UNIVERSITÀ DEGLI STUDI
DI TRENTO
Dipartimento di Ingegneria
e Scienza dell'Informazione

Corso di Laurea in informatica

Reti di calcolatori

"Dry run" del 5 giugno 2014

Esercizio 1 (11 punti – domande e risposte brevi)

Si considerino i protocolli di tipo CSMA:

1. Come si comporta una stazione che deve trasmettere una trama e trova il canale libero?
2. Cosa indica la "persistenza" di un protocollo CSMA?
3. Nell'implementazione CSMA/CD di 802.3 (o Ethernet) in cosa consiste la funzione di Collision Detection?
4. Come viene implementata?

Sempre in relazione ai protocolli di livello 2

5. A cosa serve il "preambolo" prima di un delimitatore di trama nei protocollo data-link per canali broadcast?
6. Si faccia un esempio di tecnica per delimitare le trame

Esercizio 2 (11 punti)

Si consideri la seguente matrice delle adiacenze, che descrive in termini "da calcolatore" una rete. Una casella vuota indica che non c'è un link, altrimenti è il costo del link stesso

		Nodo sorgente				
		A	B	C	D	E
nodo destinazione	A		2	3		
	B	1			1	
	C	3				
	D			5		2
	E			1	2	

1. Disegnare la rete corrispondente.
2. Il grafo che descrive la rete ha link simmetrici o diretti?
3. Supponendo di usare il protocollo OSPF e che il costo dei link sia già stato distribuito tra tutti i nodi, si calcoli la tabella di instradamento (usando i metodi propri di OSPF) dei nodi A ed E
4. Supponendo che il nodo A debba inviare un messaggio di tipo link-state (flooding) per annunciare la modifica del costo del link A-B, quanti messaggi verranno inviati nella rete supponendo che a ciascun link corrisponde una diversa interfaccia di rete ai nodi.

Esercizio 3 (11 punti)

Si consideri il livello trasporto di Internet.

1. Riassumere e spiegare le funzioni fornite da TCP e UDP al livello applicativo.
2. Si spieghino gli algoritmi di slow-start e congestion avoidance di TCP.
3. In assenza di perdite e assumendo che la receiver window è RCW = 64kbytes e non varia durante le comunicazioni, la MTU è la minima ammissibile (576 bytes) e SSTHR = RCW/2 calcolare quanti RTT servono affinchè la congestion window CNW raggiunga la sua dimensione massima.
4. In una rete a 100Mbit/s (livello fisico), senza perdite e con RTT = 200ms dominato dal tempo di propagazione e quindi sostanzialmente costante, si calcoli il tempo impiegato a trasferire un file di 1 Gbyte con UDP oppure con TCP, la dimensione della MTU è quella calcolata al punto precedente.

