

ALMA MATER STUDIORUM · UNIVERSITA'  
DI BOLOGNA

**SCUOLA DI SCIENZE**  
**Corso di Laurea in Informatica Per Il Management**

**Un Confronto tra Tecniche MPEG-DASH  
e SVC**

**Relatore:**

Dott. Luca Bedogni

**Presentata da:**

Andrea Elia Tallaros

**Matricola 732017**

Sessione III

Anno Accademico 2017-2018

# Indice

---

<b>Capitolo 1-</b> Dove tutto comincio? .....	4
1.1 Nascita Dello Streaming Dinamico.....	6
1.1.1 Protocolli di Compressione Video in Ambito Streaming.....	6
1.1.2 Terminologie e Standard.....	9
1.2 Soluzioni pre-DASH.....	9
<b>Capitolo 2-</b> La Nuova Era .....	12
2.1 DASH.....	12
2.2 Soluzioni Proprietarie.....	13
2.2.1 Test Effettuati Con I Protocolli Proprietari.....	20
2.3 Algoritmi Adattivi.....	23
2.4 Caratteristiche e Quality of Experience (QoE).....	26
2.4.1 AIS.....	27
<b>Capitolo 3-</b> Protocollo MPEG-DASH .....	30
3.1 MPEG-DASH.....	30
3.2 Come e' Possibile Uno Streaming Efficiente.....	31
3.3 Struttura Media Presentation Description.....	33
3.4 Struttura Representations.....	33
3.5 Elemento SegmentBase.....	35
3.6 Elemento SegmentList.....	36
3.7 Elemento SegmentTemplate.....	36
3.8 Test effettuati con MPEG-DASH.....	39
<b>Capitolo 4-</b> Scalable Video Coding (SVC).....	41
4.1 Introduzione.....	41
4.2 Scalabilita' Temporale.....	46
4.2.1 Valutazione di Analisi.....	49
4.3 Scalabilita' Spaziale.....	55
4.3.1 Valutazione di Analisi.....	57
4.4 Scalabilita' Qualitativa.....	58

4.4.1 Valutazione di Analisi.....	60
<b>Capitolo 5- Algoritmi Usati da SVC.....</b>	<b>64</b>
5.1 Prediction Module.....	64
5.2 Deblocking Filter.....	66
5.3 Pre-processing Filter.....	68
<b>Capitolo 6- Esempi di Configurazione.....</b>	<b>70</b>
6.1 Configurazione Ideale per Streaming Qualitativo.....	70
6.2 Configurazione Ideale per Streaming Performante.....	71
<b>Capitolo 7 –Confronto Tecnologico.....</b>	<b>73</b>
<b>Conclusioni .....</b>	<b>77</b>
<b>Ringraziamenti.....</b>	<b>78</b>
<b>Bibliografia e Riferimenti.....</b>	<b>79</b>

# Capitolo 1

---

## Dove tutto comincio'

Il 5 settembre 1995, ESPN SportsZone ha trasmesso in diretta radiofonica una partita di baseball tra i Seattle Mariners e i New York Yankees a migliaia di suoi abbonati in tutto il mondo, utilizzando una tecnologia all'avanguardia sviluppata da una società di start-up con sede a Seattle denominata Progressive Networks. È stato il primo evento di livestreaming al mondo. Qualche anno dopo l'azienda cambierà il suo nome in RealNetworks e in breve tempo si troverà coinvolta in un'aspra guerra tecnologica e legale con Microsoft per il dominio di un mercato tecnologico nuovo di zecca: quello dello streaming media. Sebbene la prospettiva di streaming su internet avesse sempre entusiasmato figure importanti come i CEO, gli anni dell'infanzia dello streaming sono stati principalmente rovinati da problemi pragmatici come il modo di guardare con successo lo streaming video su linee modem 56k. Microsoft è emersa dalla guerra con RealNetworks come vincente (grazie alle tecnologie Windows Media), ma non è stata in grado di capitalizzare la vittoria. Mentre l'azienda statunitense di Redmond ha perso il suo vantaggio, Macromedia (successivamente acquisita da Adobe Systems) ha lentamente ma inesorabilmente eroso la quota di mercato di Windows Media a metà degli anni 2000 a favore del suo sempre più popolare Flash Player. Flash ha scosso l'industria dei media in streaming sposando per la prima volta l'interattività, il Web 2.0 e i media in streaming. Cose assolutamente importanti, ma i vecchi problemi rimasero ancora: larghezza di banda, scalabilità e portata.

A metà degli anni 2000 la stragrande maggioranza del traffico Internet era basato su HTTP e le reti di distribuzione dei contenuti (CDN) erano sempre più utilizzate per garantire la distribuzione di contenuti popolari a un vasto pubblico. I media trasmessi tramite un miscuglio di protocolli proprietari - tutti basati principalmente sull'UDP, molto meno popolare - si trovarono improvvisamente a faticare a tenere il passo con la

domanda. Nel 2007 una società denominata Move Networks ha introdotto una tecnologia e un servizio che ancora una volta avrebbe cambiato il settore: streaming adattivo basato su HTTP.

Invece di affidarsi a protocolli di streaming proprietari e di lasciare gli utenti in mano a un mercato monopolistico, Move Networks ha utilizzato il protocollo HTTP per fornire media in piccoli blocchi di file mentre utilizzava l'applicazione player per monitorare la velocità di download e richiedere blocchi di qualità variabile (dimensioni) in risposta alle mutevoli condizioni di rete. La tecnologia ha avuto un impatto enorme perché ha permesso di distribuire in lungo e in largo i media in streaming utilizzando CDN (su HTTP standard) e memorizzandoli nella cache per garantire l'efficienza, eliminando allo stesso tempo i fastidiosi problemi di buffering e connettività per i clienti.

In questa proposta si andrà a discutere i risultati di esperimenti che implementano le due tecnologie, il modo con il quale sono stati implementati e come arrivano allo stesso scopo prendendo strade diverse.

## 1.1 Nascita Dello Streaming Dinamico

L'evoluzione della tecnologia digitale e il continuo miglioramento delle infrastrutture di comunicazione sta spingendo un gran numero di applicazioni multimediali interattive, come le videoconferenze in tempo reale, il web streaming e la televisione mobile a rivisitare il modo con il quale i dati vengono trasmessi e con quale efficienza.

Le nuove possibilità hanno creato un esigente mercato di consumatori, che richiede la migliore qualità video ovunque si trovino e qualunque sia il loro supporto di rete. A tal fine, il video trasmesso deve corrispondere alle caratteristiche del ricevitore, quali il **bitrate**, la **risoluzione** e il **frame rate** richiesti, con l'obiettivo di fornire la migliore qualità in funzione dei limiti del ricevitore e della rete. Quindi il problema che si pone è' come riuscire a dinamicizzare il processo di codifica del flusso di dati tenendo sempre presente l'efficienza ed i costi dello stesso.

In questa proposta vengono argomentate due soluzioni che usano algoritmi e metodologie differenti per la codifica del flusso di dati ma hanno come scopo finale la medesima cosa. Nel dettaglio, vedremo come le proposte di MPEG-DASH e di SVC (Scalable Video Coding) realizzano il loro obiettivo di dinamicità ed esamineremo le loro caratteristiche con esempi tratti da applicazioni reali.

### 1.1.1 Protocolli di Compressione Video in Ambito Streaming

**Protocollo UDP (User Datagram Protocol):** è uno dei membri principali della suite di protocolli Internet. Esso è' stato progettato da David P. Reed nel 1980 e definito formalmente in RFC 768. Con l'UDP, le applicazioni informatiche possono inviare messaggi, in questo caso

denominati datagrammi, ad altri host su una rete IP (Internet Protocol). Non sono necessarie comunicazioni preventive per impostare canali di comunicazione o percorsi dati.

UDP utilizza un semplice modello di comunicazione senza connessione, con un meccanismo di protocollo minimo. UDP fornisce un checksum per l'integrità dei dati e numeri di porta per indirizzare diverse funzioni all'origine e alla destinazione del datagramma. Non ha dialoghi di handshake e quindi espone il programma dell'utente a qualsiasi inaffidabilità della rete sottostante; non vi è alcuna garanzia di consegna, ordine o protezione da duplicati. Se sono necessari strumenti di correzione degli errori a livello di interfaccia di rete, un'applicazione può utilizzare il Transmission Control Protocol (TCP) o il Stream Control Transmission Protocol (SCTP) progettati a tale scopo.

UDP è adatto per scopi in cui il controllo e la correzione degli errori non sono necessari o vengono eseguiti nell'applicazione; UDP evita l'overhead di tale elaborazione nello stack di protocolli. Le applicazioni sensibili al fattore tempo utilizzano spesso l'UDP perché la caduta dei pacchetti è preferibile all'attesa di pacchetti ritardati a causa della ritrasmissione, che potrebbe non essere un'opzione in un sistema in tempo reale.

**Transmission Control Protocol (TCP):** è uno dei principali protocolli della suite di protocolli Internet. Esso ha avuto origine nell'implementazione iniziale della rete in cui integrava il protocollo Internet (IP). Pertanto, l'intera suite è comunemente denominata TCP/IP. Il TCP fornisce una consegna affidabile, ordinata e controllata dagli errori di un flusso di ottetti (byte) tra applicazioni in esecuzione su host che comunicano attraverso una rete IP. Le principali applicazioni Internet come il World Wide Web, la posta elettronica, l'amministrazione remota e il trasferimento di file si affidano al TCP.

**RTP:** il Real-time Transport Protocol (RTP) è un protocollo di rete per la trasmissione di audio e video su internet. RTP è utilizzato nei sistemi di comunicazione e di intrattenimento che comportano lo streaming dei

media, come la telefonia, le applicazioni di videoconferenza, tra cui WebRTC, i servizi televisivi e le funzionalità push-to-talk basate sul web.

**RTCP:** è un protocollo strettamente relazionato al Real Time Transport Protocol (RTP). Le sue funzionalità di base e la struttura dei pacchetti sono definite nell'RFC 3550. RTCP fornisce statistiche fuori banda e informazioni di controllo per una sessione RTP. Collabora con RTP nella consegna e nel confezionamento di dati multimediali, ma non trasmette direttamente i dati multimediali. [1]

La funzione principale è quella di fornire un feedback sulla qualità del servizio (QoS) nella distribuzione dei media, inviando periodicamente informazioni e statistiche come il numero di byte e pacchetti trasmessi, la perdita di pacchetti, la variazione del ritardo dei pacchetti e il tempo di ritorno ai partecipanti ad una sessione multimediale in streaming. A differenza dell'HTTP, RTCP risulta stateful proprio perché tiene conto di un identificatore di sessione.

