Quesito 1 (punti 8). Cinque processi *batch*, identificati dalle lettere $A - E$ rispettivamente, arrivano all'elaboratore agli istanti $0, 1, 3, 5, 8$ rispettivamente. Tali processi hanno un tempo di esecuzione stimato di $3, 7, 4, 6, 2$ unità di tempo rispettivamente. Per ognuna delle seguenti politiche di ordinamento:

1. FCFS (un processo per volta, sino al completamento)
2. RR (divisione di tempo, senza priorità e con quanto di tempo di ampiezza 2)
3. RR (divisione di tempo, con priorità e prerilascio, e quanto di tempo di ampiezza 2)
4. SJF_{SRTN} (senza considerazione di valori di priorità espliciti[1] e con prerilascio)

determinare, trascurando i ritardi dovuti allo scambio di contesto: (i) il tempo medio di risposta; (ii) il tempo medio di attesa; (iii) il tempo medio di *turn around*.
Ove la politica di ordinamento in esame consideri i valori di priorità, tali valori, mantenuti staticamente per l'intera durata dell'esecuzione, sono rispettivamente: $2, 3, 5, 3, 2$ (con 5 valore maggiore).
Nel caso di arrivi simultanei di processi allo stato di pronto, fatta salva l'eventuale considerazione del rispettivo valore di priorità, si dia la precedenza ai processi usciti dallo stato di esecuzione rispetto a quelli appena arrivati.

Quesito 2 (punti 10). In una pubblicazione del 1968 sulla rivista IBM Systems Journal, lo studioso James W. Havender propose una strategia per prevenire l'insorgere di situazioni di stallo (*deadlock*) in presenza di condivisione di risorse da parte di processi concorrenti. La strategia di Havender imponeva al sistema l'adozione di almeno una tra le seguenti condizioni:

1. ogni processo deve richiedere tutte le risorse necessarie in una sola volta e non può procedere finché non gli siano state tutte garantite
2. un processo che già detenga risorse a cui venga negato l'accesso a una ulteriore risorsa, deve rilasciarle tutte e, se necessario, richiederle tutte (inclusa quella aggiuntiva) di nuovo come stipulato al punto 1
3. a tutti i processi viene imposto un ordinamento lineare (p.es.: identificatore crescente) delle risorse: un processo che già detenga risorse, potrà richiederne altre solo in modo conforme all'ordinamento fissato (p.es.: il processo P_a che già possieda le risorse $\{R_1, R_3, R_6\}$ potrà richiedere la risorsa R_7 ma non la risorsa R_2).

Si discuta se e in che modo le condizioni proposte da Havender concorrano effettivamente a evitare strutturalmente lo stallo. In caso affermativo si discutano eventuali controindicazioni all'applicazione della strategia proposta.

Quesito 3 (punti 6). Si specifichino le finalità degli algoritmi denominati rispettivamente leaky bucket e token bucket e si discutano concisamente le principali differenze algoritmiche che intercorrono tra loro.

Quesito 4 (punti 8). Lo schema logico riportato in figura 1 rappresenta la rete dati di una piccola azienda composta da due reparti distinti, *produzione* e *amministrazione*. In tale rete sono presenti tre nodi serventi: (1) *Web server* che offre servizi *Web* accessibili prevalentemente da utenti esterni all'azienda e che compone la zona *DMZ* dell'azienda; (2) *server_A* che offre servizi vari ai 6 utenti del reparto *amministrazione*; (3) *server_P* che offre servizi vari ai 3 utenti del reparto *produzione*.

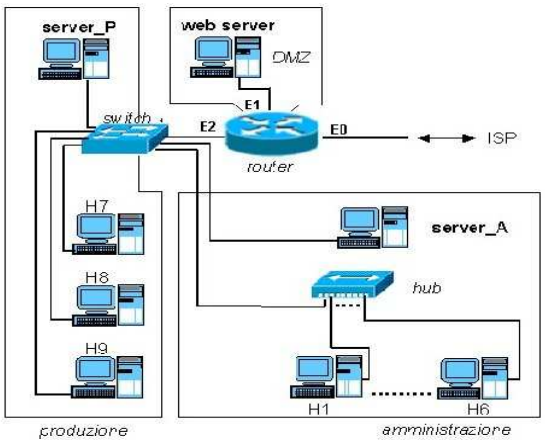

Figura 1: Architettura della rete interna dell'azienda.

