

**1. Servizi Multimediali e
Qualità del Servizio (QoS) su IP**
1.1 Introduzione alle QoS su IP

Prof. Raffaele Bolla



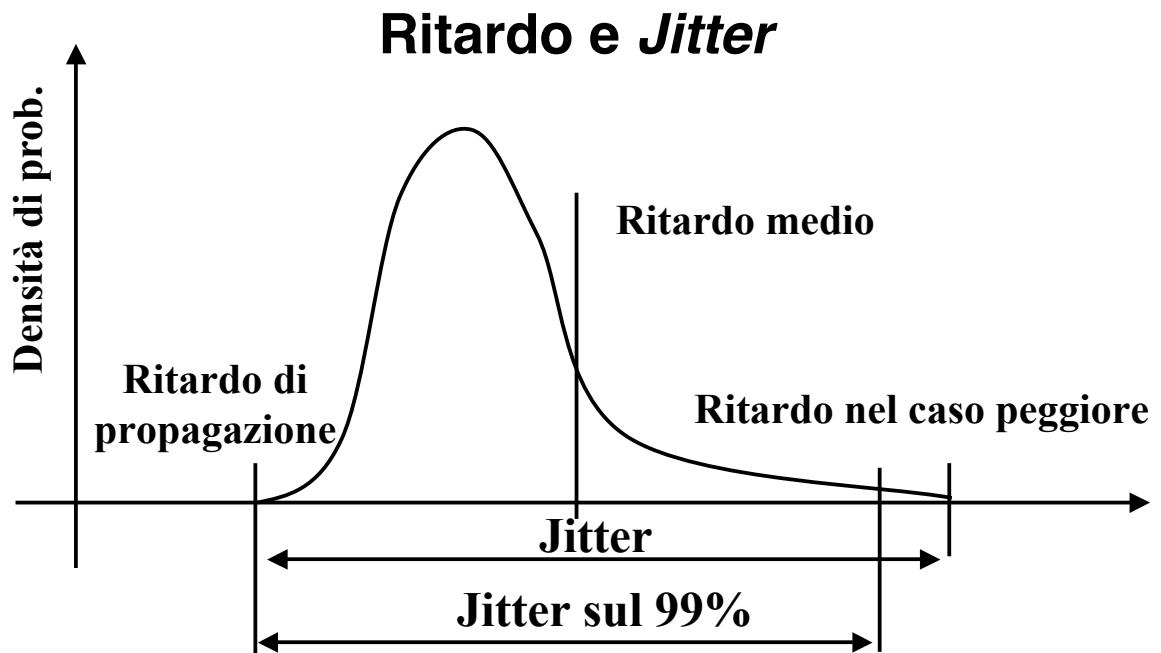
R. Bolla Telematica 2, n. o.

Qualità del Servizio (QoS)

- Due possibili QoS:
 - Misurata sul traffico generato dal servizio (traffico utente)
 - » Si valuta sulla base di indici di prestazione
 - A livello di pacchetto o bit
 - *Troughput*, banda
 - **Ritardo (medio, massimo, jitter)**
 - Probabilità di perdita
 - Probabilità di errore
 - A livello di connessione o flusso
 - Probabilità di blocco
 - Percepita dall'utente del servizio (P-QoS)
 - » Qualità percepita dall'utente
 - *Mean Opinion Score* (MOS)

Spesso considerate
insieme come
perdita di pacchetto

Qualità del Servizio (QoS)



Nessun recupero d'errore, misurata su i pacchetti correttamente ricevuti.

Qualità del Servizio (QoS)

- Il mantenere la P-QoS almeno ad un livello di accettabilità per ciascun servizio dovrebbe essere il vero obiettivo della rete
- Nella pratica i meccanismi di controllo della rete (quando presenti) operano sulla base di indici di prestazioni relativi alla QoS misurata sul traffico utente.
- L'idea generale è quella di cercare di realizzare una tabella di corrispondenze (*mapping*) fra i due insiemi di indici di prestazione
 - Ad esempio per ottenere una MOS di 4 (buona qualità) su un traffico voce si determina che la probabilità di perdita/errore debba essere inferiore a 10^{-4} , il ritardo massimo minore di 20 ms e il jitter minore di 5 ms.