**HTTP**, Hypertext Transfer Protocol: è il protocollo più diffuso per la trasmissione di dati sul web [3]. La logica del HTTP è basata sul concetto di client/server: il client manda una richiesta ed il server ritorna una risposta. Le specifiche del protocollo sono gestite dal World Wide Web Consortium (W3C). Come già menzionato HTTP è stateless. Quando il server ha mandato la risposta (positiva/negativa) al client, termina il canale di comunicazione con esso.

**L'H.264:** o MPEG-4 AVC (Advanced Video Codec) è un formato standard di compressione video digitale con perdita (lossy) creato da Moving Picture Experts Group. Nel maggio 2003 è stata presentata la versione iniziale la quale ha avuto un grande successo. Tuttavia, con l'avanzare della tecnologia e quindi anche delle richieste da parte degli utenti si è riscontrato il problema dello streaming adattivo. Non potendo accontentare questa dinamicità H.264 andava revisionato. Per aggirare questo inconveniente, il concetto di scalabilità per la codifica video è stato recentemente proposto come soluzione emergente per supportare, in una data rete, endpoint con capacità di elaborazione video distinte, con



l'aggiunta della G-extension (*Luglio 2007, progetto totalmente approvato*) per H.264/AVC, diventato poi H.264/SVC, come vedremo in seguito.

## 1.1.2 Terminologie e Standard

Si possono definire Standard le metriche quantificabili alle quali le parti si adoperano per consentire un certo "spazio" per la trasmissione di dati [18]. Gli standard derivano da una varietà di fonti e processi nel corso del tempo. Possono essere suddivisi in due categorie:

- Standard de jure
- Standard de facto

Nel caso dello Standard de jure l'insieme delle norme vengono decise e convalidate da vari enti internazionali preposti, come l'Institute of Electrical and Electronics Engineers o IEEE, senza che vi sia una applicazione obbligatoria.

Uno standard de jure e' per esempio l'Open System Interconnection o il modello ISO-OSI nelle reti di calcolatori.

Uno standard de facto, invece, e' sviluppato da una o piu' aziende e acquisisce importanza grazie alla sua diffusione nel mercato, per esempio la pila di protocolli TCP/IP o il protocollo di comunicazione RTSP.

## 1.2 Soluzioni pre-DASH

In passato per la maggior parte dei servizi di audio e video veniva usato RTSP. Il Real Time Streaming Protocol (RTSP) è un protocollo di controllo di rete progettato per l'uso in sistemi di intrattenimento e comunicazione per controllare i server multimediali in streaming. Il protocollo viene utilizzato per stabilire e controllare le sessioni multimediali tra i punti finali.

I client dei media server emettono comandi in stile VCR, come **play**, **record** e **pause**, per facilitare il controllo in tempo reale dello streaming

multimediale dal server ad esso (Video On Demand) o da un client al server (Voice Recording).

La trasmissione di dati in streaming di per sé non è un compito di RTSP. La maggior parte dei server RTSP utilizza il Real-time Transport Protocol (RTP) insieme al Real-time Control Protocol (RTCP) per la trasmissione dei flussi multimediali. Tuttavia, alcuni fornitori implementano protocolli di trasporto proprietari. Il software RTSP di RealNetworks, ad esempio, utilizzava anche Real Data Transport (RDT), proprietario di RealNetworks.

RTSP è stato sviluppato da RealNetworks, Netscape e Columbia University, con la prima bozza presentata a IETF nel 1996. È stato standardizzato dal Multiparty Multimedia Session Control Working Group (MMUSIC WG) dell'Internet Engineering Task Force (IETF) e pubblicato come RFC 2326 nel 1998.

### **Esempio di chiamata on Demand PLAY**

Una richiesta di PLAY causa la riproduzione di uno o tutti i flussi multimediali. Le richieste di riproduzione possono essere eseguite inviando più richieste di PLAY. L'URL può essere l'URL aggregata (per riprodurre tutti i flussi multimediali), o una singola URL del flusso multimediale (per riprodurre solo quel flusso, caso sottostante). È possibile specificare un intervallo. Se non viene specificato, lo stream viene riprodotto dall'inizio fino alla fine, oppure, se lo stream è in pausa, viene ripreso nel punto in cui è stato messo in pausa.

```
C->S: PLAY rtsp://example.com/media.mp4 RTSP/1.0
      CSeq: 4
      Range: npt=5-20
      Session: 12345678
```

```
S->C: RTSP/1.0 200 OK
      CSeq: 4
      Session: 12345678
      RTP-Info:
url=rtsp://example.com/media.mp4/streamid=0;seq9810092;
rtptime=3450012
```

**Figure 1 Esempio di richiesta VOD**

Sebbene per certi versi simile all'HTTP, RTSP definisce sequenze di controllo utili per monitorare la riproduzione multimediale. Mentre HTTP è senza stato (per definizione), RTSP ha uno stato; un identificatore è usato quando risulta necessario tenere traccia di sessioni simultanee. Come HTTP, RTSP utilizza TCP per mantenere una connessione end-to-end e, mentre la maggior parte dei messaggi di controllo RTSP sono inviati dal client al server, alcuni comandi viaggiano nella direzione opposta.

Come si vedrà anche in seguito, con l'aumentare delle necessità provenienti quotidianamente, un approccio OD (On Demand) non può garantire efficienza nella riproduzione di contenuti multimediali. Non solo è uno spreco di risorse ma i costi che accompagnano il mantenimento di un sistema con la predetta architettura sono elevati. Questo ha come conseguenza la limitazione dell'utilizzo di architetture OD: preso in considerazione il fatto che molti contesti professionali/lavorativi utilizzano tecnologie di trasmissione di dati, una situazione nella quale non si riesce a captare o decodificare con successo un flusso può essere pericolosa (ad esempio conferenza medica). Ecco perché la scoperta delle architetture adattive sono essenziali per la trasmissione di dati.

## Capitolo 2

---

### La Nuova Era

#### 2.1 DASH

Lo streaming adattivo è una tecnica utilizzata per lo streaming multimediale su reti di computer. Mentre in passato la maggior parte delle tecnologie di streaming video o audio utilizzava protocolli di streaming come RTP con RTSP, oggi le tecnologie di streaming adattivo sono quasi esclusivamente basate su HTTP e progettate per funzionare in modo efficiente su grandi reti come Internet. Una delle problematiche che lo streaming video crea e' sicuramente il buffering. Questa problematica risulta particolarmente fastidiosa in quanto, durante la fruizione del contenuto scelto, possono verificarsi una o più pause e quindi avere un impatto negativo sulla QoE (Quality Of Experience). L'adaptive streaming agisce come soluzione al problema rilevando la larghezza di banda e la capacità della CPU di un utente in tempo reale, regolando di conseguenza la qualità del flusso multimediale cercando di fornire la migliore QoE all'utente. Richiede l'uso di un encoder in grado di codificare un singolo supporto sorgente (video o audio) a velocità di trasmissione multipla. Il client commuta tra lo streaming delle diverse codifiche a seconda delle risorse disponibili.

**Il risultato:** pochissimo buffering, tempo di avvio rapido e una buona esperienza sia per le connessioni high-end che low-end. [2]

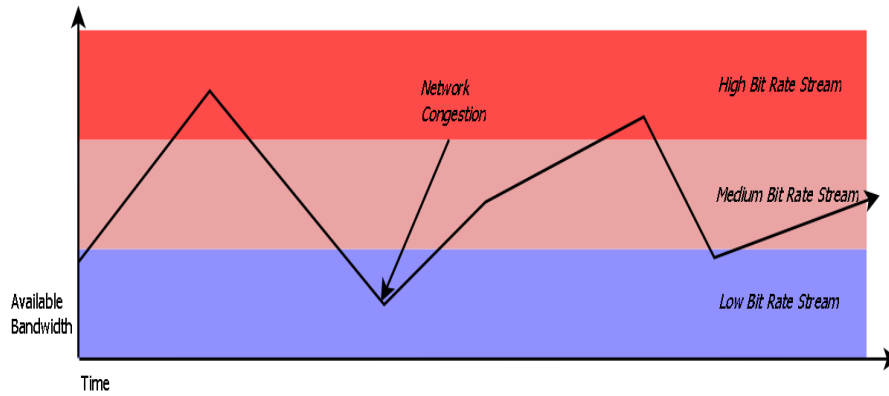


Figura 1. Adattabilità Adaptive Streaming durante varie fluttuazioni di segnale.

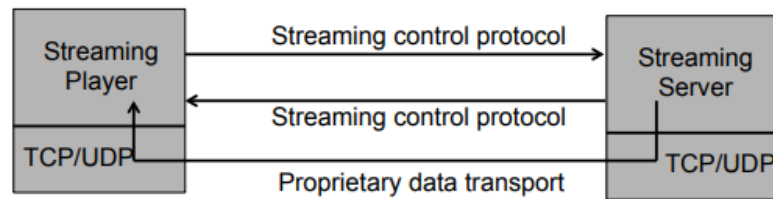
## 2.2 Soluzioni Proprietarie

Alcune software house hanno prodotto protocolli di codifica che supportano lo streaming adattivo su HTTP e quindi creano la flessibilità desiderata a seconda della situazione di ogni dispositivo.

In generale, nello streaming adattivo HTTP, un video viene codificato a più bitrate discreti. Ogni flusso di bitrate è suddiviso in più "chunks" che come grandezza variano tra i 2 e i 4 secondi. L'iesimo pezzo è allineato nella linea del tempo del video da un altro flusso di bitrate in modo che il lettore video possa passare agevolmente da una sorgente all'altra. Come mostrato in Figura 2, i protocolli di streaming adattivo basati su HTTP differiscono dai tradizionali protocolli di trasporto video orientati alla connessione in diversi aspetti importanti:

- in primo luogo, i client utilizzano il protocollo standard HTTP che fornisce una portata più onnipresente, in quanto questo traffico può attraversare NAT e firewall [25].
- in secondo luogo, i server sono server Web o cache; l'uso delle tecnologie CDN e server esistenti è stato un fattore chiave per una rapida crescita e bassi costi.
- in terzo luogo, l'uso di HTTP implica che le cache utilizzate dalle imprese e dai fornitori di servizi migliorano automaticamente le prestazioni e riducono i costi/carico di rete.

