Quesito 1. Il progettista che abbiamo incontrato nel quesito 2 del primo appello d'esame di questa sessione, vuole proseguire la propria comparazione tra diverse architetture di *file system*, usando come metrica il rapporto inflattivo da esse prodotto tra l'ampiezza di memoria complessiva necessaria e quella effettivamente richiesta per l'allocazione di un *file* tipo composto da 10.000 strutture indivisibili, non partizionabili su 2 blocchi distinti, ciascuna ampia 100 *byte* (B). Vogliamo questa volta fornire al nostro progettista il dato riguardante l'architettura Ext2fs, sotto le seguenti condizioni:

- blocchi su disco e nodi indice (*i-node*) ampi 1 kB
- indici di blocco su disco e di nodo indice ampi 64 *bit*
- per ciascun nodo indice principale, 16 indici diretti, ed 1 indice ciascuno per blocchi di prima, seconda e terza indirezione
- sul disco sono presenti 100 sequenze distinte di blocchi contigui, ciascuna ampia 25 blocchi.

Vogliamo poi confrontare nel merito il risultato ottenuto con il valore di $2,6\%$ già determinato per NTFS.

Quesito 2. Cinque processi "a lotti" (*batch*), ossia con singola esecuzione non ripetitiva, identificati dalle lettere dalla *A* alla *E*, arrivano all'elaboratore agli istanti $0,2,4,6,8$ rispettivamente. Tali processi hanno tempo di esecuzione stimato di $3,5,3,9,5$ unità di tempo e valore di priorità statica pari a $2,5,3,1,4$ (con 5 valore di priorità massima) rispettivamente. Per ognuno degli algoritmi di ordinamento sotto indicati, si determinino il tempo medio di risposta, il tempo medio di *turn-around*, ed il tempo medio di attesa, trascurando i ritardi dovuti allo scambio di contesto:

- FCFS (*First-Come-First-Served*), un processo per volta, fino al suo completamento
- *Round Robin*, a divisione di tempo, con quanto ampio 3 unità di tempo ed immediata riassegnazione di frazioni di quanto inutilizzate

  Caso 1: senza priorità. *Caso 2*: con priorità, ma senza prerilascio. *Caso 3*: con priorità e con prerilascio
- SJF (*Shortest Job First*), trascurando la priorità statica di ciascun processo

  Caso 1: senza prerilascio. *Caso 2*: con prerilascio.

In caso di arrivo simultaneo in coda dei pronti da parte di processi appena creati e di processi già attivati ma sottoposti a prerilascio, si dia precedenza a quelli già attivati.

Quesito 3. Il caricamento dinamico di moduli di servizio e di altre utilità quali, per esempio, i gestori di dispositivi rimuovibili, richiede al nucleo del sistema operativo la capacità di assegnar loro zone contigue di memoria principale. Una tecnica che facilita l'acquisizione ed il rilascio di zone contigue di dimensione arbitraria è nota come "*buddy algorithm*", una variante correttiva del quale è adottata da Linux.
Si mostri, passo per passo, l'effetto dell'applicazione del "*buddy algorithm*" su una memoria principale paginata, con pagine ampie 4 kB, a partire dalla situazione iniziale mostrata in figura 1 a fronte della sequenza di richieste e rilasci di blocchi di pagine riportata in figura 2. Successivamente, si indichi il principale difetto del "*buddy algorithm*", suggerendo come esso possa essere attenuato o corretto.

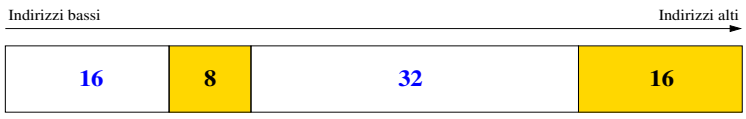

Figura 1: Stato iniziale della memoria principale. I rettangoli in chiaro mostrano le sequenze di pagine libere disponibili; quelli in scuro le sequenze di pagine occupate.

tempo	evento	quantità (kB)
1	P1.richiesta	12
3	P2.richiesta	20
5	P3.richiesta	24
7	P4.richiesta	28
9	P2.rilascio	20
11	P1.rilascio	12
13	P5.richiesta	36
15	P3.rilascio	24
17	P5.rilascio	36
19	P4.rilascio	28

Figura 2: Sequenza di richieste e rilasci di blocchi di pagine da trattare, con valori espressi in kB.