Reti per servizi multimediali

Caratteristiche fondamentali

- Tipicamente richiedono il rispetto di un qualche vincolo sul ritardo (massimo, *jitter*, medio)
- Ma sono “tolleranti” alla perdita (perdite poco frequenti causano “solo” una leggera perdita di qualità)
- I requisiti, in questo senso, sono esattamente opposti a quelli richiesti dai servizi dati.
- I servizi multimediali sono spesso indicati anche col nome di “*continuous media*”

Lezione 1.1, v. 1.0

Classi di applicazioni

continuous media:

- ***Streaming stored audio and video*** (video diffusivo non in tempo reale)
- ***Streaming live audio and video*** (video diffusivo in tempo reale)
- ***Real-time interactive audio and video*** (video interattivo in tempo reale)

5

Reti per servizi multimediali

Streaming stored audio and video

- Un *client* richiede un file audio/video ad un server e concatena/parallelizza la ricezione dalla rete con la visualizzazione.
- E' interattivo: l'utente può eseguire delle operazioni di controllo (operando in modo simile ad un video registratore : *pause, resume, fast forward, rewind, etc.*)
- Ritardo: dalla richiesta del *Client* alla partenza della visualizzazione possono passare da 1 a 10 s.

Lezione 1.1, v. 1.0

Streaming live audio and video

- Molto simile alle TV e radio attuali ma realizzato tramite internet.
- Non interattivo, solo “guarda e ascolta”.

Interactive Real-Time :

- Conferenze audio/video.
- Vincoli sul ritardo più stringenti a causa dell'interazione in tempo reale.
- Video: accettabile < 150 msec
- Audio: buono < 150 msec, accettabile < 400 msec⁶

Il supporto di IP

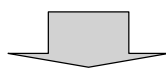
- TCP/UDP/IP suite fornisce un servizio *best-effort* senza garanzie sul ritardo o sulla variabilità del ritardo.
 - Le applicazioni *streaming* (con un ritardo iniziale di 5-10 sec.) sono oggi relativamente comuni, ma le prestazioni si deteriorano in modo inaccettabile quando le linee tendono a congestionarsi.
 - Le applicazioni in tempo reale interattive hanno dei requisiti più stringenti sia sul ritardo che sul *jitter*, requisiti che non si riescono a rispettare in modo affidabile usando il TCP/IP.

Lezione 1.1, v. 1.0

7

Perchè IP Reti integrate nei servizi

- Uno degli obiettivi delle reti di telecomunicazioni nell'ultimo decennio è stato l'integrazione dei servizi: creare una rete in grado di supportare ogni tipo di servizio (previsto e prevedibile)
- Un rete integrata permetterebbe una elevata efficienza e renderebbe appetibile l'offerta dinamica di nuovi servizi di tlc ma richiede una tecnologia molto flessibile



Commutazione di pacchetto

Lezione 1.1, v. 1.0

8

Perchè IP IP e ATM

- Inizialmente gli standard (ITU-T) e la ricerca hanno proposto ATM (Asynchronous Transfer Mode)
 - Commutazione pacchetto veloce
 - Orientato alla connessione
 - Pensato principalmente per servizi *real-time* (video-voce)
 - ATM non si trova molto "a suo agio" con il traffico dati, al contrario di IP che è invece stato pensato solo per i dati.
- Con Internet, il TCP/IP si è imposto sia come tecnologia che come "infrastruttura" di rete, quindi è sembrato "naturale" proporre **Internet** anche come **rete integrata universale**

Linee evolutive di Internet verso il supporto delle comunicazioni multimediali (QoS)

- Ci sono due filosofie su come agire per realizzare servizi con QoS in IP
 - Nessun intervento architetturale, si lascia IP così come è, sovradimensionando la banda trasmissiva;
 - Introduzione di nuove architetture, elementi di rete e protocolli specifici per la QoS:
 - » Architetture InteServ e DiffServ
 - » MPLS

Linee evolutive: nessun intervento

- In questo caso si parte dal presupposto che la banda è, almeno potenzialmente, una risorsa disponibile in abbondanza (Fibre ottiche, WDM)
 - Quindi non si opera nessuna distinzione di classe e nessuna prenotazione di banda.
 - Quando la domanda sale, si fornisce più banda in generale (aumentare le capacità delle linee o il loro numero), quindi agire in sede di pianificazione (*Planning*).
 - Si cerca di localizzare i contenuti sul bordo della rete (*Cache e server* nella rete d'accesso degli ISP) .