[1] Esclusi ovviamente i valori di priorità impliciti determinati dalla durata (residua) dei processi.

Il reparto *amministrazione* è composto da 6 utenti ($H1 - H6$), caratterizzati da un traffico prevalentemente di tipo utente-servente facente capo al *server_A*. Il reparto *produzione* è invece composto da 3 postazioni di lavoro ($H7 - H9$), tutte caratterizzate da un traffico di tipo utente-servente facente capo al *server_A*, le prime 2 delle quali contraddistinte anche da un traffico multimediale di tipo utente-servente abbastanza costante, stimato in 5 Mbps, facente capo al *server_P*.
Sotto queste ipotesi, determinare i valori massimi teorici del traffico nei vari remi della rete nel caso peggiore, ove cioè tutti i singoli utenti necessitino del loro traffico massimo.

L'azienda accede a *Internet* mediante un unico indirizzo IP statico fornito direttamente dal proprio ISP, che le attribuisce anche una sottorete di indirizzi IP pubblici da utilizzare liberamente al suo interno:

configurazione della porta del *router* $E0$ (lato ISP)	
IP address	$150.1.0.125$
subnet mask	$255.255.255.192$
default gateway	$150.1.0.126$
sottorete a disposizione dell'azienda (da ripartire internamente)	
Subnet IP address	$150.1.0.128$
subnet mask	$255.255.255.192$

Sotto queste ipotesi, determinare la configurazione IP (*IP address*, *subnet mask* e *default gateway*) di tutti i dispositivi della rete aziendale, facendo in modo che abbiano tutti la stessa *subnet mask*, e quindi che le sottoreti interne abbiano tutte la stessa dimensione.

Soluzione 1 (punti 8).

- FCFS (un processo per volta, sino al completamento)

```
processo A  AAA                      LEGENDA DEI SIMBOLI
processo B  -bbBBBBBBB               - non ancora arrivato
processo C  ---ccccccCCCC            x (minuscolo) attesa
processo D  -----ddddddddDDDDDD      X (maiuscolo) esecuzione
processo E  --------eeeeeeeeeeeeEE   . coda vuota

CPU         AAABBBBBBBCCCCDDDDDDEE
coda        .bbccccccddddeeeeee..
            .....dddddeeee........
            ........ee............
```

processo	tempo di risposta	tempo di attesa	*turn-around*
A	0	0	0+3= 3
B	2	2	2+7= 9
C	7	7	7+4=11
D	9	9	9+6=15
E	12	12	12+2=14
medie	6,00	6,00	10,40

- RR (divisione di tempo, senza priorità e con quanto di tempo di ampiezza 2)

```
processo A  AAaaA                    LEGENDA DEI SIMBOLI
processo B  -bBBbbbBBbbbbbbBBbbB     - non ancora arrivato
processo C  ---ccCCccccCC            x (minuscolo) attesa
processo D  -----ddddDDddddddDDdDD   X (maiuscolo) esecuzione
processo E  --------eeeeeEE          . coda vuota

CPU         AABBACCBBDDCCEEBBDDBDD
coda        .baacbbddcceebbddbbd..
            ...cbddcceebbdd.......
            ........ebbdd.........
```

processo	tempo di risposta	tempo di attesa	*turn-around*
A	0	2	2+3= 5
B	1	1+3+6+2=12	12+7=19
C	2	2+4= 6	6+4=10
D	4	4+6+1=11	11+6=17
E	5	5	5+2= 7
medie	2,40	7,20	11,60

