che A ha ricevuto dai suoi vicini contenga distanze avvelenate. (b) Determinare se l'insieme di vincoli $\begin{cases} x = w + 2 \\ z = 2w + 1 \end{cases}$ soddisfa i vincoli determinati al punto precedente.



E4 (6 punti). Supponiamo che solo tre nodi (A, B, C) di una rete Ethernet debbano trasmettere ciascuno un frame di 64 byte e che al tempo t_0, dopo aver verificato che il canale è inattivo, inizino tutti e tre a trasmettere simultaneamente il proprio frame: A per la prima volta, e B e C per la seconda volta (essendo i loro frame già incorsi precedentemente in una collisione). Supponendo che la frequenza di trasmissione sia di 10 Mbps e che il ritardo di propagazione tra ogni coppia di nodi sia di 0,8 microsecondi, indicare – giustificando la risposta – se la probabilità che né B né C riescano a trasmettere con successo il loro frame entro t_0+85 microsecondi è maggiore di ¾.

Traccia della soluzione

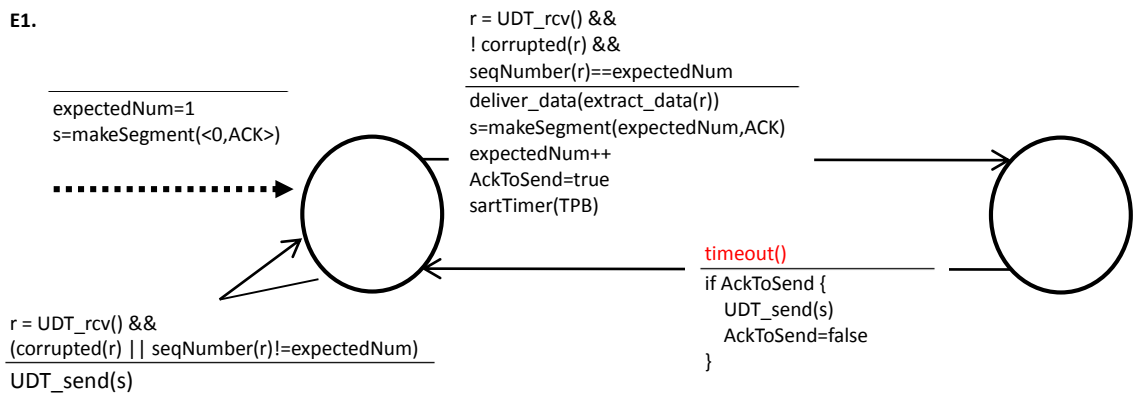
Q1. Se D è la distanza tra A e B, deve valere $\frac{1}{10^7} + \frac{D}{2 \cdot 10^8} = \frac{256 \cdot 8}{10^7}$ ovvero D=40940 metri.

Q2. A seguito della ricezione di R, il nuovo valore della finestra di congestione è determinato applicando la formula: $CongWin' = CongWin + \frac{MSS}{CongWin} \times MSS$. Se quindi *CongWin= n MSS* abbiamo: $n + \frac{1}{n} = \frac{10}{3}$ quindi $n = \frac{\frac{10}{3} \pm \sqrt{\left(\frac{10^2}{3^2}\right) - 4}}{2}$ ovvero $n = 3$ oppure $n = \frac{1}{3}$. La dimensione della finestra di congestione subito prima della ricezione di R era quindi 3 MSS (nell'ipotesi che il valore di *CongWin* non scenda mai sotto 1 MSS).

Q3. R aggiorna sempre a n+1 la sua distanza per Z, dato che $D_V(Z)+1=n+1<n+2$.

Q4. I finger del nodo 73 sono: 81, 81, 81, 81, 93, 115, 13

E1.