Linee evolutive: nessun intervento

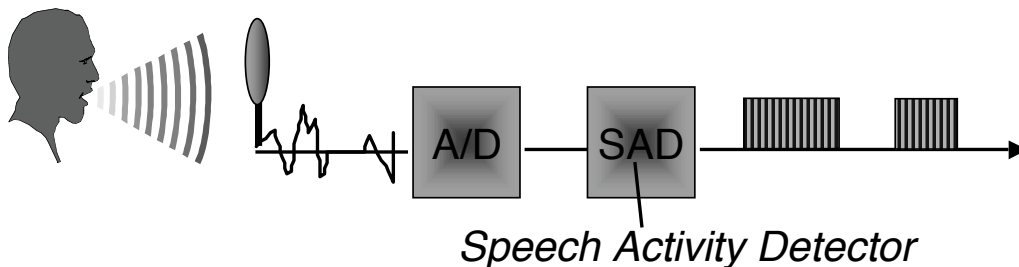
- L'idea di base è che:
 - Se si riesce a realizzare un “*over-provisioning*” sistematico nella rete allora
 - » nessuna linea o nodo sarà mai in condizione di congestione, anche solo accennata
 - » e quindi tutti i parametri di QoS saranno rispettati per tutti i pacchetti
- Questo modello non appare ad oggi realmente applicabile nella sua forma estrema, ma se si fa riferimento ad una sua versione meno estrema può essere considerato valido.

Linee evolutive: nessun intervento

- Pur senza introdurre elementi nuovi, si possono comunque attuare alcune azioni per mitigare l'effetto negativo del trattamento "best-effort":
 - Usare UDP evitando la fase di *slow-start* ed il recupero d'errore del TCP.
 - Usare un buffer ed un ritardo iniziale di visualizzazione a destinazione per compensare il *jitter*.
 - Introdurre dei *timestamp* così da permettere la riproduzione temporale corretta dei dati.
 - Adattare la compressione alla banda disponibile.
 - Inviare pacchetti ridondanti per mitigare le perdite.

Multimedia su *best-effort*: Un esempio di servizio voce su IP

- Una sorgente vocale può essere modellata come un sistema a due stati: *parlo* e *silenzio*.



- Durata media del periodo attività (*talkspurt*) = 1.00 s
- Durata media del periodo di silenzio (*gap*) = 1.35 s
- Percentuale di attività ≈ 42 %

Multimedia su *best-effort*: Un esempio di servizio voce su IP

- Nel caso in esempio, supponiamo che quando la sorgente è nello stato *parlo* l'applicazione generi dati con tasso pari a 64 Kb/s (8000 campioni/s, ognuno di 8 bit), nello stato *silenzio* ovviamente non genera dati.
- Questa significa che l'applicazione, ogni 20 ms genera un blocco di 160 Bytes.
- Al blocco aggiunge una intestazione ed il tutto viene inserito in un pacchetto UDP ed inviato in rete.
- Il ricevitore riceve questi pacchetti ma non tutti, perché alcuni vanno persi, e con ritardo variabile (eventualmente anche fuori sequenza).
- Il ricevitore deve decidere quando riprodurre un blocco e cosa fare se mancano dei blocchi.

Multimedia su *best-effort*: un esempio di telefonia su IP

- Sono due i problemi principali che vanno risolti
 - Perdite di pacchetti:
 - » Si potrebbe usare TCP, ma la ritrasmissione introduce ritardo e il controllo di flusso limita il tasso.
 - » Anche usando il UDP, in ogni modo, bisognerebbe inserire numeri di sequenza per accorgersi di eventuali perdite.
 - Ritardo
 - » Il ritardo *end to end* deve essere inferiore a 400 ms, altrimenti ne risente l'interattività.
 - » Il *jitter* in questo caso è dato dal fatto che i pacchetti sono trasmessi a 20 ms uno dall'altro ma possono arrivare al ricevitore sia più vicini che più lontani.
 - » Si dovrebbe comunque inserire dei *timestamp* per capire quando un blocco dati debba essere riprodotto.