- RR (divisione di tempo, con priorità e prerilascio, e quanto di tempo di ampiezza 2)

```
processo A  AaaaaaaaaaaaaaaaaaAA     LEGENDA DEI SIMBOLI
processo B  -BBbbbbBBbbBBbbB         - non ancora arrivato
processo C  ---CCCC                  x (minuscolo) attesa
processo D  -----ddddDDddDDdDD       X (maiuscolo) esecuzione
processo E  --------eeeeeeeeeeeeEE   . coda vuota

CPU         ABBCCCCBBDDBBDDBDDAAEE
coda        .aabbbbddbbddbbdaaee..
            ...aaddaaaaaaaaaee....
            .....aa.eeeeeeee......
```

processo	risposta	tempo di attesa	*turn-around*
A	0	17	17+3=20
B	0	4+2+2= 8	8+7=15
C	0	0	0+4= 4
D	4	4+2+1= 7	7+6=13
E	12	12	12+2=14
medie 3,20	8,80	13,20	

- $\mathsf{SJF}_{\mathsf{SRTN}}$ (senza considerazione di valori di priorità espliciti e con prerilascio)

```
processo A  AAA                     LEGENDA DEI SIMBOLI
processo B  -bbbbbbbbbbbbbbbBBBBB   - non ancora arrivato
processo C  ---CCCC                 x (minuscolo) attesa
processo D  -----ddDddDDDDD         X (maiuscolo) esecuzione
processo E  --------EE              . coda vuota

CPU         AAACCCCDEEDDDDDBBBBBB
coda        .bbbbbbbddbbbbb......
            .....dd.bb..........
```

processo	risposta	tempo di attesa	*turn-around*
A	0	0	0+3= 3
B	14	14	14+7=21
C	0	0	0+4= 4
D	2	2+2=4	4+6=10
E	0	0	0+2= 2
medie	3,20	3,60	8,0

Soluzione 2 (punti 10). Dalla diapositiva 33 del corso conosciamo le 4 condizioni che concorrono all'insorgere di situazioni di stallo. Per essere efficace, una strategia di prevenzione, e dunque anche quella proposta da Havender, deve essere capace di impedire il verificarsi di almeno una di tali condizioni. Per risolvere il quesito dobbiamo dunque studiare se e quale condizione di stallo ciascuno dei 3 requisiti proposti da Havender effettivamente previene:

- il requisito 1 previene l'accumulo parziale di risorse, perchè stipula che un processo possa eseguire (già dall'inizio) solo avendo acquisito tutte le risorse necessarie; naturalmente questa strategia è pesantemente inefficiente per almeno due motivi: (1) perché assegna più risorse di quante ne siano probabilmente necessarie, visto che l'assegnazione deve contemplare il caso di massima domanda, e per un tempo probabilmente più lungo del necessario; e (2) perché comporta per i processi il serio rischio di posticipazione indefinita
- il requisito 2 di fatto realizza il prerilascio preventivo delle risorse perché impegna il processo cui venisse negata una risorsa supplementare a rilasciare immediatamente tutte quelle in suo possesso; questa strategia ínefficiente perchè rischia di impedire al processo in questione di completare il proprio lavoro sulle risorse in proprio possesso e di doverlo ricominciare, eventualmente anche daccapo (a seconda del tipo di risorsa e dell'operazione interrotta) in un momento successivo
- il requisito 3 impedisce il formarsi di condizioni di attesa circolare; si lascia come esercizio al lettore la semplice verifica di questa asserzione. È facile comprendere che anche in questo caso l'applicazione di una tale strategia risulta troppo rigida per essere utilmente applicabile in un sistema reale.