Quesito 4. Dato il sistema descritto dalla seguente rappresentazione insiemistica di assegnazione delle risorse:

$$\begin{aligned} P &= \{P_1, P_2, P_3, P_4, P_5, P_6\} \\ R &= \{R_1^1, R_2^2, R_3^2, R_4^1\} \\ E &= \{P_1 \to R_3, P_2 \to R_1, P_2 \to R_3, P_2 \to R_4, P_3 \to R_4, P_5 \to R_2, P_6 \to R_2, \\ &\quad R_1 \to P_1, R_2 \to P_2, R_2 \to P_3, R_3 \to P_4, R_3 \to P_5, R_4 \to P_6\} \end{aligned}$$

si rappresenti il corrispondente grafo di allocazione delle risorse, verificando se il sistema si trovi attualmente in situazione di stallo oppure no. Successivamente, si verifichi se e come la situazione si modifichi qualora il processo P_4 richieda l'uso della risorsa R_1.

Quesito 5. La modalità di assegnazione degli indirizzi IP detta *Classless Inter-Domain Routing* (CIDR) si propone di alleviare il problema della sempre maggiore scarsità di indirizzi liberi per nuove reti. Per evitare che l'uso di tale tecnica appesantisca oltre modo le tabelle di instradamento dei *router*, si tende ad assegnare indirizzi a reti in modo che gruppi di essi possano essere aggregati nelle tabelle di quanti più *router* possibile. Assumendo la disponibilità presso l'ICANN di 2^{14} indirizzi IP consecutivi a partire dal valore 194.24.0.0, si mostri un'assegnazione CIDR a fronte delle seguenti richieste, pervenute e soddisfatte in quest'ordine: (1) rete unive2: 1.800 indirizzi; (2) rete univr2: 1.000 indirizzi; (3) rete unipd2: 3.700 indirizzi; illustrandone anche una possibile aggregazione a fini di *routing*.

Quesito 6. Lo schema logico riportato in figura 3 rappresenta la rete dati di una piccola Azienda composta da due reparti operativi ed una stanza per i gestori della rete.

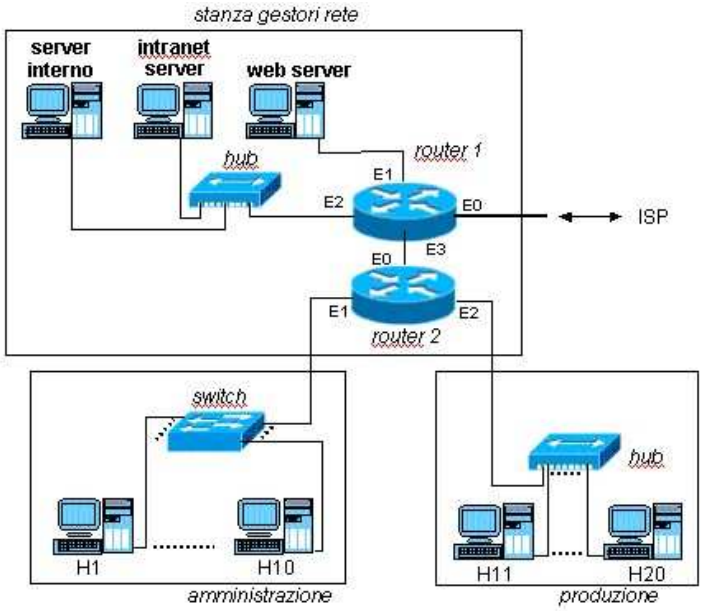

Figura 3: Schema logico della rete dati dell'azienda.

Il reparto *amministrazione* è composto da 10 postazioni di lavoro il cui traffico è prevalentemente di tipo utente-servente, le quali fanno capo al server interno. Le 10 postazioni del reparto *produzione* sono invece caratterizzate da un traffico al 50% di tipo utente-servente e facente capo al server interno, e per il rimanente 50% diretto verso l'intranet server. La postazione web server, infine, posizionata nella *stanza gestori rete*, offre servizi *Web* accessibili prevalentemente da utenti esterni all'Azienda. Sapendo che tutti i dispositivi di rete sono di standard *Fast-Ethernet*, e quindi operano a 100 Mbps, si calcolino i flussi di traffico massimo determinati dalla configurazione fisica della rete nel caso peggiore di traffico simultaneo da parte di tutti gli utenti.

L'Azienda accede ad *Internet* mediante un unico indirizzo IP statico fornito direttamente dal proprio ISP. Al suo interno, invece, decide di condividere gli indirizzi privati di una intera classe C (192.168.129.*X*), sfruttando la funzione di traduzione degli indirizzi (NAT) disponibile all'interno del proprio *router*. Si proponga una suddivisione degli indirizzi utili in sottoreti di uguali dimensioni e con medesima *subnet mask*. Si compili poi una tabella di associazione tra gli indirizzi IP disponibili ed i dispositivi di rete dell'Azienda.