- Infine, un client recupera ogni pezzo indipendentemente e mantiene lo stato della sessione di riproduzione, cosa che i server non hanno bisogno di fare. Questo rende possibile al cliente di ricevere chunks da più server abilitando così il bilanciamento del carico sia sul lato CDN (su più server) che sul lato client. [22]



(a) Connection-oriented streaming

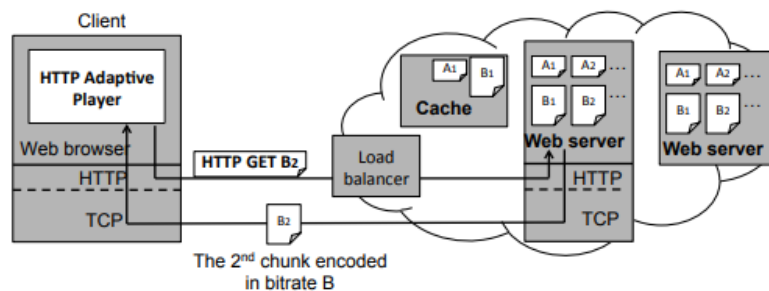


Figura 2. Differenze tra architetture streaming

Ref: <http://www.cs.cmu.edu/~junchenj/video-homepage/>

### Definizioni:

- **HDS**

HDS (HTTP Dynamic Streaming) è stato sviluppato da Adobe. Fornisce supporti MP4 adattativi basati su standard di bitrate su normali connessioni HTTP. Questo formato funziona solo su dispositivi che supportano Flash e quindi subito si può dedurre che se un dispositivo non supporta Flash non può usufruire dei servizi di HDS [19]. Inoltre HDS non può essere implementato su ordinari server HTTP. Possono essere considerati come difetti o mancanze le seguenti affermazioni:

- 1) Non è uno Standard come MPEG-DASH
- 2) Non dispone di canali multipli audio (commenti, lingue, etc.)

- 3) L'inserimento di pubblicit  non   efficiente e mirato come su MPEG-DASH
- 4) Il protocollo non supporta CDN multipli quindi la flessibilit    abbastanza limitata e non   agnostico sui codec audio e video, il che comporta una impossibilit  di aggiungere/rimuovere livelli di miglioramento durante lo streaming.
- 5) HTML 5 non risulta essere supportato

- **HSS**

Microsoft ha sviluppato un proprio formato di bitrate adattivo chiamato HTTP Smooth Streaming. Fa parte di IIS Media Services e consente lo streaming di media a Silverlight e ad altri clienti tramite HTTP. HSS rileva la larghezza di banda locale e le condizioni della CPU per commutare i bitrate in tempo quasi reale, per offrire la pi  alta qualit  video che le condizioni della rete e dei dispositivi consentiranno. I file sono consegnati in un formato MP4 frammentato e sono memorizzati come file ISMV [20]. Anche qui si possono notare alcune mancanze rispetto a MPEG-DASH:

- 1) Non   uno Standard come MPEG-DASH
- 2) Server HTTP ordinari non possono essere usati
- 3) HTML 5 non   supportato
- 4) Il protocollo non supporta CDN multipli quindi la flessibilit    abbastanza limitata e non   agnostico sui codec audio e video, il che comporta una impossibilit  di aggiungere/rimuovere livelli di miglioramento durante lo streaming.

- **HLS**

La soluzione di Apple per la consegna adattiva di bitrate si chiama HTTP Live Streaming. I file sorgente di questo formato sono impacchettati in un Transport Stream MPEG-2 che viene poi suddiviso in una serie di piccoli pezzi (file .ts) e referenziato in un indice di file di playlist .M3U8. I clienti leggono il file di indice per richiedere i pezzi nella sequenza corretta. A seconda della larghezza di banda disponibile, il client sceglie un pezzo da un bitrate appropriato che consente una transizione senza interruzioni. Questo formato è ideale per lo streaming video su dispositivi iOS in quanto è supportato nativamente su iOS 3.0 e successivi e su Safari 4.0 o successive [21]. Con un confronto si può notare che anche con questo protocollo risultano esserci delle mancanze rispetto ad altri protocolli:

- 1) Non è uno Standard
- 2) Avendo il buffer abbastanza grande, lo switching tra i canali disponibili non risulta veloce e quindi in caso di problemi la fluidità può essere compromessa
- 3) Non è agnostico sui codec audio e video
- 4) HTML 5 non è supportato quindi l'uso in applicazioni moderne è limitato

### **Distribuzione su server HTTP standard**

MPEG-DASH e anche Apple HLS possono essere utilizzati con i normali server HTTP come Apache, Nginx, IIS, ecc.

Adobe, così come Microsoft, utilizzano meccanismi lato server che richiedono una logica aggiuntiva sul server e quindi non possono usufruire di server ordinari.

### **Standard Internazionale Ufficiale**

MPEG-DASH è uno standard internazionale, ratificato nel 2012 e attualmente adottato da YouTube, Netflix, ecc.

Diversi membri di diverse aziende come Microsoft, Adobe, Apple, Samsung, Akamai, Cisco, Dolby, Ericsson, Harmonic, Qualcomm, Netflix, Intel, Bitmovin, InterDigital hanno contribuito alla standardizzazione. La Apple HLS IETF Internet-Draft \* è guidata da un'azienda che è in grado di



cambiare direzione da un giorno all'altro. Tuttavia, da maggio 2009, Apple non ha fatto alcuno sforzo per spostare l'HLS IETF Internet Draft ad uno standard internazionale IETF.

\*Gli Internet-Draft sono progetti di documenti validi per un massimo di sei mesi e possono essere aggiornati, sostituiti o obsoleti da altri documenti in qualsiasi momento. Essi non hanno alcuno status formale e sono soggetti a modifiche o rimozioni; pertanto non devono essere citati in alcun documento formale.

### **Canali audio multipli**

Il passaggio tra più canali audio è particolarmente importante per i contenuti multilingue. MPEG-DASH supporta questa funzione così come Apple HLS e Microsoft Smooth Streaming.

### **Protezione flessibile dei contenuti con crittografia comune**

MPEG Common Encryption (CENC) permette ai contenuti criptati di essere compatibili con più sistemi DRM. Questo è possibile, in quanto quasi ogni sistema DRM supporta AES come metodo di crittografia dei contenuti e solo lo scambio delle chiavi di licenza tra il client e il server è diverso.

### **Inserimento efficiente degli annunci**

L'inserimento degli annunci è possibile in tutti i formati attraverso la sostituzione del pezzo. Questo significa che i singoli pezzi del video originale saranno semplicemente sostituiti da pezzi che contengono annunci pubblicitari. MPEG-DASH consente un'interfaccia standardizzata attraverso periodi che consente un inserimento efficiente degli annunci. Ciò significa che i normali server HTTP possono ancora essere utilizzati e non è necessaria una logica proprietaria aggiuntiva per reindirizzare le richieste di pezzi specifici verso i pezzi che contengono l'annuncio.

### **Commutazione veloce dei canali**

Il cambio canale è una caratteristica che è direttamente correlata alla dimensione del pezzo, poiché i pezzi più piccoli consentono tempi di cambio canale più rapidi rispetto ai pezzi più grandi. Apple HLS utilizza solitamente pezzi da 10 secondi ed è ottimizzato per quella dimensione del pezzo. Adobe, Microsoft e MPEG-DASH sono progettati per funzionare

con blocchi di 2 o 4 secondi che consentono un cambio canale più veloce. Inoltre, l'overhead del formato MP4 utilizzato in MPEG-DASH e Microsoft e' notevolmente inferiore rispetto al formato MPEG-2 Transport Stream (MP2TS) utilizzato in Apple HLS.

Il compromesso tra grandi e piccole dimensioni dei chunk è che le piccole dimensioni dei predetti consentono un cambio canale veloce, riducendo la latenza di avvio per alcuni sistemi (alcuni lettori avviano la riproduzione solo se hanno ricevuto completamente il primo chunk), e consentono un comportamento di adattamento più flessibile. D'altra parte, le piccole dimensioni dei chunk riducono l'efficienza di codifica in quanto il Group of Pictures (GOP) deve essere più piccolo e quindi il codec probabilmente non può beneficiare pienamente di maggiori ridondanze temporali quando sono consentiti anche GOP più grandi. Inoltre, diminuendo le dimensioni dei chunk si introduce un overhead sul livello di rete, in quanto ogni chunk deve essere richiesto tramite HTTP.

### **Supporto di CDN multipli in parallelo**

L'uso di CDN multipli è possibile con tutti i formati, ma solo MPEG-DASH offre un meccanismo di segnalazione attraverso BaseURLs a livello MPD per il client dove sono in uso più CDN. L'applicazione client è quindi in grado di selezionare il CDN che meglio si adatta, cioè il CDN che offre le migliori prestazioni nella rete del singolo client in quel dato momento. Tutti i formati che non possono essere distribuiti sui normali server HTTP, come Adobe HDS e Microsoft Smooth Streaming, sono più costosi perché richiedono funzionalità CDN specializzate.

Per avere un'idea più concreta rispetto alle differenze dei protocolli in relazione a MPEG-DASH, segue una rappresentazione grafica:

Feature	Adobe HDS	Apple HLS	MS Smooth Streaming	MPEG-DASH
Deployment on Ordinary HTTP Servers		✓		✓
Official International Standard (e.g., ISO/IEC MPEG)				✓
Multiple Audio Channels (e.g., Languages, Comments, etc.)		✓	✓	✓
Flexible Content Protection with Common Encryption (DRM)	✓	✓	✓	✓
Closed Captions / Subtitles	✓	✓	✓	✓
Efficient Ad Insertion				✓
Fast Channel Switching	✓		✓	✓
Protocol Support's multiple CDNs in parallel				✓
HTML5 Support				✓
Support in HbbTV (version 1.5)				✓
HEVC Ready (UHD/4K)				✓
Agnostic to Video Codecs				✓
Agnostic to Audio Codecs				✓
ISO Base Media File Format Segments	✓		✓	✓
MPEG-2 TS Segments		✓		✓
Segment Format Extensions beyond MPEG				✓
Support for multiplexed (Audio + Video) Content	✓	✓		✓
Support for non-multiplexed (separate Audio, Video) Content		✓	✓	✓
Definition of Quality Metrics				✓

Figura 3. Differenze tra i principali protocolli,

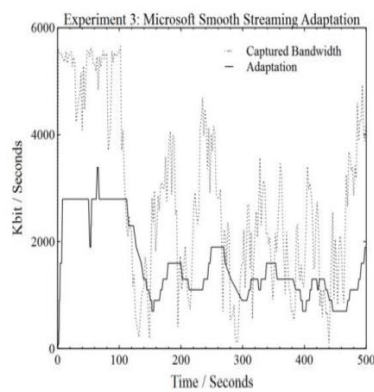
Ref: <https://bitmovin.com/mpeg-dash-vs-apple-hls-vs-microsoft-smooth-streaming-vs-adobe-hds/>

## 2.2.1 Test Effettuati Con I Protocolli Proprietari

Nella sezione sottostante si valutano i risultati di vari esperimenti riguardanti i software proprietari che gestiscono ed implementano tecniche HAS(HTTP Adaptive Streaming) [1].