Multimedia su *best-effort*: Compensazione del *jitter*

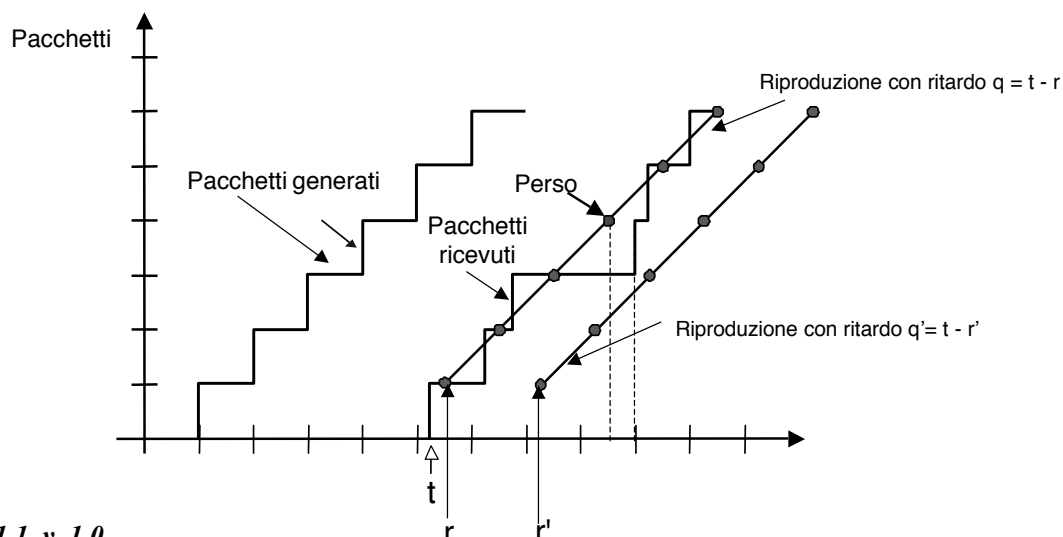
- Il *Jitter* viene in genere compensato utilizzando un buffer ed inserendo un ritardo q di riproduzione, ossia:
 - Se un blocco con un *timestamp* t è ricevuto entro $t + q$, viene riprodotto all'istante $t + q$
 - Altrimenti viene scartato
- Questa strategia non richiede numeri di sequenza per gestire le perdite (in pratica trasforma in perdita tutto quanto arriva con un ritardo superiore a q).
- Il problema è fissare q
 - q lunghi implicano meno perdite (per *jitter*) ma peggiore interazione;
 - q corti: l'interazione è più efficace ma più perdite.

Lezione 1.1, v. 1.0

17

Multimedia su *best-effort*: Compensazione del *jitter*

- La sorgente genera un pacchetto ogni 20 ms.
- Il primo pacchetto è ricevuto all'istante t .
- Nel primo caso (ritardo q) è riprodotto a partire dall'istante r .
- Nel secondo caso (ritardo q') è riprodotto a partire dall'istante r' .



Lezione 1.1, v. 1.0

18

Multimedia su *best-effort*: Compensazione delle perdite

● *Forward Error Correction (FEC)*

- Siccome la perdita significa il non arrivo del pacchetto, le tecniche FEC devono coinvolgere più di un pacchetto.
- Possiamo fare due esempi di possibili strategie:

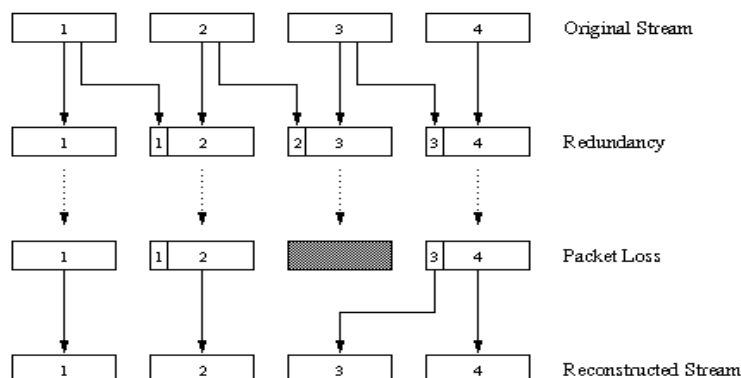
● **Strategia 1**

- Ogni n blocchi, si costruisce un blocco ridondante ottenuto come XOR degli n precedenti e lo si invia in coda al gruppo
- Se perdo un blocco solo degli $n+1$, all'istante di ricezione del blocco $n+1$ lo posso ricostruire
- La banda aumenta di un fattore $1/n$ (ad es. $1/10$ se $n=10$)
- Il ritardo fisso iniziale d deve essere tale da permettere la ricezione di almeno $n+1$ pacchetti (se $n=10$, $d > 200$ msec).
- Quindi aumentando n aumenta l'efficienza ma aumenta anche il ritardo e la probabilità di avere più di una perdita in una sequenza (situazione in cui questo meccanismo diventa inefficace).

Multimedia su *best-effort*: Compensazione delle perdite

Strategia 2

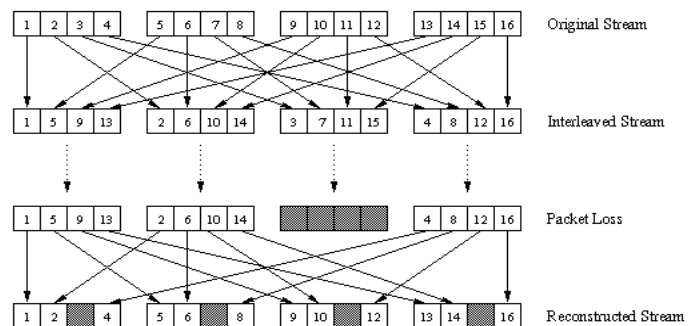
- Si comprime i dati usando due codifiche a risoluzioni diverse: una bassa (B) ed una alta (A) (ad es. un PCM a 64 Kbit/s e un GSM a 13 Kbit/s).
- Si costruisce il pacchetto n prendendo il blocco n del flusso codificato con A (160 Byte) e il blocco $n-1$ del flusso codificato con B (32,5 byte)
- Quando un pacchetto viene perso può essere ricostruito (con minore qualità) all'arrivo de pacchetto successivo
- Si può proteggere anche nei confronti di due perdite consecutive aggiungendo anche il blocco $n-2$ del flusso B.



Multimedia su *best-effort*: Compensazione delle perdite

● *Interleaving*

- I blocchi sono spezzato in unità più piccole (ad esempio in 4 unità da 40 byte).
- Le unità vengono mescolate in pacchetti successivi che quindi contengono parti di più blocchi.
- I blocchi originali vengono ricostruiti al ricevitore.
- Se un pacchetto viene perso la perdita viene distribuita su n blocchi e quindi è più facilmente compensabile.



- Ricostruzione pacchetto/unità persa
 - Si usano dei dati che assomigliano ai precedenti (ripetizione o interpolazione)
 - Funziona abbastanza bene per perdite rare e pacchetti/unità piccole (4-40 msec)

Lezione 1.1, v. 1.0

21

QoS

- Per poter fornire supporto a servizi differenti, bisogna poter assicurare una **Qualità** ossia rispettare dei requisiti diversi in termini
 - Perdita massima ↓
 - Ritardo massimo ↓
 - Jitter ↓
 - Throughput ↑
- In corrispondenza di flussi di traffico differenti per
 - Tasso di generazione
 - Statistiche di generazione (CBR, VBR,...)