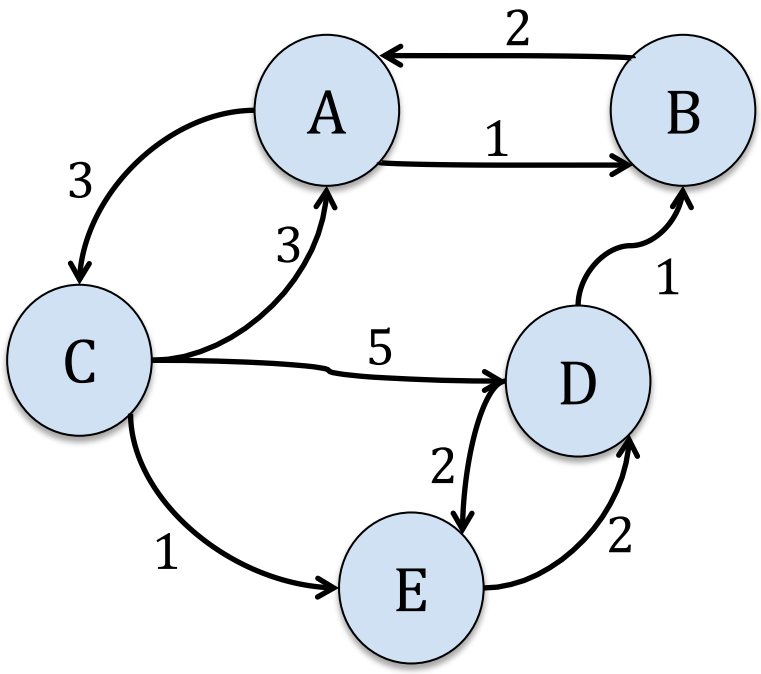
Traccia di possibili soluzioni

Esercizio 1

1. Ascolta il canale per un tempo predefinito Ta e, se il canale resta libero, trasmette la trama
2. La persistenza p è la probabilità d trasmettere immediatamente quando il canale si libera, posto che una stazione abbia qualcosa da trasmettere e non ci siano state collisioni.
3. La funzione CD serve a identificare una collisione mentre si sta trasmettendo, consentendo di interrompere immediatamente la trasmissione in modo da non sprecare tempo sul canale. Per la sua corretta implementazione, cioè per consentire a tutte le stazioni di sapere che vi è stata una collisione, è necessario trasmettere una speciale sequenza di bit, detta di "jamming" di lunghezza opportuna per garantire che tutte le stazioni rilevino la collisione.
4. Si ascolta il canale durante la trasmissione per rilevare se vi è più potenza (o energia) rispetto a quella iniettata per la trasmissione stessa.
5. Il preambolo serve a consentire la sincronizzazione in frequenza, fase e tempo del ricevitore sull'orologio del trasmettitore.
6. Ad esempio in Ethernet, dove il preambolo è una sequenza 10101010 ... la sequenza 11 indica la fine del preambolo e l'inizio della trama. In altri protocolli si usa un byte di flag, ad esempio 01111110. Si deve poi garantire che non si ripeta "per caso" nei dati, ciò si ottiene con con tecniche dette di bit o byte "stuffing".

Esercizio 2

1. La rete è la seguente



2. I link sono asimmetrici come costo, quindi la rete va descritta come un grafo diretto.
3. OSPS usa l'algoritmo di Djikstra, che calcola il MST con radice nel nodo che sta eseguendo l'algoritmo. Le due istanze dell'algoritmo sono rappresentabili dalle tabelle nella figura 1 che riassumono i 5 passi necessari, si noti come i due alberi sono differenti tra loro. Il costo dei diversi cammini è calcolato applicando la formula ricorsiva D(i,j) = D(i,k) + c(k,j), dove D() è il costo corrente del cammino e c() è il costo del link. L'algoritmo esplora il grafo aggiungendo al set N di nodi raggiungibili il nodo a costo minimo ad ogni passo. In caso di pareggio si sceglie a caso.
4. In caso di link diretti con interfacce di rete separate la procedura di flooding prevede la trasmissione di esattamente una copia del messaggio per ciascuna interfaccia, quindi nella rete analizzata avremo 9 copie del messaggio, una per ogni link.

Nodo A

passo	Set N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(E),p(E)
0	A	**1,A**	3,A	-,-	-,-
1	AB		**3,A**	-,-	-,-
2	ABC			8,C	**4,C**
3	ABCE			**6,E**	
4	ABCED				

Nodo E

passo	Set N	D(B),p(B)	D(C),p(C)	D(D),p(D)	D(A),p(A)
0	E	-,-	-,-	**2,E**	-,-
1	ED	**3,D**	-,-		-,-
2	EDB		-,-		**5,B**
3	EDBA		**8,A**		
4	EDBAC				