### Smooth-Streaming Adaptation

**Adaptation:**



**Buffer:**

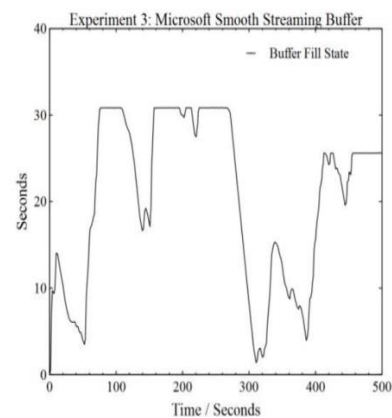


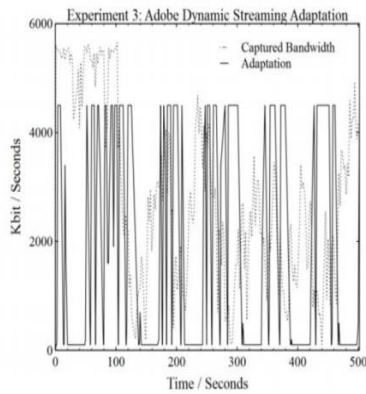
Figura HSS. Test effettuati sui campioni in [1]

Ref: [1]

Qui il bitrate del video si adatta con il throughput stimato mentre si può notare una prestazione abbastanza costante con un leggero problema di fluidità' (buffering) verso i 300 secondi. La causa di ciò, per esempio, potrebbe trovare riscontro nel fatto che la CPU sta processando altri dati e quindi rende di meno.

## HTTP Dynamic Streaming

**Adaptation:**



**Buffer:**

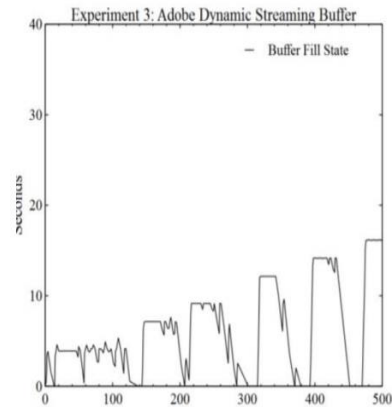


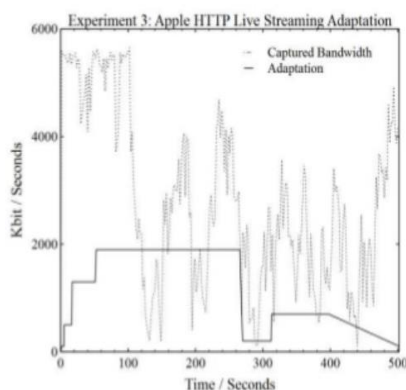
Figura HDS. Test effettuati sui campioni in [1]

Ref: [1]

ADS (HDS) ha degli sbalzi tra le representation (qualitative) perché cerca di offrire la migliore qualità possibile. Avendo però sbalzi così frequenti causa dei periodi di non fluidità e quindi anche il buffer ne risente non avendo una media costante. Questo tipo di problema si nota maggiormente su conessioni di qualità medio-bassa che faticano a riprendersi dopo uno switch.

## HTTP-Live Streaming

**Adaptation:**



**Buffer:**

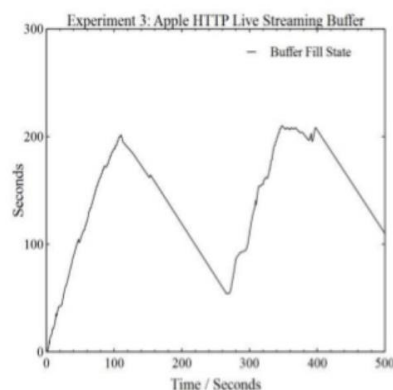


Figura HLS. Test effettuati sui campioni in [1]

Ref: [1]

Nei diagrammi sovrastanti che riguardano HLS si vede che l'approccio è progressivo ma ad un bitrate moderato rispetto a HSS. Inoltre siccome HLS viene progettato per dispositivi mobili, il buffer risulta più grande per migliorare l'efficienza del predetto.

Elaborando quanto menzionato sopra si può dedurre che sebbene le soluzioni proposte siano efficaci, manca una standardizzazione avente come scopo il supporto della maggior parte dei dispositivi, indipendentemente dal sistema operativo o dalla tecnologia supportata di ogni dispositivo. Come proposta emergente, nell'aprile 2009 è MPEG che pubblicando un invito presenta alcune idee per uno standard di streaming HTTP. Nei due anni successivi, MPEG ha sviluppato la specifica con la partecipazione di molti esperti e con la collaborazione di altri gruppi di standard, come il Third Generation Partnership Project (3GPP). Sono state coinvolte più di 50 aziende - Microsoft, Netflix e Adobe inclusi - e lo sforzo è stato coordinato con altre organizzazioni del settore come il Digital Entertainment Content Ecosystem, LLC (DECE), OIPF e World Wide Web Consortium (W3C). Ciò ha portato allo sviluppo dello standard **MPEG-DASH** che verrà presentato in seguito.

## 2.3 Algoritmi Adattivi

MPEG-DASH, come qualsiasi altro protocollo relazionato allo streaming adattivo trasmesso tramite HTTP, fa uso di tecniche HAS (HTTP Adaptive Streaming). In questa sezione verranno presentati gli algoritmi più significativi che vengono usati da questa tecnica.

### Throughput-based Adaptation

Questo schema di adattamento e' basato sul throughput TCP molto comune [4, 5] il quale si suddivide in quattro fasi:

#### 1) Throughput estimation

Inizialmente viene stimato il valore del throughput attraverso sonde di monitoraggio. Le sonde sono misurazioni che hanno lo scopo di stimare la banda disponibile all'interno di un segmento di rete che va da un end-point ad un altro. Il sistema di misurazione è costituito da un processo in esecuzione sul nodo sorgente e da un altro processo presente sul nodo destinatario.

#### 2) Smoothing

Il valore preso viene "smussato" con l'ausilio di diversi tool che tramite le risposte dei pacchetti-sonda (le sudette misurazioni) riescono ad minimizzare gli errori di stima, così che il valore rispecchi la situazione reale e che quindi il player carichi i file corrispondenti alla larghezza di banda. La stima della banda disponibile viene effettuata tramite l'analisi del traffico generato da altri protocolli, rendendo quindi non necessaria la generazione di un traffico di prova [28].

#### 3) Quantization

Dopo il processo di valutazione, il throughput viene mappato nelle varie representations descritte nel MPD rendendo così possibile lo streaming adattivo.

#### 4) Scheduling

La richiesta del segmento successivo viene valutata appena la inter-request time (tempo tra le richieste dei segmenti) e' nota. Questo viene calcolato per poter stimare anche la larghezza di banda.

## IMPLEMENTAZIONI

**Conventional:** è un semplice algoritmo di adattamento, basato sul modello a quattro fasi, che equipara l'attuale larghezza di banda disponibile con il throughput, misurato durante il precedente download del segmento. Quindi, il bitrate del video proposto viene prodotto applicando un filtro di media mobile ponderata esponenziale. L'algoritmo determina il tempo di inter-request del segmento successivo utilizzando uno scheduler bi-modale, in base al quale la richiesta del segmento successivo viene pianificata con un ritardo costante quando il buffer è pieno o immediatamente dopo.

**PANDA [5]:** è una variazione avanzata del modello a quattro fasi, ma con due modifiche distinte. Nella fase di stima questo algoritmo utilizza un meccanismo di sondaggio più proattivo, progettato per minimizzare le oscillazioni del bitrate video. La seconda modifica è nella fase di schedulazione, dove viene considerato uno scheduler più sofisticato che spinge il livello del buffer verso il massimo livello di occupazione  $B_{max}$ . Allo stesso tempo l'inter-request time viene abbinato al tempo necessario per completare il download sulla base del valore stimato della larghezza di banda disponibile.

## Buffer-based Adaptation

**BBA:** è un algoritmo di adattamento basato su buffer molto noto. In [6], gli autori introducono una mappa basata sulla dimensione media dei segmenti per ogni rappresentazione. La mappa è definita da due soglie:

- 1) una soglia superiore che spinge l'intelligenza artificiale a selezionare la qualità massima disponibile ( $R_{max}$ ) una volta che l'occupazione istantanea del buffer la supera
- 2) una soglia inferiore che detta la qualità più bassa disponibile ( $R_{min}$ ), se il buffer è inferiore a tale soglia.

Nella regione di buffer tra queste soglie, l'intelligenza artificiale può utilizzare qualsiasi funzione non decrescente per selezionare la qualità.



**BOLA** [7]: è un algoritmo di adattamento basato su buffer che utilizza l'ottimizzazione Lyapunov per indicare il bitrate video di ogni segmento. In pratica, l'algoritmo è stato progettato per massimizzare la funzione di utilità congiunta che premia un aumento della qualità media e penalizza potenziali eventi di riacquisizione. In particolare, una variazione chiamata BOLA-O, mitiga le oscillazioni del bitrate video introducendo una forma di limitazione del bitrate quando si passa a bitrate più elevati.

La performance degli algoritmi è altamente influenzata dalle variazioni del throughput di rete e quindi per evitare stalli indesiderati il predetto deve cercare di essere maggiore della minore qualità disponibile nel server.

## **Time-based Adaptation**

**ABMA+** [8]: Si tratta di un algoritmo di adattamento e gestione del buffer che seleziona la rappresentazione video in base alla probabilità prevista di stallo video. L'algoritmo stima continuamente il tempo di download del segmento e utilizza una mappa buffer di play-out precalcolato per selezionare la massima rappresentazione video, che garantisce una riproduzione fluida dei contenuti. La stima del tempo di download del segmento si basa sullo stesso meccanismo di sondaggio del metodo basato sul throughput.

## 2.4 Caratteristiche e Quality of Experience (QoE)

La QoE e' un fattore molto importante quando si parla di streaming adattivo. Infatti, lo scopo e' quello di massimizzare la QoE per favorire agli utenti un'esperienza di streaming fluida e senza interruzioni, dato che ormai molte cose vengono fatte tramite canali di comunicazione.

Ci sono 3 variabili (AIS) che possono rendere valutabile la QoE.