Lezione 1.1, v. 1.0

22

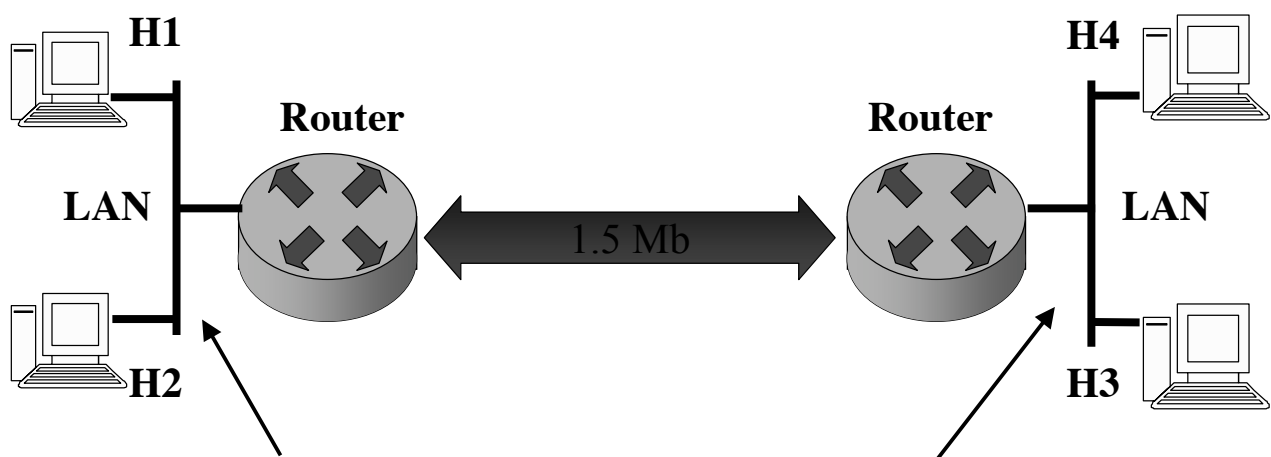
QoS su reti a pacchetto

- Internet o, in generale, l'architettura TCP/IP è nata ed è attualmente strutturata per fornire solo servizi *best-effort*, ossia con
 - Perdita di pacchetto
 - Ritardi non superiormente limitati
 - Ritardi variabili (*Jitter*)
- Per poter utilizzare Internet come rete universale, bisogna poter assicurare un grado di QoS ad applicazioni o utenti che ne facciano richiesta, e quindi bisogna introdotti dei **nuovi componenti architetturali**.