Soluzione 1 (punti 5). Dalla soluzione del citato quesito 2 (che non ripeteremo qui) sappiamo che sono necessari $T = 1.000$ blocchi per ospitare il *file* campione con le caratteristiche data. Per descrivere tale *file* avremo dunque bisogno di T indici di blocco. Di questi, 16 sono immediatamente disponibili nel nodo indice principale. Sapendo che ciascun indice di blocco occupa $I = \frac{64}{8} = 8$ B, per descrivere i restanti $R_1 = 1.000 - 16 = 984$ blocchi, dovremo ulteriormente utilizzare del nodo indice principale:

1. l'indice di prima indirezione, tramite il quale potremo accedere ad $E_1 = 1$ nodo indice di prima indirezione, interamente utilizzato per contenere $S = \frac{1.024}{I} = 128$ indici di blocco, con ciò restandone $R_2 = 984 - 128 = 856$

2. l'indice di seconda indirezione, tramite il quale accederemo ad $E_2 = 1$ nodo indice di seconda indirezione, che può contenere al più $K = \frac{1.024}{I} = 128$ indici di nodi indice di prima indirezione

3. ricordando dal passo 1 che un singolo nodo indice di prima indirezione descrive al più $S = 128$ indici di blocco, ci basterà utilizzare $E_3 = \lceil \frac{R_2}{S} \rceil = \lceil \frac{856}{128} \rceil = 7$ indici di nodi indice di prima indirezione nel nodo indice di seconda indirezione (così sprecandone il $\frac{K-E_3}{K} = \frac{121}{128} = 94.5\%$!) per descrivere l'intero *file*.

Dovremo dunque utilizzare $V = 1 + E_1 + E_2 + E_3 = 1 + 1 + 1 + 7 = 10$ nodi indice (ciascuno di dimensione uguale ad 1 blocco su disco), da aggiungersi ai T blocchi necessari per il *file* stesso, così producendo un rapporto inflattivo di $\frac{(1.000+10)\ \text{kB}}{10.000 \times 100\ \text{B}} = 3.4\%$, superiore dello 0.8% al dato ottenuto per NTFS.
La ragione di tale incremento, indicatore di prestazioni inferiori rispetto al criterio di raffronto adottato dal nostro progettista, è con tutta evidenza determinata dal fatto che Ext2fs non trae alcun beneficio dall'eventuale contiguità dei blocchi su disco, necessitando di un numero di indici di blocco linearmente proporzionale all'ampiezza del *file*.

Soluzione 2 (punti 5). Richiamiamo le definizioni: tempo risposta è il tempo che intercorre tra l'arrivo del processo e la sua prima selezione per l'esecuzione; tempo di attesa è la somma dei tempi che il processo trascorre nella coda dei pronti; tempo di *turn-around* è il tempo che intercorre tra l'arrivo del processo ed il suo completamento. Notiamo inoltre che occorrerà considerare con attenzione l'annotazione finale del quesito che specifica il criterio di precedenza da applicare in caso di arrivo simultaneo di processi nella coda dei pronti.

- FCFS (*First-Come-First-Served*), un processo per volta, fino al suo completamento

```
processo A     AAA                          LEGENDA DEI SIMBOLI
processo B     --bBBBBB                     - non ancora arrivato
processo C     ----ccccCCC                  x (minuscolo) attesa
processo D     ------dddddDDDDDDDDD         X (maiuscolo) esecuzione
processo E     --------eeeeeeeeeeeeEEEEE    . coda vuota

CPU            AAABBBBBCCCDDDDDDDDDEEEEE
coda           ..b.ccccdddeeeeeeeee.....
               ......ddeee.............
```

processo	tempo di risposta	tempo di attesa	tempo di *turn-around*
A	0	0	$0+3=3$
B	1	1	$1+5=6$
C	4	4	$4+3=7$
D	5	5	$5+9=14$
E	12	12	$12+5=17$
media	4,40	4,40	9,40

- *Round Robin*, caso 1 (quanto = 3, senza priorità)

```
processo A     AAA                              LEGENDA DEI SIMBOLI
processo B     --bBBB|bbbBB                     - non ancora arrivato
processo C     ----cc|CCC                       x (minuscolo) attesa
processo D     ------|dddddDDDdddDDDddDDD       X (maiuscolo) esecuzione
processo E     ------|--eeeeeeEEEeeeEE          . coda vuota
                     * D arriva in coda simultaneamente a B
                     * ma, in assenza di criteri di priorita',
                     * la precedenza in accodamento deve andare a B!
```

```
                |
CPU             AAABBB|CCCBBDDDEEEDDDEEDDD
coda            ..b.cc|bbbddeeedddeee.....
                ......|dddee.............
                ......|..e...............
```

processo	tempo di risposta	tempo di attesa	tempo di *turn-around*
A	0	0	$0+3=3$
B	1	$1+3=4$	$4+5=9$
C	2	2	$2+3=5$
D	5	$5+3+2=10$	$10+9=19$
E	6	$6+3=9$	$9+5=14$
media	2,80	5,00	10,00