Soluzione 3 (punti 6). Gli algoritmi detti *leaky bucket* e *token bucket* vengono entrambi utilizzati per il controllo di congestione del traffico di rete, ciò che dunque li colloca al livello rete (*network*, IP) nella gerarchia dei protocolli ISO/OSI e TCP/IP. L'algoritmo *leaky bucket* produce un flusso di uscita costante o nullo a fronte di flussi di ingresso variabili e irregolari. Concettualmente, esso può essere rappresentato come dotato di un contenitore di ingresso a coda (*bucket*: secchio) a capacità prefissata e un'attività di estrazione da esso a periodo fissato e quantità costante di N pacchetti per unità di tempo. Il traffico

in ingresso si accoda nel contenitore, ma solo finché vi trova spazio, altrimenti viene perduto. Il traffico in uscita viene invece prodotto a velocità costante, fintantoché vi siano pacchetti in coda.
L'algoritmo *token bucket* può essere visto come una variante del precedente, nella quale ciascun pacchetto in ingresso viene avviato all'uscita solo consumando un gettone (*token*) appositamente prelevato dal *bucket*, secondo un rapporto configurabile di S B per gettone. L'algoritmo produce gettoni secondo una legge fissata (p.es.: periodo fissato e costante, M gettoni per unità di tempo) e li accumula nel *bucket* fino alla sua capacità massima. In assenza di traffico in ingresso l'algoritmo accumula capacità di uscita, producendo così traffico in uscita di profilo variabile, con periodi di stasi e con picchi. Inoltre, a differenza del *leaky bucket*, l'algoritmo *token bucket* non contempla perdita di pacchetti in ingresso (posto che disponga di una coda di pacchetti in ingresso sufficientemente grande) ma solo di gettoni, in presenza di *bucket* pieno e traffico in ingresso nullo.

Soluzione 4 (punti 8). Le tre porte del *router*1 ($E0, E1E2$) identificano altrettanti reti locali:

LAN-0	rete esterna, di collegamento tra l'ISP e la rete aziendale
LAN-1	settore non protetto della rete aziendale (DMZ), contenente i servizi *Web* esternalizzati
LAN-2	rete aziendale interna, composta da un unico dominio di diffusione

Il calcolo dei flussi nel caso peggiore può essere affrontato semplicemente limitando l'analisi alla sola rete interna LAN-2. Per prima cosa si individueranno i flussi utili, ovvero quelli descritti dal que sito. Detti:

- X flusso di dati scambiato con il *server_A* dal generico utente di reparto *amministrazione* ($H1 \ldots H6$)
- Y flusso di dati scambiato con il *server_A* dal generico utente del reparto *produzione* ($H7 \ldots H9$)
- Z flusso aggiuntivo di dati scambiato con il *server_P* da ciascuna delle 2 postazioni multimediali $H7 - H8$ del reparto *produzione*, di valore noto pari a $5\ Mbps$;

si ottiene facilmente la distribuzione rappresentata in figura 2, nella quale il traffico utile è indicato in grassetto, ed il traffico effettivo (quando diverso da quello utile) è racchiuso in un ovale.

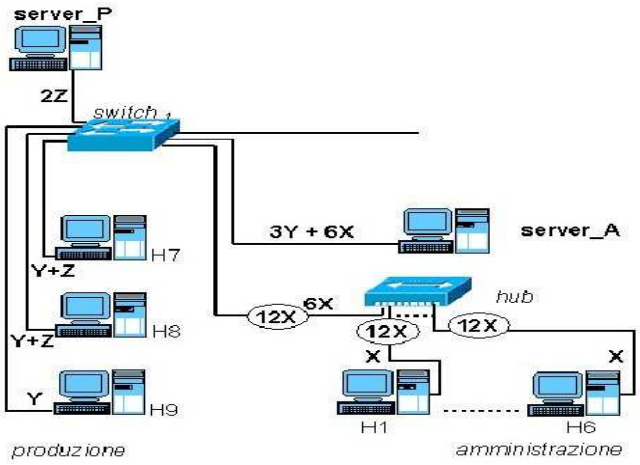

Figura 2: Determinazione dei flussi utili di caso peggiore.