Lezione 1.1, v. 1.0

23

Scenario di riferimento

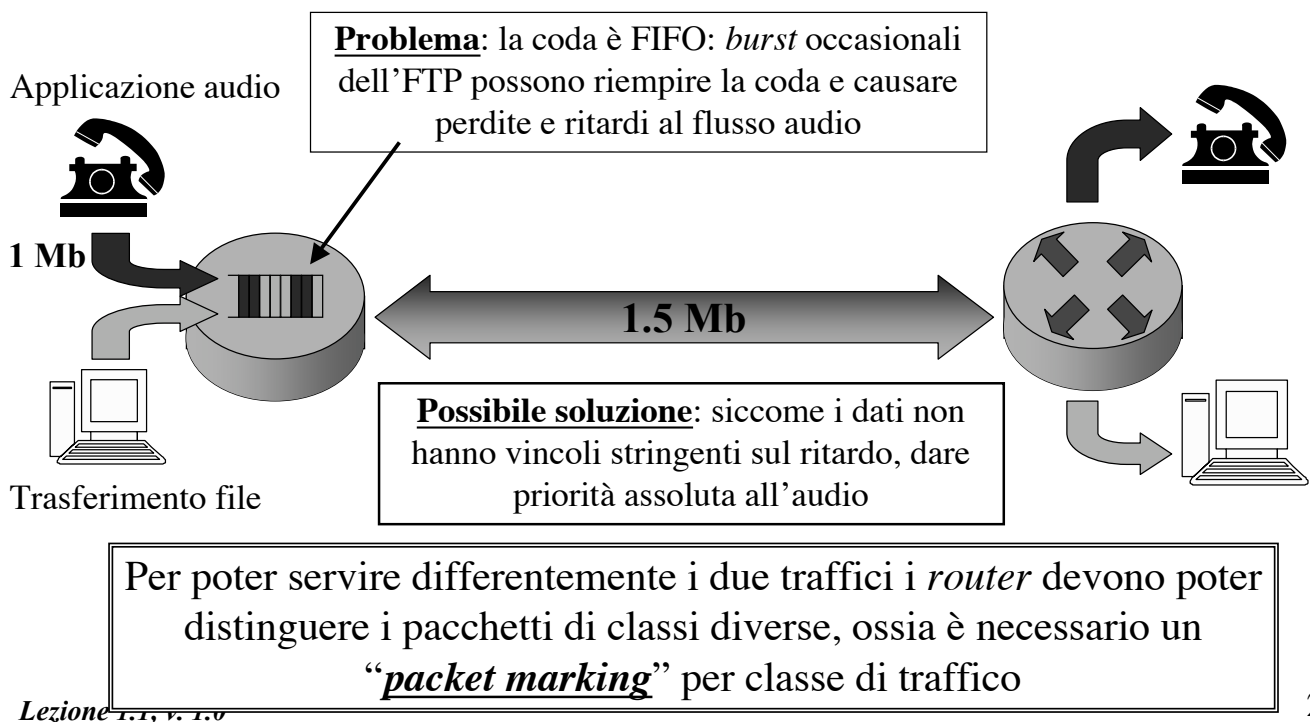


Si supponga che la LAN non sia un elemento critico (nel senso che ha una velocità molto più alta di quella presente fra i due *router*)