- *Round Robin*, caso 2 (quanto = 3, con priorità, ma senza prerilascio)

```
processo A      AAA                             LEGENDA DEI SIMBOLI
processo B      --bBBBBB                        - non ancora arrivato
processo C      ----cccc|CCC                    x (minuscolo) attesa
processo D      ------dd|ddddddddDDDDDDDDD      X (maiuscolo) esecuzione
processo E      --------|eeeEEEEE               . coda vuota
                        * l'accodamento di E e D viene determinato
                        * dai rispettivi valori di priorita'
CPU             AAABBBBB|CCCEEEEEDDDDDDDDD
coda            ..b.cccc|eeeddddd.........
                ......dd|ddd.............
```

processo	tempo di risposta	tempo di attesa	tempo di *turn-around*
A	0	0	$0+3=3$
B	1	1	$1+5=6$
C	4	4	$4+3=7$
D	10	10	$10+9=19$
E	3	3	$3+5=8$
media	3,60	3,60	8,60

- *Round Robin*, caso 3 (quanto = 3, con priorità e con prerilascio). Si noti che, non essendovi processi ad uguale priorità, la divisione di tempo non influenzerà l'ordinamento!

```
processo A      AAaaaaaaaaaaaaA                 LEGENDA DEI SIMBOLI
processo B      --BBBBB                         - non ancora arrivato
processo C      ----cccCccccccCC                x (minuscolo) attesa
processo D      ------ddddddddddDDDDDDDDD       X (maiuscolo) esecuzione
processo E      --------EEEEE                   . coda vuota

CPU             AABBBBBCEEEEECCADDDDDDDDD
coda            ..aacccacccccaad.........
                ....aaadaaaaadd..........
                ......d.ddddd............
```

processo	tempo di risposta	tempo di attesa	tempo di *turn-around*
A	0	13	$13+3=16$
B	0	0	$0+5=5$
C	3	$3+5=8$	$8+3=11$
D	10	10	$10+9=19$
E	0	0	$0+5=5$
media	2,60	6,20	11,20

- SJF (*Shortest Job First*), caso 1 (senza prerilascio)

```
processo A      AAA                             LEGENDA DEI SIMBOLI
processo B      --bBBBBB                        - non ancora arrivato
processo C      ----ccccCCC                     x (minuscolo) attesa
processo D      ------ddddddddddDDDDDDDDD       X (maiuscolo) esecuzione
processo E      --------eeeEEEEE                . coda vuota

CPU             AAABBBBBCCCEEEEEDDDDDDDDD
coda            ..b.ccccddddddddd.........
                ......ddeee..............
```

processo	tempo di risposta	tempo di attesa	tempo di *turn-around*
A	0	0	$0+3=3$
B	1	1	$1+5=6$
C	4	4	$4+3=7$
D	10	10	$10+9=19$
E	3	3	$3+5=8$
media	3,60	3,60	8,60

- SJF (*Shortest Job First*), caso 2 (con prerilascio)

```
processo A      AAA                             LEGENDA DEI SIMBOLI
processo B      --bBbbbBBBB                     - non ancora arrivato
processo C      ----CCC                         x (minuscolo) attesa
processo D      ------ddddddddddDDDDDDDDD       X (maiuscolo) esecuzione
processo E      --------eeeEEEEE                . coda vuota

CPU             AAABCCCBBBBEEEEEDDDDDDDDD
coda            ..b.bbbdeeeeddddd.........
                ......d.ddd..............
```

processo	tempo di risposta	tempo di attesa	tempo di *turn-around*
A	0	0	$0+3=3$
B	1	$1+3=4$	$4+5=9$
C	0	0	$0+3=3$
D	10	10	$10+9=19$
E	3	3	$3+5=8$
media	2,80	3,40	8,40

Soluzione 3 (punti 4). Ricapitoliamo il modo in cui opera il "*buddy algorithm*":

1. ciascuna quantità di pagine richiesta viene arrotondata alla più piccola potenza di 2 maggiore di essa
2. si identifica la più piccola zona libera di memoria di indirizzo di base più basso e di ampiezza sufficiente e la si divide in due, ripetendo l'operazione su una delle due metà fino a raggiungere una zona di dimensione pari all'ampiezza necessaria
3. ad ogni rilascio, il blocco rilasciato viene accorpato con ogni eventuale blocco libero adiacente per formare un unico blocco libero.

La figura 4 mostra la progressione d'effetti da esso prodotto a seguito della sequenza di eventi specificati nel quesito, a partire dalla situazione iniziale nota.
Il principale difetto del "*buddy algorithm*" è chiaramente quello causato dal suo passo 1, nel quale l'arrotondamento di ampiezza alla più prossima potenza di 2 produce indesiderabile frammentazione interna. Linux affronta il problema modificando il passo 2 dell'algoritmo, assegnando solo la quantità di memoria effettivamente richiesta e prevedendo una gestione specifica per i frammenti liberi in tali modo determinati.