Assumiamo ora che ogni ramo della rete possa trasferire al massimo $100\ Mbps$ (notando che la metodologia di soluzione non dipende ovviamente dalla portanza della rete). Fissato questo dato, otteniamo le seguenti condizioni:

$$6X + 6X \leq 100\ Mbps \quad \text{(porte dell'}\textit{hub}\text{)} \quad (1)$$

$$3Y + 6X \leq 100\ Mbps \quad \text{(collegamento al }\textit{server\_A}\text{)} \quad (2)$$

$$Y \leq 100\ Mbps \quad \text{(host } H9\text{)} \quad (3)$$

$$Y + Z \leq 100\ Mbps \quad \text{(host } H7 - H8\text{)} \quad (4)$$

$$2Z \leq 100\ Mbps \quad \text{(collegamento al }\textit{server\_P}\text{)} \quad (5)$$

Dalla condizione (2) ricaviamo: $X \leq 8,3\ Mbps$. Dalla condizione (3), prendendo per buono il valore massimo di $X = 8,3\ Mbps$, ricaviamo: $Y = \frac{100-6X}{3} = \frac{100-50}{3} = 16,6\ Mbps$. Dalla condizione (4) ricaviamo: $Y \leq 100\ Mbps$, che essendo meno restrittivo del valore già attribuito ad Y, può essere trascurato. Dalla condizione (5) ricaviamo: $Y \leq 95\,Mbps$, ancora una volta meno restrittivo di quanto già ricavato per Y, e che dunque trascuriamo. La condizione (5) infine viene automaticamente soddisfatta dato che $Z \leq 5\ Mbps$ per definizione. Globalmente, i valori massimi teorici del traffico nei vari punti della rete, nel caso peggiore in cui tutti gli utenti necessitino del traffico massimo, valgono rispettivamente:

$$X_{MAX} = 8,3\ Mbps \qquad Y_{MAX} = 16,6\ Mbps.$$

Per quanto riguarda la ripartizione degli indirizzi IP della rete aziendale, l'indirizzo IP statico acquistato dall'azienda è un dato di fatto ininfluente, dato che la configurazione della porta $E0$ del *router* è imposta dall'ISP medesimo. Per gli altri dispositivi abbiamo invece a disposizione una sottorete di indirizzi pubblici $150.1.0.128$, originariamente appartenenti alla classe B, ma ora associati alla maschera $/26$ e dunque resi *classless* (ricordiamo che la notazione $/26$ è la forma compatta di: $255.255.255.192$). Come mostrato in figura 3, la rete interna comprende 2 sole sottoreti: $LAN-1$ e la $LAN-2$.

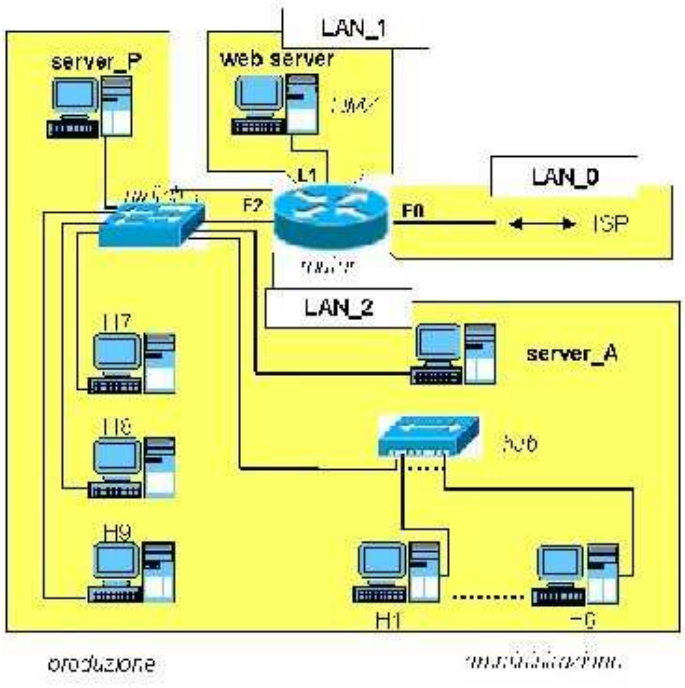

Figura 3: Architettura delle sottoreti interne all'azienda.