Lezione 1.1, v. 1.0

24

Scenario 1



25

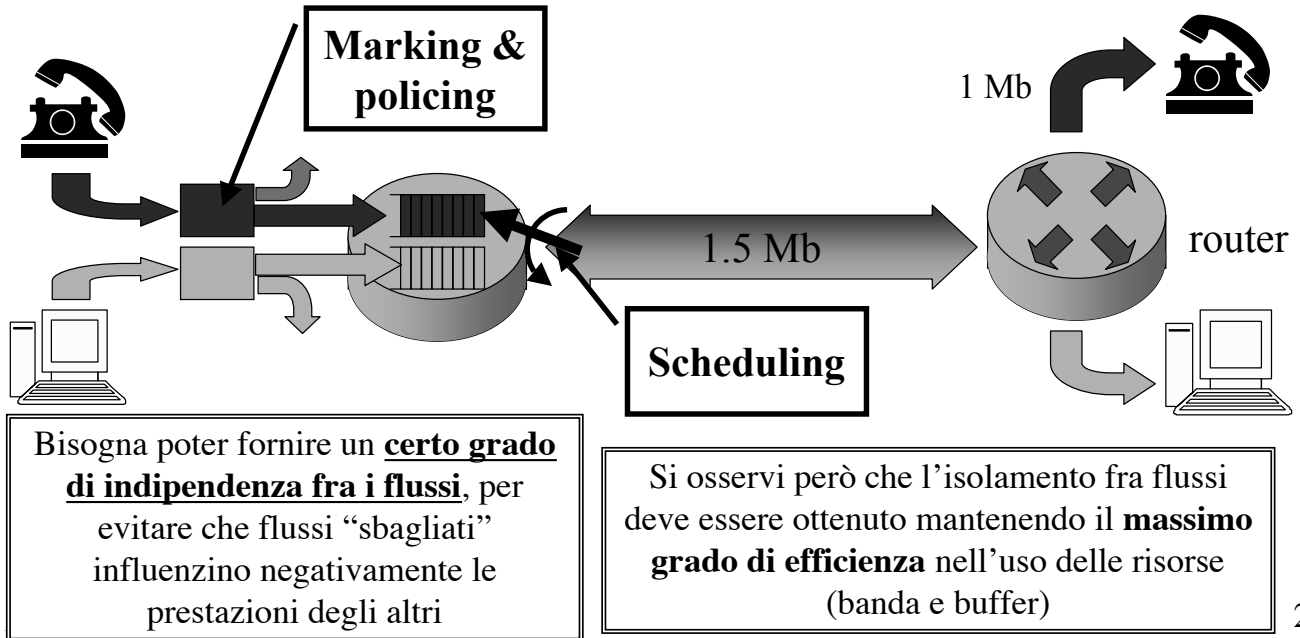
Scenario 1

- Se l'utente del trasferimento file ha pagato molto per avere un servizio di "prima classe", mentre l'utente audio ha pagato poco per avere un servizio a basso prezzo, non è detto che la priorità dell'audio debba essere assoluta.
- Ci sono tre considerazioni da fare:
 - Il *Marking* esplicito serve solo a distinguere i pacchetti ma non determina necessariamente il tipo di servizio che riceverà un pacchetto.
 - Il modo con cui il *router* classifica (*classification*) i pacchetti può basarsi su diversi criteri (sorgente, destinazione, stato della linea) di cui l'identificazione esplicita è solo uno degli elementi
 - Il modo in cui i diversi pacchetti vengono classificati e poi trattati dipende dalla "politica" di servizio.

Non basta fare il marking per assicurare una QoS
Occorrono anche altri meccanismi

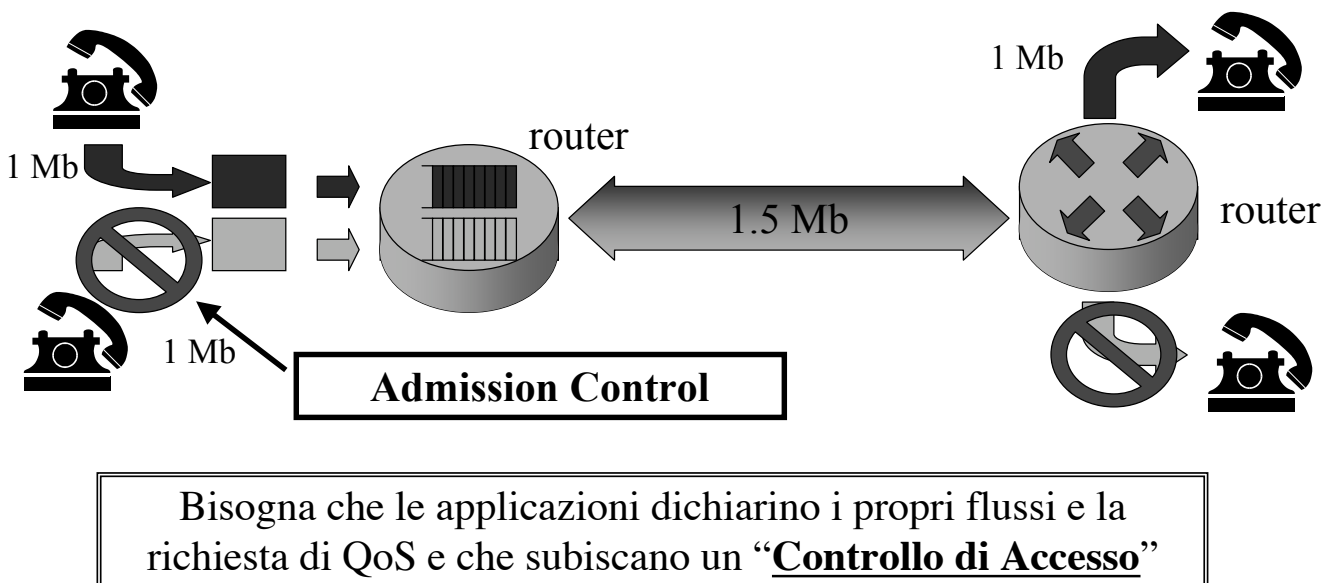
Scenario 2

L'applicazione audio (volontariamente o per errore) comincia a generare più di 1 Mbps; se arriva a generare ≥ 1.5 Mbps, l'FTP non passa più



Scenario 3

Se si attivano due applicazioni audio da 1 Mbps, qualunque *classification* o *policing* o partizione (*scheduling*) non permette di mantenere la QoS



Elementi

- In sostanza gli elementi che devono essere considerati per fornire una QoS su IP sono:
 - **Classificazione dei pacchetti (*Packet Classification*)**
 - **Isolamento dei flussi: *Scheduling e Policing***
 - **Efficace utilizzazione delle risorse (Allocazione dinamica delle risorse)**
 - **Controllo di ammissione (*Call Admission*)**