E2. Assumiamo che il server mantenga in un array B i K indirizzi IP associati a www.myserver.com.

```
I=0;
while true do {
    m=getDNSrequest();
    if (getQTYPE(m) =="A" && getQNAME(m) =="www.myserver.com")
        then { setDNSAnswer(m,<m.getQNAME(),B[i],"A">); sendDNSAnswer(m); i= i++ MOD K; }
}
```

E3. (a) I vettori inviati ad A conterranno il minimo dei valori indicati in ogni casella, dove i valori evidenziati in grassetto causerebbero l'avvelenamento della distanza. Quindi, affinché i vettori che A ha ricevuto dai suoi vicini non contengano distanze avvelenate, devono valere i vincoli: $\begin{cases} z < x + w \\ w < x \\ 2w < z \end{cases}$

	B	C	D	E
B	-	$min\begin{pmatrix} z, \\ \boldsymbol{x + w} \end{pmatrix}$	$min\begin{pmatrix} w + z, \\ \boldsymbol{x + z}, \\ \boldsymbol{x + 2w} \end{pmatrix}$	$min\begin{pmatrix} z, \\ \boldsymbol{x + 2w} \end{pmatrix}$
C	$min\begin{pmatrix} z, \\ \boldsymbol{x + w} \end{pmatrix}$	-	$min\begin{pmatrix} w, \\ x + 2z \end{pmatrix}$	$min\begin{pmatrix} w, \\ 2z \end{pmatrix}$
D	$min\begin{pmatrix} w + z, \\ x + 2w, \\ \boldsymbol{x + z} \end{pmatrix}$	$min\begin{pmatrix} w, \\ \boldsymbol{x + 2z} \end{pmatrix}$	-	$min\begin{pmatrix} x, \\ w + 2z, \\ 2w \end{pmatrix}$

(b) L'insieme di vincoli $\begin{cases} x = w + 2 \\ z = 2w + 1 \end{cases}$ soddisfa i vincoli determinati per (a) , infatti: $\begin{cases} 2w + 1 < w + 2 + w = 2w + 2 \\ w < w + 2 \\ 2w < 2w + 1 \end{cases}$

E4. Calcoliamo la probabilità p che B e/o C riescano a trasmettere un frame entro t_0+85μs. Notiamo che il tempo necessario per trasmettere un frame di 64 byte è 51,2μs e quindi solo uno tra B e C può riuscire a trasmettere il proprio frame. Notiamo inoltre che il numero massimo di collisioni che si possono verificare affinché B o C riesca a trasmettere con successo è n, dove n(0,8+4,8+0,8+9,6)+51,2<85, quindi n≤2.

B o C trasmette con successo subito dopo la collisione iniziale se A genera il numero casuale 1, mentre B (o C) genera 0 e C (o B) genera un numero >0, come illustrato dalla seguente tabella:

Dopo 1a collisione			
A	B	C	*p*
1	0	[1,3]	$\frac{6}{2 \times 4 \times 4}$
1	[1,3]	0	

B o C riesce a trasmettere con successo dopo la seconda collisione nei seguenti casi:

Dopo 1a collisione			Dopo 2a collisione			
A	B	C	A	B	C	*p*
1	0	0		0	[1,7]	$\frac{1}{2 \times 4 \times 4} \times \frac{14}{8 \times 8}$
				[1,7]	0	
0	0	0	[1,3]	0	[1,7]	$\frac{1}{2 \times 4 \times 4} \times \frac{42}{4 \times 8 \times 8}$
			[1,3]	[1,7]	0	
0	0	[1,3]	[1,3]	0		$\frac{3}{2 \times 4 \times 4} \times \frac{3}{4 \times 8}$
0	[1,3]	0	[1,3]		0	$\frac{3}{2 \times 4 \times 4} \times \frac{3}{4 \times 8}$

Quindi $p = \frac{6}{2^5} + \frac{1}{2^5} \times \frac{7}{2^5} + \frac{1}{2^5} \times \frac{21}{2^7} + \frac{1}{2^5} \times \frac{18}{2^5} = \frac{6}{2^5} + \frac{1}{2^5} \times \left(\frac{7}{2^5} + \frac{21}{2^7} + \frac{18}{2^5}\right) < \frac{7}{2^5}$ e quindi $(1 - p) > 1 - \frac{7}{2^5} = \frac{25}{32} > \frac{24}{32} = \frac{3}{4}$.