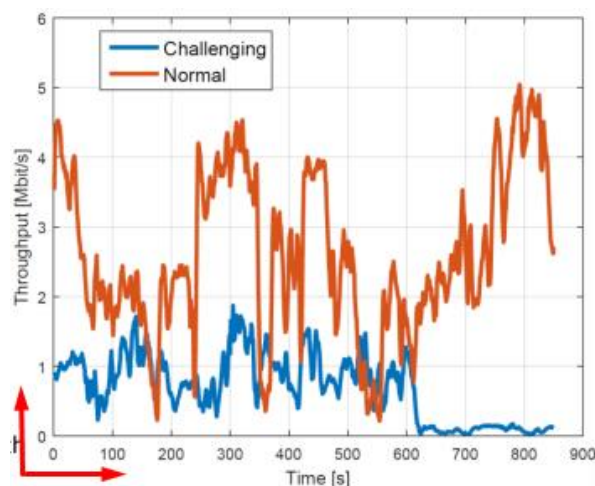
1. Adaptability,
2. Instability,
3. Smoothness

In generale la versatilità dell'algoritmo usato nello streaming adattivo risulta essere un fattore critico. Siccome le connessioni non hanno un throughput fissato, le condizioni di rete possono fluttuare da un momento all'altro. Per massimizzare in questo campo la qualità di esperienza, l'algoritmo deve cercare di adattarsi ad ogni condizione così da poter evitare stalli indesiderati. Se non riesce ad adattarsi, questo comporta una certa instabilità e quindi fa diminuire notevolmente la QoE. Se per esempio siamo fuori in un bosco e stiamo cercando di visualizzare del contenuto multimediale ma non riusciamo perché ogni 20 secondi si verifica uno stallo, questo ha un impatto negativo sulla nostra esperienza.

Definiamo come:

**Normal:** Situazione giornaliera (autobus) e

**Challenging:** Situazione estrema (siamo nella metro)



La riduzione del bitrate in base alla potenza del throughput e' segno della presenza di un algoritmo ABR (Adaptive Bit-Rate)

Figura 3. Esempio caratteristico dei profili di rete in campo reale selezionati

## 2.4.1 AIS

### 1. Adaptability

Gli algoritmi BB (Buffer Based) raggiungono una maggiore adattabilità in condizioni normali.

Sono più efficaci, per progettazione, nel

conservare alti livelli di rappresentazioni, anche superiori al throughput disponibile. Gli algoritmi basati sul throughput e gli algoritmi basati sul tempo mostrano una capacità leggermente diminuita di abbinare la rappresentazione al throughput medio disponibile. Ciò è dovuto alla significativa variazione di throughput che caratterizza i profili di rete selezionati.

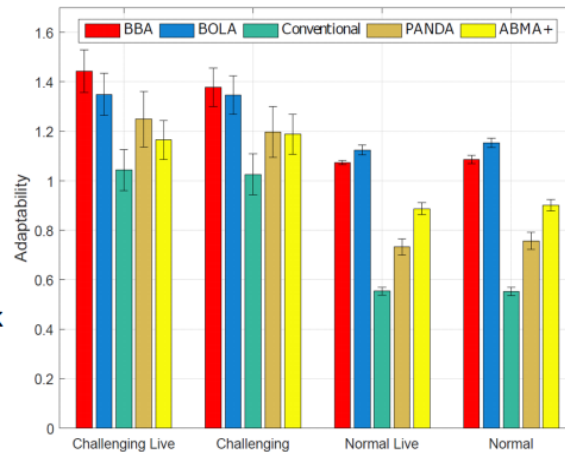


Figura 4. Risultati di adattabilità degli algoritmi

Ref: <http://www.tsi.telecom-paristech.fr/>

### 2. Instability

Nella figura accanto si nota che gli algoritmi basati su buffer hanno circa il 40% in più di probabilità di fare uno switch quando il buffer è piccolo.

- BOLA è ottimizzato per essere efficiente ma, l'aspetto della stabilità non è considerato nell'ottimizzazione, dal

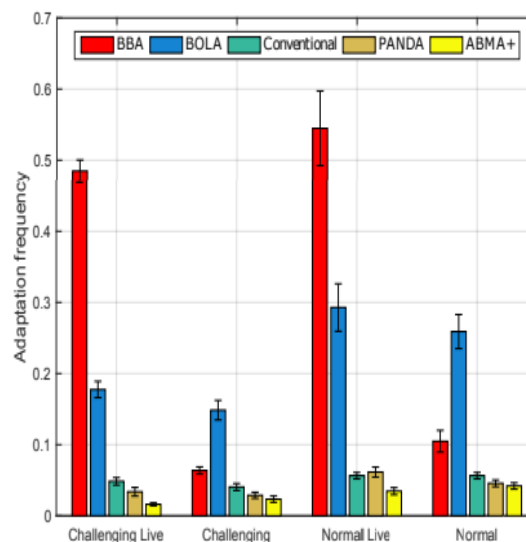


Figura 5. Frequenza con la quale si adatta ogni algoritmo.

Ref: <http://www.tsi.telecom-paristech.fr/>

momento che è indirizzata con un'opzione euristica in una seconda fase.

- BBA ha una threshold preselezionata più alta che rende il segmento di mappatura meno dinamico alla variazione di throughput quando il buffer massimo è piccolo. Al contrario, gli algoritmi basati sul throughput e sul tempo sembrano cambiare qualità meno spesso, ma con ampiezza di adattamento simile, che è complementare alla frequenza di adattamento se si vuole trarre una conclusione sulla stabilità. La figura sottostante mostra la distanza media normalizzata tra gli switch. Le prestazioni di questa metrica mostrano un leggero vantaggio a favore degli algoritmi basati sul throughput, sia in profili normali che impegnativi.

E' importante ricordare che nessuna metrica deve essere trattata separatamente e che solo la combinazione di tutte le metriche permette un confronto intuitivo. Nel complesso, i risultati coincidono con quelli di [7]. Inoltre, i risultati possono essere verificati con quelli in [8] dove PANDA e BBA sono confrontati con ABMA+.

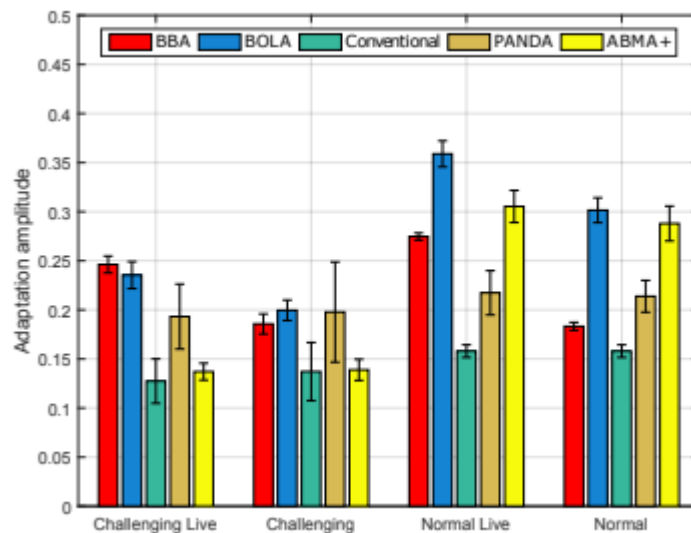


Figura 6. Ampiezza di adattabilità per ogni classe

Ref: <http://www.tsi.telecom-paristech.fr/>

Nella Tabella 1 sottostante notiamo le classi di algoritmi più performanti, per elemento QoE. Questa tabella può servire da spunto per la selezione della classe algoritmica più appropriata, a seconda dei parametri applicativi (live, VOD, ecc.) e delle condizioni di rete comunemente sperimentate.

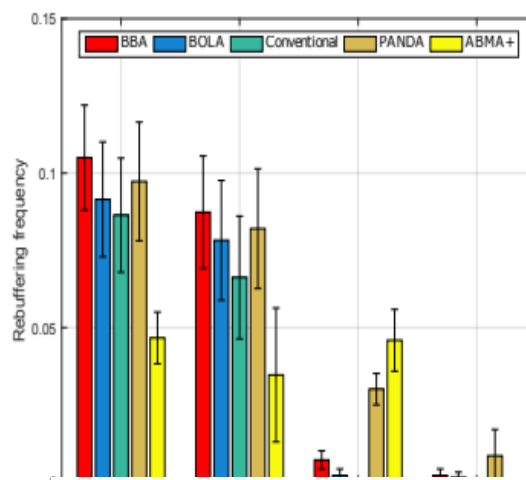
Buffer Size	Network condition	Adaptability	Re-buffering frequency	Re-buffering duration	Adaptation frequency	Adaptation amplitude
Small (16 s)	Normal	Buffer	Time	Time	Time	Throughput
Small (16 s)	Challenging	Buffer	Time	Time	Time	Throughput
Large (92 s)	Normal	Buffer	Buffer	Buffer	Time	Throughput
Large (92 s)	Challenging	Buffer	Buffer	Buffer	Time	Throughput

**Tabella 1. Architettura degli algoritmi piu' performanti rispetto alla grandezza del buffer**

Ref: [29]

### 3. (UN) Smoothness

Dalla figura accanto possiamo notare che la probabilità di re-buffering è leggermente superiore nei casi dove il buffer risulta essere di dimensioni limitate (esempio: live streaming). Per quanto riguarda le frequenze di re-buffering per classe di algoritmi, possiamo notare che anche se le prestazioni sono molto vicine, su profili



**Figura 7. Rebuffering relazionata alla situazione di banda**

Ref: <http://www.tsi.telecom-paristech.fr/>

impegnativi gli algoritmi basati su buffer, insieme a PANDA, sono leggermente più tendenti a soffrire di re-buffering.

Per gli scenari normali si assiste ad uno streaming fluido da quasi tutti gli algoritmi, a causa dell'assenza di lunghe interruzioni di throughput di questo profilo. Tuttavia, ABMA+ mostra una un-smoothness leggermente aumentata rispetto al resto degli algoritmi nel profilo normale con un piccolo buffer, a causa del fatto che il numero di sonde (50) proposto dai progettisti dell'algoritmo è molto elevato rispetto al buffer massimo. Pertanto un breve drop nel throughput non viene registrato nella stima in tempo utile prima che il buffer si sia esaurito.

## Capitolo 3

---

### Protocollo MPEG-DASH

#### 3.1 MPEG-DASH

##### Obiettivi chiave e vantaggi di MPEG-DASH:

- riduzione dei ritardi di avvio e di buffering durante il video (cache)
- continuo adattamento alla situazione della larghezza di banda del cliente
- logica di streaming basata su client che consente la massima scalabilità e flessibilità
- uso di CDN, proxy e cache esistenti e convenienti dal punto di vista economico basati su HTTP
- bypassamento efficiente di NAT e Firewalls mediante l'utilizzo di HTTP
- Crittografia comune - segnalazione, consegna e utilizzo di più schemi DRM simultanei dallo stesso file.
- giunzione semplice e inserimento (mirato) di annunci

Negli ultimi anni, MPEG-DASH è stato integrato in nuovi sforzi di standardizzazione, ad esempio, le HTML5 Media Source Extensions (MSE) che consentono la riproduzione DASH tramite il tag *video* e *audio* HTML5, così come le HTML5 Encrypted Media Extensions (EME) che consentono la riproduzione protetta da DRM nei browser web. Oggi, MPEG-DASH sta guadagnando sempre più diffusione, accelerata da servizi come Netflix o Google che di recente è passato a questo nuovo standard. Con queste due grandi fonti di traffico Internet, il 50% del traffico Internet totale è già MPEG-DASH!