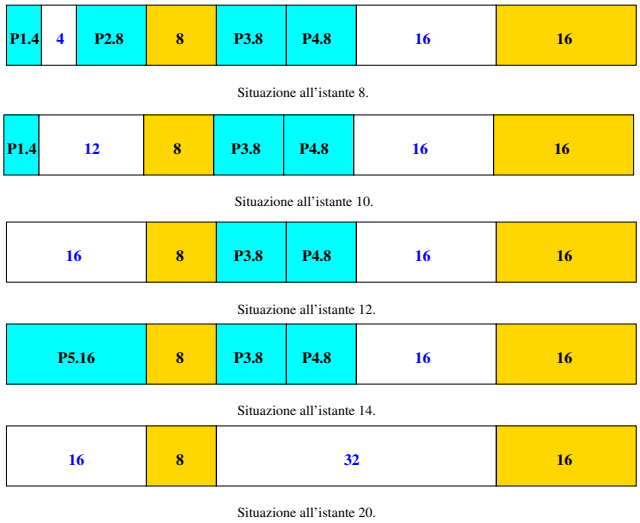

Figura 4: Evoluzione dello stato della memoria in esame.

Soluzione 4 (punti 5). La figura 5 riporta la versione grafica della rappresentazione insiemistica data, al cui interno si distinguono 4 percorsi chiusi.

percorso 1: $\{P_3 \to R_4 \to P_6 \to R_2 \to P_3\}$
percorso 2: $\{P_2 \to R_3 \to P_5 \to R_2 \to P_2\}$
percorso 3: $\{P_2 \to R_4 \to P_6 \to R_2 \to P_2\}$
percorso 4: $\{P_1 \to R_3 \to P_5 \to R_2 \to P_2 \to R_1 \to P_1\}$

All'interno di tutti tali percorsi è presente almeno una risorsa a molteplicità maggiore di uno, per cui non si può affermare nulla a priori, ma occorre analizzare il caso specifico.
Tutti i percorsi condividono la risorsa multipla R_2, mentre i percorsi 2 e 4 condividono anche la risorsa multipla R_3. Ciò comporta per tutti i percorsi la stessa caratteristica: o sono tutti bloccati o tutti si potranno liberare successivamente.
Una molteplicità della risorsa R_3 è assegnata al processo P_4 che non è sospeso: esso può quindi avanzare regolarmente, fino a rilasciare la sua istanza di R_3. Quando ciò avverrà, P_1 o P_2 accederanno all'istanza appena liberata della risorsa R_3, potendo così riprendere ad avanzare.

Caso 1. Qualora il Sistema Operativo attribuisse l'istanza in questione al processo P_1, sarebbe questi a riprendere, al suo termine rilasciando sia l'istanza di R_3 che la risorsa R_1. Ciò consentirebbe a P_2 di acquisire le due risorse richieste, dovendo però rimanere ancora sospeso in attesa di R_4, ancora occupata. Il sistema entrerebbe pertanto in stato di stallo, ove risulterebbero bloccati i processi P_2, P_3, P_5 e P_6.

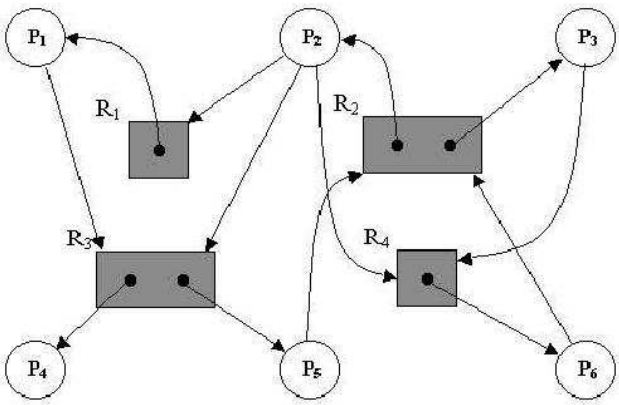

Figura 5: Grafo di allocazione delle risorse per il sistema in esame.

Caso 2. Qualora invece il Sistema Operativo attribuisse l'istanza rilasciata di R_3 al processo P_2, questi non potrebbe comunque riprendere ad avanzare, lasciando quindi il sistema in stato di stallo, i processi bloccati essendo questa volta P_1, P_2, P_3, P_5 e P_6.

L'eventuale richiesta del processo P_4 di accedere alla risorsa R_3 comporterà semplicemente il blocco definitivo del richiedente.