Fig. 1: Tabelle di calcolo per l'algoritmo di Djikstra ai nodi A ed E

Esercizio 3

1. UDP: protocollo connectionless; fornisce solamente funzioni di framing (no segmentazione) e multiplazione; fornisce la funzione di rilevazione delle trame errate con un checksum a 16 bit (opzionale, i bit possono essere messi a zero). UDP non garantisce la consegna dei segmenti né l'ordine di consegna. Non ha controllo di flusso e congestione.
   TCP: protocollo orientato alla connessione, affidabile con consegna ordinata delle informazioni (ad eccezione dei segmenti che usano il flag URG); fornisce funzioni di framing e segmentazione, controlla la correttezza dell'informazione trasmessa (checksum a 16 bit analogo a quello di UDP, ma obbligatorio) e richiede la ritrasmissione dei segmenti errati o mancanti; fornisce controllo di flusso e anche di congestione nella rete. Il protocollo è bi-direzionale e gli acknowledgement possono essere associati ai dati nella direzione opposta (piggy-backing).
2. La fase di slow start è implementata all'apertura della connessione e implica che la finestra di trasmissione venga aumentata di 1 MSS ogni ACK valido ricevuto. In condizioni normali questo porta al raddoppio della finestra di trasmissione ogni RTT, e quindi una crescita geometrica di ragione 2 (in segmenti) nel tempo.
   La fase di congestion avoidance, che viene usata quando la finestra di trasmissione supera una soglia detta SSTHR, implica che la finestra di trasmissione venga aumentata di MSS/CNGW (in byte) ogni ACK valido ricevuto. L'effetto è quello di aumentare la finestra di trasmissione di un solo segmento (MSS) ogni RTT e quindi avere una crescita della finestra lineare nel tempo.
3. Con MTU 576 si ha MSS=536 (si sottraggono 20 bytes di header IP e 20 di TCP), quindi la receiver window equivale a int (65536 bytes / 536 bytes) = 122 segmenti, la frazione di segmento oltre il 122^ non verrà in generale sfruttata grazie all'algoritmo di silly window avoidance.
   SSHTR = RCW/2 = 61 segmenti. SSHTR viene raggiunta in $\log_2(61)$ arrotondato all'intero superiore perché appena viene raggiunta la soglia TCP passa da Slow Start a Congestion avoidance e quindi aspetta la trasmissione di una intera finestra prima di incrementare la stessa di un MSS. Quindi vi vogliono 7 RTT per raggiungere SSHTR.
   In seguito la finestra aumenta di 1 segmento per RTT, quindi la finestra di trasmissione raggiunge RCW in 61+7=68 RTT.

4. UDP ha un semplice header da 8 bytes, quindi la dimensione dei suoi segmenti è di 576 – 28 = 548 bytes.
In entrambi i casi bisogna considerare anche l'overhead del framing di livello 2. Supponendo di usare Ethernet abbiamo un overhead di 22 bytes. In realta dipende ancora dallo specifico livello fisico usato, ma una qualsiasi approssimazione tra 16 e 32 bytes è valita in assenza di ulteriori specifiche).
UDP non ha alcuna forma di controllo della trasmissione, quindi sfrutterà tutti i 100Mbit/s. In questo caso basta calcolare l'overhead dei protocolli, cioè 50 bytes ogni 548 bytes utili ovvero il 9.12%, per calcolare la quantità di dati da trasmettere al livello fisico, ovvero $1.0912 * 8 * 2^{30}$ = 9,373,336,627 bit = 9.373 Gbit.
Il tempo di trasferimento del file con UDP è quindi circa **9.373 Gbit / 0.1 Gbit/s = 93.7 s.**
Per quanto riguarda TCP invece bisogna prima di tutto capire se RCW consente di sfruttare tutta la banda necessaria. In particolare possiamo calcolare il prodotto banda-ritardo: 100Mbit/s*RTT = 0.2*100 Mbit/s / 8 = 2.5 Mbyte >> RCW, quindi il ritardo è dominato dal numero di RTT che TCP impiegherà a trasferire il file, e quindi indipendente dall'overhead in byte, ma dipendente solo dal numero di segmenti. Il file verrà segmentato in $2^{30}/536$ = 2,003,250 segmenti, che vengono trasmessi a finestre di 122 segmenti ciascuno (stiamo trascurando il transitorio iniziale di crescita della finestra, ma è lecito date le dimensioni del file da trasmettere). Il tempo necessario al trasferimento è quindi
2003250/122*0.2 = 3284 s