## 3.2 Come si rende possibile uno streaming efficiente?

L'idea di base di MPEG-DASH è la seguente:

l'obiettivo è quello di tagliare il file multimediale in segmenti che possano essere codificati a bitrate differenti. I segmenti sono forniti su un server web e possono essere scaricati attraverso richieste GET conformi allo standard HTTP/HTTPS come mostrato nella figura sottostante, dove il server HTTP serve tre diverse qualità, cioè, **Low**, **Medium** e **Best**, tagliati in segmenti di uguale lunghezza. L'adattamento al bitrate o alla risoluzione viene fatto sul lato client per ogni segmento, il che significa che il client può passare ad un bitrate più alto se la larghezza di banda lo permette.

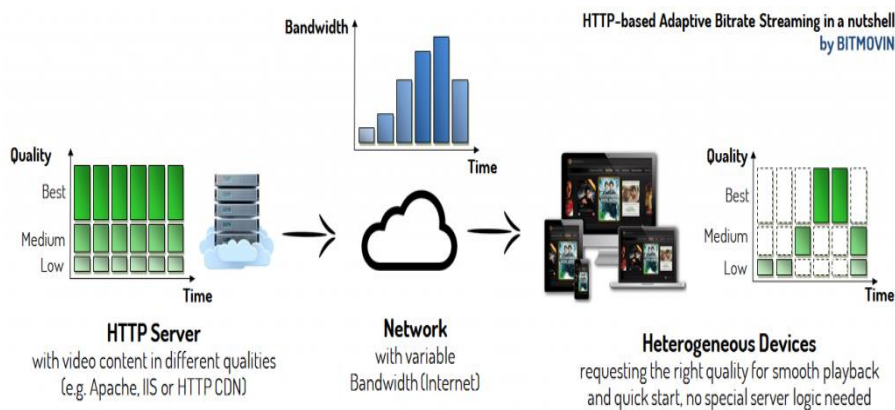


Figura 8. Architettura del processo di trasmissione dati

Ref: [24]

Per descrivere le relazioni temporali e strutturali tra i segmenti, MPEG-DASH ha introdotto la **Media Presentation Description (MPD)**. Il MPD è un file XML che rappresenta le diverse qualità del contenuto multimediale e dei singoli segmenti di ogni qualità con gli URL. Questa struttura fornisce il legame dei segmenti al bitrate (risoluzione, ora di inizio, durata dei segmenti, etc). Di conseguenza, ogni client richiede innanzitutto il MPD che contiene le informazioni temporali e strutturali per il contenuto dei media e, sulla base di tali informazioni, richiede poi i singoli segmenti che meglio si adattano alle sue esigenze.

### 3.3 Struttura Media Presentation Description (MPD)

La descrizione del MPD è un modello di dati gerarchico. Ogni MPD può contenere uno o più Periodi. Ognuno di questi Periodi contiene componenti multimediali come componenti video, diversi angoli di visualizzazione o codec diversi, componenti audio per lingue diverse o con diversi tipi di informazioni (ad esempio, con commenti del regista, ecc.), componenti di sottotitoli o didascalie, ecc.

Questi componenti hanno determinate caratteristiche come il bitrate, il frame rate, i canali audio, ecc. che non cambiano durante un periodo. Tuttavia, il cliente è in grado di adattarsi durante un Periodo in base al bitrate, risoluzioni, codec, ecc. disponibili. Inoltre, un Periodo potrebbe separare il contenuto, ad esempio, per l'inserimento di un annuncio, per cambiare l'angolazione della telecamera in una partita di calcio dal vivo, ecc.

Se un annuncio dovesse essere disponibile solo in alta risoluzione mentre il contenuto è disponibile dalla definizione standard all'alta definizione, è sufficiente introdurre un Periodo che contiene solo il contenuto dell'annuncio in alta definizione. Dopo e prima di questo periodo, ci sono altri periodi che contengono il contenuto effettivo (ad esempio, film) in bitrate multipli e risoluzioni da standard ad alta definizione.

Tipicamente i Periodi sono disposti in **AdaptationSets**. Ogni periodo può contenere uno o più AdaptationSets che consentono il raggruppamento di diversi componenti multimediali correlate tra loro. Ad esempio, componenti con lo stesso codec, lingua, risoluzione, formato del canale audio, ecc. potrebbero trovarsi all'interno dello stesso AdaptationSet.

*[Questo meccanismo permette al cliente di eliminare una serie di componenti multimediali che non soddisfano i suoi requisiti e quindi risparmiare sulle risorse impiegate durante il playback.] [17]*



Un `AdaptationSet` è costituito da un insieme di `Representations` che contengono versioni intercambiabili dei rispettivi contenuti come diverse risoluzioni, bitrate ecc.

Anche se una singola rappresentazione sarebbe sufficiente a fornire un flusso riproducibile (che però sarà della stessa qualità), rappresentazioni multiple danno al cliente la possibilità di adattare il flusso multimediale alle condizioni di rete e ai requisiti della larghezza di banda attuali, garantendo così una riproduzione fluida e adattiva.

Naturalmente, ci sono anche altre caratteristiche oltre alla larghezza di banda che descrivono le diverse rappresentazioni e consentono l'adattamento. Le rappresentazioni possono differire per quanto riguarda il codec utilizzato, la complessità di decodifica e quindi le risorse necessarie della CPU e/o la tecnologia di rendering per visualizzare il flusso.

### **3.4 Struttura delle Representations**

Le rappresentazioni sono suddivise in segmenti per consentire il passaggio tra esse durante la riproduzione. Quest'ultimi sono descritti da un URL e in alcuni casi da un intervallo di byte aggiuntivo se i segmenti sono memorizzati in un file più grande e continuo. I segmenti in una rappresentazione hanno di solito la stessa lunghezza in termini di tempo e sono disposti secondo la timeline di presentazione mediatica che rappresenta la timeline per la sincronizzazione, consentendo il passaggio fluido delle rappresentazioni durante la riproduzione. I segmenti potrebbero anche avere un tempo di disponibilità segnalato come orario wall-clock da cui sono accessibili scenari di live streaming.

A differenza di altri sistemi, MPEG-DASH non limita la lunghezza del segmento e nemmeno fornisce consigli sulla lunghezza ottimale. Questo può essere scelto a seconda dello scenario dato, ad esempio, segmenti più lunghi permettono una compressione più efficiente in quanto ci sarà un numero limitato di richieste HTTP dato che ad ogni richiesta viene introdotta una certa quantità di overhead. Segmenti più brevi sono invece utilizzati per scenari live e per condizioni di larghezza di banda molto variabili come le reti mobili, in quanto consentono una commutazione più rapida e flessibile tra i singoli bitrate.

I segmenti possono anche essere suddivisi in sottosegmenti più piccoli che rappresentano un insieme di unità di accesso nel segmento in questione. In questo caso, è disponibile un indice che descrive l'intervallo di tempo di presentazione e la posizione in byte dei sottosegmenti che può essere scaricato in anticipo dal cliente per generare le richieste appropriate utilizzando HTTP 1.1.

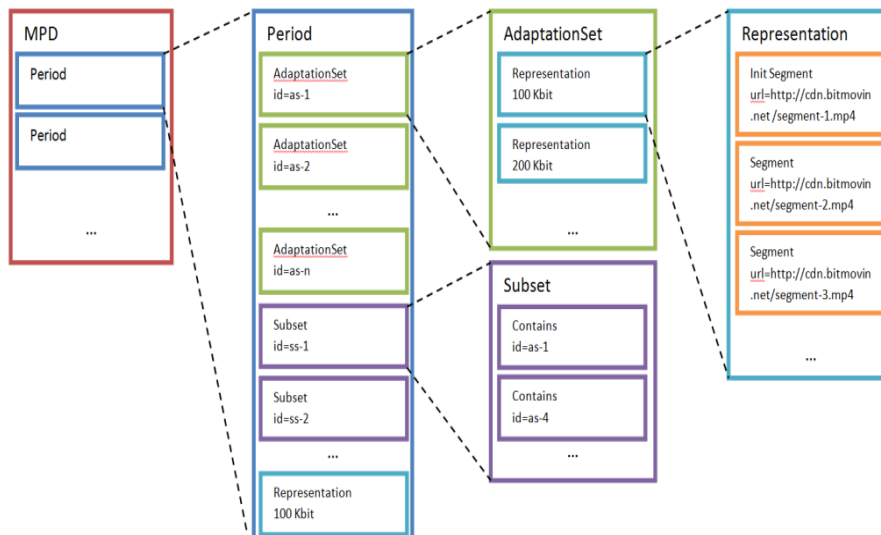


Figura 9. Una rappresentazione grafica della struttura di un MPD,

Ref: [24]

Durante la riproduzione del contenuto, non è possibile passare arbitrariamente da una rappresentazione all'altra in nessun punto del flusso e devono essere considerati alcuni vincoli. I segmenti non possono sovrapporsi e non sono ammesse dipendenze tra di loro. Per consentire il passaggio da una rappresentazione all'altra, MPEG-DASH ha introdotto gli **Stream Access Point (SAP)** con cui ciò è possibile. Ad esempio, ogni segmento inizia tipicamente con un frame IDR contenente informazioni che permettono al player di cambiare rappresentazione durante la riproduzione.

### 3.5 SegmentBase

SegmentBase è il modo più banale di referenziare i segmenti nello standard MPEG-DASH, in quanto esso verrà utilizzato quando esiste un solo segmento di supporto per la rappresentazione, che sarà poi referenziato attraverso un URL nell'elemento BaseURL. Se una rappresentazione deve contenere più segmenti, è necessario utilizzare l'elemento SegmentList o SegmentTemplate.

#### La rappresentazione con Segment Base può avere questo aspetto

```
<Representation mimeType="video/mp4"
    frameRate="24"
    bandwidth="1558322"
    codecs="avc1.4d401f"    width="1277"
height="544">
  <BaseURL>http://domain.com/rep/video-
1500k.mp4</BaseURL>
  <SegmentBase indexRange="0-834"/>
</Representation>
```

L'esempio di rappresentazione fa riferimento ad un singolo segmento attraverso il BaseURL con una qualità video a 1500 kbps. L'indice di qualità è descritto dall'attributo SegmentBase indexRange. Ciò significa che le informazioni sui Random Access Point (RAP) e altre informazioni di inizializzazione sono contenute nei primi 834 byte e quindi per quella durata lì non si può cambiare segmento.