Soluzione 5 (punti 7). Notiamo, per prima cosa, che, senza assegnazione CIDR, il blocco di indirizzi liberi indicati dal quesito apparterrebbe alla classe C, consentendo pertanto solo reti con ≤ 256 indirizzi, non rispondenti alle richieste da soddisfare. Come sappiamo, l'assegnazione CIDR opera con una logica molto simile a quella del *buddy algorithm*: lo spazio contiguo di indirizzi liberi viene suddiviso in blocchi ampi quanto la più piccola potenza di 2 maggiore della richiesta in esame ed attribuisce ad essa il 1^o blocco disponibile. Tale tecnica ha il considerevole difetto di causare sia frammentazione interna, arrotondando ciascuna richiesta in eccesso, che frammentazione esterna, assegnando alla richiesta il 1^o blocco libero tra quelli di uguale dimensione in cui viene nozionalmente suddivisa l'intera zona.
La tabella 1 mostra il modo in cui la tecnica CIDR ripartisce l'insieme contiguo di indirizzi disponibili per soddisfare le richieste. Alla prima ed alla seconda richiesta (di unive2 ed univr2) vengono assegnate zone contigue poichè il primo blocco è di ampiezza multipla del secondo. Alla terza richiesta (di unipd2) viene invece assegnato il 1^o blocco libero (di 3 disponibili) di ampiezza $2^{12} > 3.700$, creando così una zona libera di ampiezza $2^{12} - (2^{11} + 2^{10}) = 2^{10}$ tra il blocco di univr2 e quello di unipd2.

richiedente	**quantità richiesta**	**quantità assegnata**	**da indirizzo**	**ad indirizzo**	*subnet mask*
unive2	1.800	$2^{N_1=11} = 2.048$	194.24.0.0	194.24.7.255	$/21(= 32 - N_1)$
univr2	1.000	$2^{N_2=10} = 1.024$	194.24.8.0	194.24.11.255	$/22(= 32 - N_2)$
liberi		1.024	194.24.12.0	194.24.15.255	
unipd2	3.700	$2^{N_3=12} = 4.096$	194.24.16.0	194.24.31.255	$/20(= 32 - N_3)$
liberi		$2^{14} - 2 \times 2^{12} = 2^{13} = 8.192$	194.24.32.0	194.24.63.255	

Tabella 1: Trattamento CIDR delle richieste in esame.

Un criterio che consente a *router* esterni a e remoti da queste 3 nuove reti di semplificare le proprie tabelle di instradamento è quello di aggregarne gli indirizzi in un unico indirizzo di rete fissato alla base del gruppo (194.24.0.0) e con *subnet mask* determinata dall'ampiezza di indirizzi coperta fino all'estremo superiore di esso (unive2 + univr2 +1.024+ unipd2 $= 8.192 = 2^{N=13}$), ossia $/19(= 32 - N)$. In questo modo, tutti i *router* tranne quello (o quelli) di ingresso nella zona contenente le 3 nuove reti, e dunque un numero cospicuo di loro, userebbero 1 sola nuova voce in tabella invece di 3.

Soluzione 6 (punti 6). Il *router* aziendale separa le sottoreti interne isolandone i domini di diffusione. Ai fini del calcolo di flusso massimo, possiamo pertanto analizzare il traffico separatamente per ogni sottorete. Lo schema logico è composto da 6 sottoreti interne, come mostrato in figura 6 utilizzando la simbologia mostrata in tabella 2, in cui possiamo facilmente individuare i flussi utili derivanti dalle caratteristiche dell'architettura di rete descritta nel quesito. Di seguito discuteremo ogni sottorete interna singolarmente.

flusso dati	**proveniente da**	**sottorete**
X	nodi Hi, $i \in \{1..10\}$	LAN 4, reparto *amministrazione*
Y	nodi Hj, $j \in \{11..20\}$	LAN 5, reparto *produzione*

Tabella 2: Tipologie di flussi presenti nella rete aziendale.

LAN0 e LAN1:
Il loro traffico dipende solo dagli utenti che navigano sul sito *Web*, e quindi sono limitati solo dal traffico esterno.

LAN4:
Essa fa a capo alla porta E1 del *router 2*, racchiude al suo interno il reparto *amministrazione*, ed è gestita globalmente da uno *switch* che segmenta totalmente i vari utenti, così preservando la banda disponibile ad ogni sua porta. Possiamo quindi imporre le seguenti condizioni: $X \leq B = 100$ Mbps, per i rami diretti verso i nodi Hi, $i \in \{1..10\}$, e $10X \leq B$ per il ramo che si collega al *router 2*, dalla più restrittiva delle quali deriviamo: $X = \frac{B}{10} = 10$ Mbps.