Il dato di progetto da cui partire per la ripartizione (*subnetting*) è chiaramente il requisito che le sottoreti abbiano tutte la stessa dimensione, ossia la stessa *subnet mask*. Dovendo creare 2 sottoreti utili e sapendo che il loro numero S è determinato dall'equazione $S = 2^N - 2$, dove N è il numero di *bit* impiegati per designarle, fissando $S = 2$ otteniamo $N = 2$. Il numero di nodi della sottorete $LAN-2$ deve essere ≥ 12 , poiché abbiamo 9 *host*, 2 *server* ed 1 porta *router*. Ne segue che la parte '*host*' dell'indirizzo IP deve essere composto da (almeno) 4 *bit*, dato che ciò consente di rappresentare $2^4 - 2 = 14$ indirizzi IP utili. In definitiva, per le 2 reti aziendali dobbiamo utilizzare 2 *bit* per designare la sottorete corrispondente e 4 *bit* per designare i rispettivi nodi.
La *subnet mask* fornita all'azienda dall'ISP vale:

$255.255.255.192$ in binario: $11111111.11111111.11111111.11000000$

lasciando a nostra discrezione proprio i 6 *bit* che ci servono. La sottorete assegnata soddisfa quindi esattamente le esigenze interne dell'azienda, la cui *subnet mask* interna varrà:

$255.255.255.240$ in binario: $11111111.11111111.11111111.11110000$

150	1	0	128		
parte di rete (classe B annullata da maschera)				*sottorete (dall'ISP)*	
10010110	00000001	00000000	10	00	0000
				sottoreti interne	*nodi di sottorete*

Tabella 1: Base dell'insieme di indirizzi di classe B riservati a uso interno dell'azienda.

In tabella 2 ricapitoliamo le caratteristiche degli indirizzi IP delle 2 sottoreti utili, mentre in tabella 3 riportiamo una possibile assegnazione di indirizzi IP per i dispositivi aziendali interni.

parte di rete (*classless*)		sottorete interna	nodo		destinazione indirizzo
10010110. 00000001. 00000000.	10	01	0000	150.1.0.144	riservato I sottorete
10010110. 00000001. 00000000.	10	01	0001	150.1.0.145	utilizzabile per 1^o nodo I sottorete
...					
10010110. 00000001. 00000000.	10	01	1110	150.1.0.158	utilizzabile per ultimo nodo I sottorete
10010110. 00000001. 00000000.	10	01	1111	150.1.0.159	riservato a *broadcast* I sottorete
10010110. 00000001. 00000000.	10	10	0000	150.1.0.160	riservato II sottorete
10010110. 00000001. 00000000.	10	10	0001	150.1.0.161	utilizzabile per 1^o nodo II sottorete
...					
10010110. 00000001. 00000000.	10	10	1110	150.1.0.174	utilizzabile per ultimo nodo II sottorete
10010110. 00000001. 00000000.	10	10	1111	150.1.0.175	riservato a *broadcast* II sottorete

Tabella 2: Ripartizione degli indirizzi nelle 2 sottoreti interne utili.

rete	**dispositivo**	*IP address*	*subnet mask*	*default gateway*
$LAN-1$	*Web server*	150.1.0.157	255.255.255.240	150.1.0.158
	router $E1$	150.1.0.158	/28	—
$LAN-2$	$H1$	150.1.0.161	/28	150.1.0.174
	$H2$	150.1.0.162	/28	150.1.0.174
	$H3$	150.1.0.163	/28	150.1.0.174
	$H4$	150.1.0.164	/28	150.1.0.174
	$H5$	150.1.0.165	/28	150.1.0.174
	$H6$	150.1.0.166	/28	150.1.0.174
	$H7$	150.1.0.167	/28	150.1.0.174
	$H8$	150.1.0.168	/28	150.1.0.174
	$H9$	150.1.0.169	/28	150.1.0.174
	server_A	150.1.0.172	/28	150.1.0.174
	server_P	150.1.0.173	/28	150.1.0.174
	router $E2$	150.1.0.174	/28	—

Tabella 3: Una possibile attribuzione di indirizzi IP interni alle postazioni della rete aziendale.