### 3.6 SegmentList

La SegmentList contiene un elenco di elementi SegmentURL che dovrebbero essere riprodotti dal client nell'ordine in cui sono stati dichiarati all'interno del MPD. Un elemento SegmentURL contiene un URL che punta ad un segmento ed eventualmente ad un intervallo di byte. Inoltre, un segmento di indice potrebbe verificarsi all'inizio della SegmentList.

### La rappresentazione con SegmentList potrebbe avere questo aspetto:

```
<Representation mimeType="video/mp4"
    frameRate="24"
    bandwidth="1558322"
    codecs="avc1.4d401f"    width="1277"
height="544">
    <SegmentList duration="10">
        <Initialization
sourceURL="http://domain.com/rep/video-1500/init.mp4"/>
        <SegmentURL media=" http://domain.com/rep /video-
1500/segment-0.m4s"/>
        <SegmentURL media=" http://domain.com/rep /video-
1500/segment-1.m4s"/>
        <SegmentURL media=" http://domain.com/rep /video-
1500/segment-2.m4s"/>
        <SegmentURL media=" http://domain.com/rep /video-
1500/segment-3.m4s"/>
        <SegmentURL media=" http://domain.com/rep /video-
1500/segment-4.m4s"/>
    </SegmentList>
</Representation>
```

## 3.7 SegmentTemplate

L'elemento SegmentTemplate fornisce un meccanismo per costruire una lista di segmenti seguendo un dato template. Questo significa che gli identificatori specifici saranno sostituiti da valori dinamici per creare una lista di segmenti. Facendo ciò, si hanno diversi vantaggi se si tiene conto che il MPD basati su SegmentList possono diventare molto grandi. Questa affermazione trova riscontro nel fatto che ogni segmento ha bisogno di essere referenziato individualmente rispetto a SegmentTemplate, che con poche righe si può indicare come costruire una grande lista di segmenti in modo dinamico.

```
<Representation mimeType="video/mp4"
    frameRate="24"
    bandwidth="1558322"
    codecs="avc1.4d401f"    width="1277"
height="544">
    <SegmentTemplate
media="http://cdn.bitmovin.net/bbb/video-1500/segment-
$Number$.m4s"

initialization="http://cdn.bitmovin.net/bbb/video-
1500/init.mp4"
startNumber="0"
```

```

        timescale="24"
        duration="48"/>
</Representation>

```

L'esempio precedente mostra il meccanismo con il quale si può raggiungere la dinamicità del processo di creazione di liste. Come si può vedere, invece di avere più riferimenti a singoli segmenti attraverso il SegmentURL, come mostrato nell'esempio SegmentList, un SegmentTemplate può descrivere questo caso d'uso in poche righe, il che rende il MPD più compatto. Questo è particolarmente utile per film più lunghi con rappresentazioni multiple, dove un MPD implementato con una SegmentList potrebbe avere più megabyte, il che aumenterebbe notevolmente la latenza di avvio di uno stream, dato che il client deve recuperare i dati forniti nel MPD prima che possa iniziare il processo di streaming.

L'elemento SegmentTemplate potrebbe anche contenere un identificatore *\$Time\$*, che sarà sostituito dal valore dell'attributo *t* della SegmentTimeline.

L'elemento SegmentTimeline fornisce un'alternativa all'attributo **duration** con caratteristiche aggiuntive come:

1. le durate arbitrarie dei segmenti
2. la durata esatta del segmento
3. le discontinuità nella cronologia dei media

Ad esempio, l'estratto di un MPD con un SegmentTemplate basato su una SegmentTimeline è mostrato di seguito.

```

<Representation mimeType="video/mp4"
    frameRate="24"
    bandwidth="1558322"
    codecs="avc1.4d401f"    width="1277"
    height="544">
  <SegmentTemplate    media="    http://domain.com/rep
/video-1500/segment-$Time$.m4s"
    initialization="
http://domain.com/rep /video-1500/init.mp4"
    timescale="24">
    <SegmentTimeline>
      <S t="0" d="48" r="5"/>

```

```
</SegmentTimeline>  
</SegmentTemplate>  
</Representation>
```

Le richieste risultanti del cliente sarebbero le seguenti:

- <http://domain.com/rep/video-1500/init.mp4>
- <http://domain.com/rep/video-1500/segment-0.m4s>
- <http://domain.com/rep/video-1500/segment-48.m4s>
- <http://domain.com/rep/video-1500/segment-96.m4s>

### 3.8 Test Effettuati Con MPEG-DASH

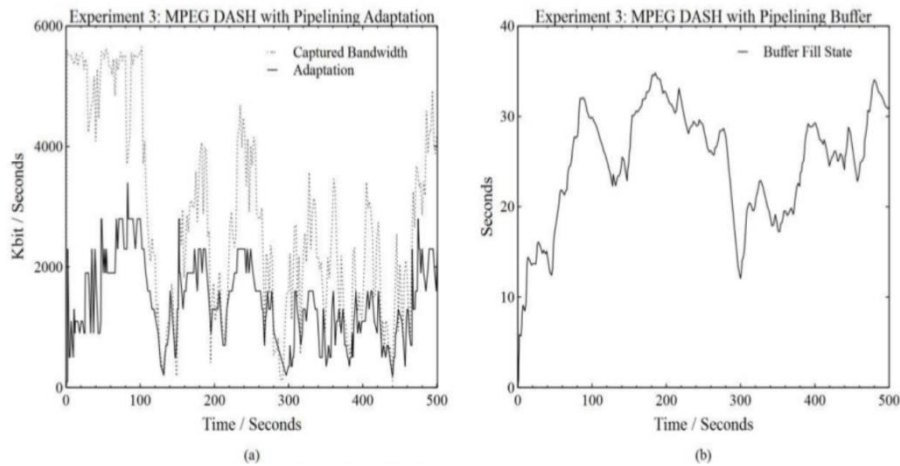


Figura DASH.

#### Comportamento di MPEG-DASH in relazione alla banda

Ref: [1]

#### Resoconto dei confronti:

A differenza degli altri protocolli, MPEG-Dash è molto più flessibile. Con il suo approccio non step-wise rende la QoE migliore data la sua dinamicità [1]. Inoltre, ha un bitrate moderato ed un buffer abbastanza stabile e con il fatto di poter inviare più richieste HTTP senza aspettare le risposte corrispondenti (pipelining), il carico di rete si riduce assicurando la fluidità desiderata.

#### Client Failover

MPEG-DASH permette attraverso BaseURLs di segnalare diversi meccanismi di failover dei CDN, in quanto è possibile scegliere più posizioni per scaricare un segmento. Un approccio simile è disponibile con Apple HLS che utilizza URL di failover, ma in questo caso la specifica MPEG-DASH consente una maggiore flessibilità e casi d'uso più sofisticati.

#### Rimuovere e aggiungere livelli di qualità durante lo streaming

MPEG-DASH consente, attraverso Periodi, l'inserimento dinamico e la cancellazione di rappresentazioni di qualità durante lo streaming senza interruzioni sul lato client.

### Viste video multiple

MPEG-DASH consente, attraverso AdaptationSets, viste multiple nello stesso MPD il che permette al client di scegliere, senza interruzioni, diversi angoli di visualizzazione durante lo streaming.

Ad esempio, un caso d'uso della Formula 1 con tre prospettive della telecamera avrebbe tre AdaptationSet che contengono le rappresentazioni rispettive dell'angolo di visione. Il cliente potrebbe quindi passare da un set di adattamento all'altro cambiando così quello che sta guardando senza particolari interruzioni. All'interno di un AdaptationSet (prospettiva) il client può adattarsi, sulla base delle rappresentazioni disponibili nell'AdaptationSet, alle condizioni di larghezza di banda o al contesto utente (dispositivo).

Name	Average Bitrate [kbps]	Average Switches [Number of Switches]	Average Unsmoothness [Seconds]
Microsoft	1522	51	0
Adobe	1239	97	64
Apple	1162	7	0
MPEG-DASH	1045	141	0

**Tabella 2. Resoconto comportamenti degli algoritmi di varie software house**

Ref: [1]



## Capitolo 4

### Scalable Video Coding (SVC)

#### 4.1 Introduzione

Il principio di un codificatore video scalabile è quello di decomporre il video convenzionale in un flusso multi-stream, composto da componenti distinti e complementari, spesso denominati layer [9]. La figura 13 illustra questo concetto raffigurando un trasmettitore che codifica la sequenza video in ingresso in quattro diversi strati complementari. Pertanto, i ricevitori possono selezionare e decodificare un numero diverso di strati - ciascuna delle quali corrisponde a caratteristiche video distinte - a seconda dell'elaborazione di rete e del dispositivo stesso.

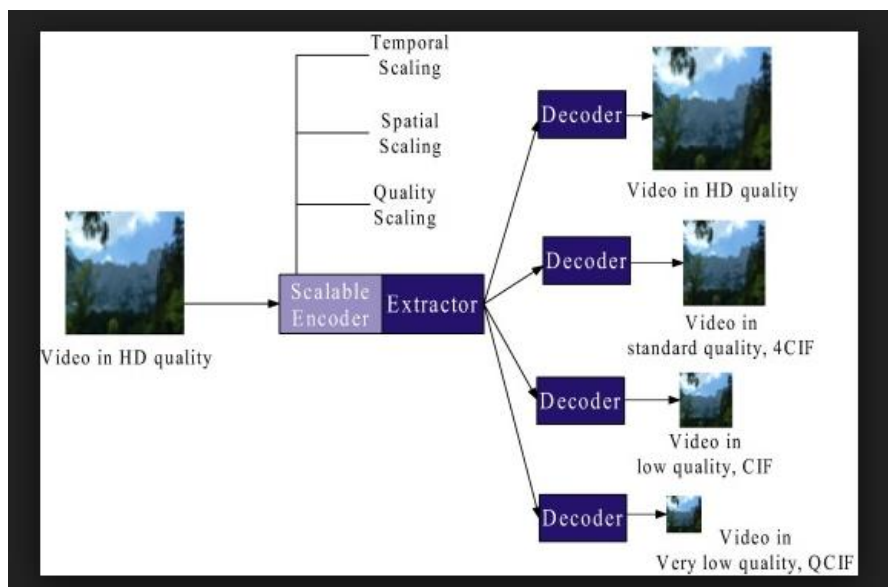


Figura 13. Trasmettitore che codifica la sequenza video in ingresso in 4 diversi strati complementari

Ref: [30]

La struttura a strati di qualsiasi contenuto video scalabile può essere definita come la combinazione di uno strato di base e diversi livelli di miglioramento aggiuntivi. Lo **strato di base** (Base Layer) corrisponde alla più bassa prestazione video supportata, mentre gli strati di miglioramento consentono il perfezionamento del suddetto. L'adattamento si basa su una combinazione all'interno dell'insieme di strategie selezionate per la scalabilità **spaziale, temporale e qualitativa** [10]. Negli ultimi anni, diversi specifici profili video scalabili sono stati inclusi in codec video come MPEG-2 (MPEG-2 Video, 2000), H.263 (H.263 ITU-T Rec., 2000) e MPEG-4 Visual (MPEG-4 Visual, 2004). Nella Figura 14 si può notare la caratteristica struttura di un flusso di dati decomposto in più strati.