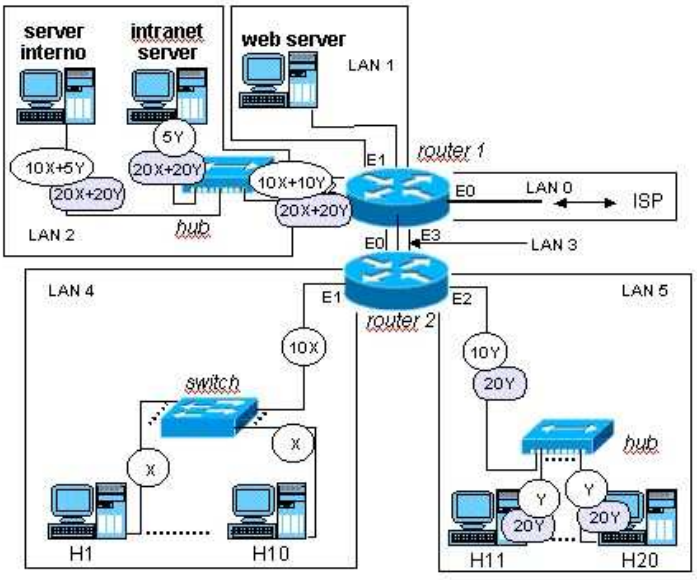

Figura 6: Ripartizione del traffico in ciascun ramo della rete aziendale.

LAN5:
Essa fa a capo alla porta E2 del *router 2*, racchiude al suo interno il reparto *produzione*, ed è gestita globalmente da un *hub*, che per sua natura condivide la banda tra i vari utenti, riflette su ciascuna delle sue 11 porte la somma di tutti i suoi flussi utili. Pertanto, su ogni ramo dell'*hub* vale la seguente condizione: $Y + Y ... + Y + 10Y = 20Y \leq B$ Mbps, che massimizziamo con: $Y = \frac{B}{20} = 5$ Mbps.
Sappiamo dal quesito che ogni nodo di questa sottorete dirige il 50% del proprio traffico verso il server interno ed il rimanente 50% verso il server intranet. Pertanto, nel caso peggiore, ogni nodo Hj, $j \in \{11..20\}$ disporrà di un traffico utile pari a $2,5$ Mbps per ciascuna di questi due utilizzi.

LAN3:
Stanti i valori di X ed Y appena calcolati, il traffico di questa rete, regolato direttamente dai due *router* dovrebbe ammontare a: $10X + 10Y = 10 \times 10 + 10 \times 5 = 150 > B$ Mbps, chiaramente al di là della capacità massima della tecnologia utilizzata dalla rete aziendale. Dovremmo quindi rivedere le espressioni precedenti ed imporre a sistema anche il vincolo appena derivato, utilizzando il fatto che l'architettura interna delle 2 sottoreti facenti capo alle porte E1 ed E2 implica necessariamente che $X = 2Y$ (ciò mostrando il vantaggio prestazionale dovuto allo *switch* rispetto al corrispondente *hub*). Otteniamo dunque: $10X + 10Y = 10 \times 2Y + 10Y = 30Y \leq B$, da cui $Y \leq \frac{B}{30}$, che quindi massimizziamo con $Y = 3,3$ ed $X = 6,6$ Mbps.

LAN2:
Il traffico su questo ramo è determinato dall'*hub* connesso ai due server. Per le considerazioni già fatte a proposito degli *hub*, possiamo allora esprimere la seguente condizione iniziale, imponendola su tutti i rami dell'*hub* in questione: $10X + 10Y + 5Y + 10X + 5Y = 20X + 20Y \leq B$. Sostituendo in essa i valori calcolati di X ed Y precedentemente calcolati però otteniamo: $20 \times 6,6 + 20 \times 3,3 = 200 > B$ Mbps, esattamente il doppio della banda disponibile. Ancora una volta dobbiamo dunque rimodulare i valori di traffico imponendo anche questa condizione. Sfruttando la relazione $X = 2Y$, che resta ovviamente valida, otteniamo: $20X + 20Y = 20 \times 2Y + 20Y = 60Y \leq B$, che massimizziamo imponendo $Y = \frac{B}{60}$, da cui: $Y = 1,6$ ed $X = 3,3$ Mbps.
In sintesi, abbiamo determinato che ogni utente del reparto *amministrazione* disporrà nel caso peggiore di un valore massimo di traffico utile, diretto verso il server interno, pari a $X = 3,3$ Mbps. Ogni utente del reparto *produzione* disporrà di $Y/2 = 0,8$ Mbps per il traffico diretto verso il server interno ed altrettanto per quello diretto verso il server intranet.

Attribuzione di indirizzi IP:
L'indirizzo acquistato dall'Azienda è un dato di fatto ininfluente rispetto alla pianificazione degli indirizzi IP della rete aziendale, fatta salva naturalmente la configurazione della porta E0 del *router*. Per gli altri dispositivi abbiamo a disposizione un'intera classe C di indirizzi riservati 192.168.129.X.

Il dato di progetto da cui partire per l'individuazione del *subnetting* appropriato è che tutte le sottoreti abbiano le stesse dimensioni e la stessa *subnet mask*.
Come sappiamo, il numero S di sottoreti utili rappresentabili con una maschera di *subnet* ampia N *bit* è dato dall'equazione $S = 2^N - 2$. Dovendo noi creare 5 sottoreti, imporremo $S = 5 = 2^n - 2$, ottenendo $n = log_2(S+2) = log_2 7 = 2,8$, e, conseguentemente $N = \lceil n \rceil = 3$, fissato al più piccolo valore intero maggiore di n.
Effettueremo pertanto il *subnetting* richiesto partizionando la parte di nodo dell'indirizzo di classe C dato, ampia 8 *bit*, in una parte *subnet* ampia 3 *bit* ed una parte di nodo interna ridotta ai rimanenti 5 *bit*. Con tali 5 *bit* possiamo rappresentare $2^5 - 2 = 30$ indirizzi IP utili per sottorete, ampiamente sufficienti per risolvere il problema proposto.
Le prime 5 sottoreti utili rappresentabili con il *subnetting* appena descritto avranno pertanto le caratteristiche illustrate in tabella 3.

parte di rete	*subnetting*	parte di nodo		
11000000.10101000.10000001.	001	0.0000	192.168.129.32	indirizzo di 1^a sottorete
11000000.10101000.10000001.	001	0.0001	192.168.129.33	1^o indirizzo di nodo in 1^a sottorete
	...	...	...	
11000000.10101000.10000001.	001	1.1110	192.168.129.62	30^o indirizzo di nodo in 1^a sottorete
11000000.10101000.10000001.	001	1.1111	192.168.129.63	diffusione in 1^a sottorete
11000000.10101000.10000001.	010	0.0000	192.168.129.64	indirizzo di 2^a sottorete
11000000.10101000.10000001.	010	0.0001	192.168.129.65	1^o indirizzo di nodo in 2^a sottorete
	...	...	...	
11000000.10101000.10000001.	010	1.1110	192.168.129.94	30^o indirizzo di nodo in 2^a sottorete
11000000.10101000.10000001.	010	1.1111	192.168.129.95	diffusione in 2^a sottorete
11000000.10101000.10000001.	011	0.0000	192.168.129.96	indirizzo di 3^a sottorete
11000000.10101000.10000001.	011	0.0001	192.168.129.97	1^o indirizzo di nodo in 3^a sottorete
	...	...	...	
11000000.10101000.10000001.	011	1.1110	192.168.129.126	30^o indirizzo di nodo in 3^a sottorete
11000000.10101000.10000001.	011	1.1111	192.168.129.127	diffusione in 3^a sottorete
11000000.10101000.10000001.	100	0.0000	192.168.129.128	indirizzo di 4^a sottorete
11000000.10101000.10000001.	100	0.0001	192.168.129.129	1^o indirizzo di nodo in 4^a sottorete
	...	...	...	
11000000.10101000.10000001.	100	1.1110	192.168.129.158	30^o indirizzo di nodo in 4^a sottorete
11000000.10101000.10000001.	100	1.1111	192.168.129.159	diffusione in 4^a sottorete
11000000.10101000.10000001.	101	0.0000	192.168.129.160	indirizzo di 5^a sottorete
11000000.10101000.10000001.	101	0.0001	192.168.129.161	1^o indirizzo di nodo in 5^a sottorete
	...	...	...	
11000000.10101000.10000001.	101	1.1110	192.168.129.190	30^o indirizzo di nodo in 5^a sottorete
11000000.10101000.10000001.	101	1.1111	192.168.129.191	diffusione in 5^a sottorete

Tabella 3: Ripartizione degli indirizzi nelle prime 5 sottoreti interne disponibili.

Da questa ripartizione possiamo determinare una possibile attribuzione di indirizzi IP alle varie postazioni della rete aziendale, riportata in tabella 4.

rete	**dispositivo**	***IP address***	***subnet mask***	
LAN 5	H11	192.168.129.161	255.255.255.224 (/27)	
	H12	192.168.129.162	/27	
	...			
	H19	192.168.129.169	/27	
	H20	192.168.129.170	/27	
	Router 2	192.168.129.190	/27	porta E2 (*router*-LAN5)
LAN 4	H1	192.168.129.129	/27	
	H2	192.168.129.130	/27	
	...			
	H9	192.168.129.137	/27	
	H10	192.168.129.138	/27	
	Router 2	192.168.129.158	/27	porta E1 (*router*-LAN4)
LAN 3	*Router 2*	192.168.129.125	/27	porta E0 (LAN3-*router 1*)
	Router 1	192.168.129.126	/27	porta E3 (*router 1*-LAN3)
LAN 2	intranet server	192.168.129.92	/27	porta server intranet
	server interno	192.168.129.93	/27	porta server interno
	Router 1	192.168.129.94	/27	porta E2 (*router*-LAN2)
LAN 1	server web	192.168.129.61	/27	porta server web
	Router 1	192.168.129.62	/27	porta E1 (*router*-LAN1)
LAN 0	*Router 1*	dati forniti dall'ISP		porta E0 (*router*-ISP)

Tabella 4: Una possibile attribuzione di indirizzi IP interni alle postazioni della rete aziendale.