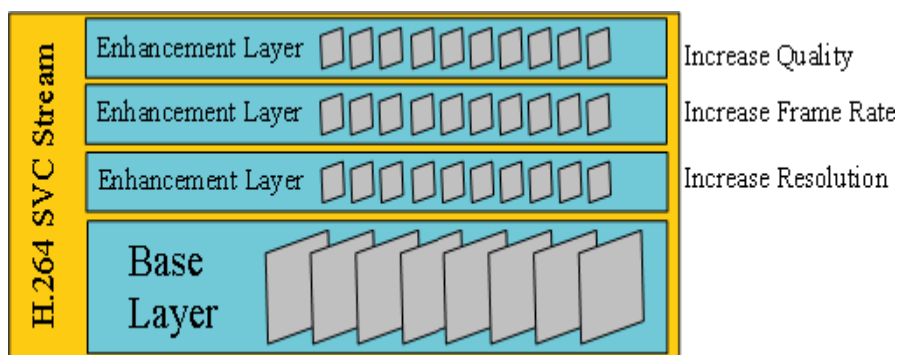


Figura 14. Possibile struttura di decodifica flusso dati

Ref: [https://www.eetimes.com/document.asp?doc\\_id=1275532](https://www.eetimes.com/document.asp?doc_id=1275532)

Nell'ottobre 2007, l'estensione scalabile del codec H.264, nota anche come H.264/SVC (Scalable Video Coding) (H.264/SVC, 2010), è stata standardizzata congiuntamente da ITU-T VCEG e ISO MPEG come emendamento allo standard H.264/AVC (Advanced Video Coding) (quindi una estensione al già' presente codec). Tra le numerose caratteristiche innovative, H.264/SVC combina scalabilità temporale, spaziale e qualitativa in un unico flusso multistrato [11].

Per esemplificare la scalabilità temporale, la Figura 15(a) presenta uno scenario semplice in cui la base è costituita da un sottogruppo di fotogrammi e dal livello di miglioramento di un altro. Un ipotetico ricevitore in una rete a bassa larghezza di banda riceverebbe solo lo strato di base, producendo così un video a scatti (15 fotogrammi al secondo, di seguito denominato fps). Al contrario, il secondo ricevitore (che beneficerebbe di una rete con una maggiore larghezza di banda) sarebbe in grado di elaborare e combinare entrambi gli strati, ottenendo così un video full-frame-rate (30 fps) e, infine, una riproduzione video più fluida.

Successivamente, la Figura 15(b) illustra un esempio di scalabilità spaziale, dove l'inclusione di strati di miglioramento aumenta la risoluzione del campione video decodificato. Come mostrato, più strati sono messi a disposizione del ricevitore, più alta è la risoluzione del video decodificato.

Infine, la Figura 15(c) mostra il concetto di scalabilità della qualità, dove gli strati di miglioramento aumentano la qualità del video ricevuto. Ancora una volta, più strati acquisisce il ricevitore, migliore è la qualità dell'esperienza dell'utente. Oltre ai benefici delle scalabilità sopra introdotte, ci sono molti altri vantaggi forniti da H.264/SVC. Una di queste notevoli caratteristiche è il supporto per l'adattamento del bit rate video a livello NAL (Network Application Layer), che aumenta significativamente la flessibilità del codificatore video. Soluzioni alternative scalabili, tuttavia, supportano l'adattamento solo a frame interi o parziali [9]. Inoltre, H.264/SVC migliora l'efficienza di compressione incorporando un meccanismo avanzato e innovativo per la stima interstrato, chiamato ILP (Inter-Layer Prediction) che riutilizza i vettori di movimento interstrato, le informazioni sulla struttura interna e i residui tra layer successivi [12].

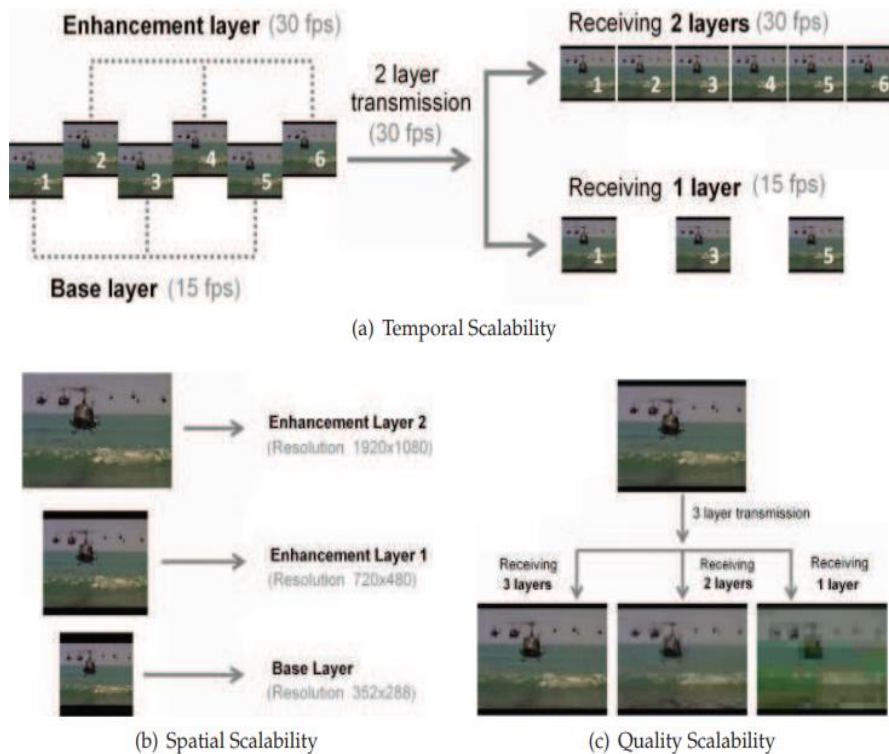


Figura 15. Le tre grandezze che rendono SVC un protocollo adattivo

Ref: [31]

Come conseguenza di tutti questi aspetti, lo standard H.264/SVC è attualmente considerato lo stato dell'arte dei codec video scalabili. A differenza dei precedenti codec video, H.264/SVC è stato progettato come un codec video scalabile flessibile e potente, che fornisce, per un dato livello di qualità, rapporti di compressione simili ad una complessità di decodifica inferiore rispetto alle sue controparti monostrato non scalabili.

Per confermare questo principio di progettazione, confrontiamo brevemente H.264/SVC con i profili non scalabili dei codec precedenti, ossia MPEG-4 Visual (MPEG 4 Visual, 2004), H.263 (H.263 ITU-T Rec., 2000) e H.264/AVC (H.264/AVC, 2010). Le prestazioni di Codec sono state analizzate sia in termini di efficienza di compressione che di qualità video (concentrandosi sul rapporto segnale/rumore PSNR(Peak signal-to-noise ratio)).

Figura 16(a)

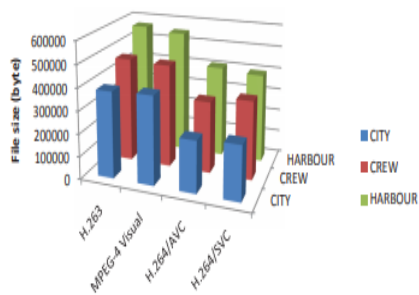
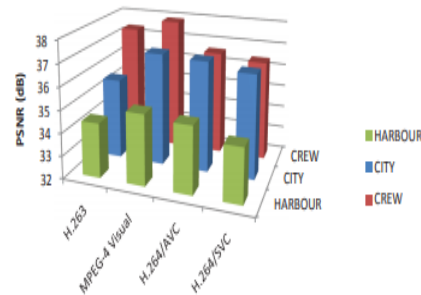


Figura 16(b)



**Differenza tra le dimensioni dei file che ogni  
architettura codifica**  
Ref: [31]

Come mostrato nella Figura 16(a), la dimensione reale del file codificato è diversa per ogni codec, anche se è stato impostato lo stesso bitrate di codifica. La ragione di questa difformità risiede nelle prestazioni delle implementazioni dei codec, che regolano liberamente il processo di codifica al bitrate specificato. Dalle Figure 16(a) e 16(b), è chiaro che H.264/SVC e H.264/AVC sono i codec che generano la dimensione più bassa del file raggiungono una qualità simile (es. 36.61 dB per H.264/AVC e 36.41 dB per H.264/SVC per la sequenza video CREW). Sulla base di queste simulazioni, si conclude che H.264/SVC supera i precedenti approcci non scalabili, supportando tre tipi di scalabilità ad alta efficienza di codifica. Questi risultati non solo valutano il comportamento teorico di ogni codec analizzato, ma dimostrano le eccezionali prestazioni di H.264/SVC rispetto ad altri approcci di codifica se applicato su un dato campione video.

Avendo in mente la definizione di SVC, ovvero:

‘Una architettura che permette lo streaming dinamico su HTTP il quale concetto sta nel codificare il file multimediale con una Base Layer dove essa potrà’ essere proiettata su la maggior parte dei dispositivi e poi a seconda della grandezza dell’immagine, della banda e della risoluzione, questa Base Layer può’ essere “estesa” con Layer aggiuntivi per avere un miglioramento qualitativo, spaziale e risolutivo.’

Andiamo a discutere nel dettaglio i concetti di **Scalabilità Temporale, Spaziale e Qualitativa**.

## 4.2 Scalabilità Temporale

Il termine "scalabilità temporale" si riferisce alla capacità di rappresentare i contenuti video con frame rate diversi. I flussi video codificati possono essere composti da tre diversi tipi di fotogrammi:

1. I (intra),
2. P (predittivo) o
3. B (Bi-predittivo).

Gli I-Frame esplorano solo la codifica spaziale all'interno dell'immagine, cioè le tecniche di compressione sono applicate alle informazioni contenute solo all'interno dell'immagine corrente, non usando riferimenti ad altre immagini. Proprio per questo motivo, gli Intra-Frame richiedono più spazio per essere codificati. Non avendo informazioni su frame precedenti o successivi, l'encoder deve comprimere l'intero frame prima di poter proseguire con la codifica.

Al contrario, entrambe i P-Frame e i B-Frame hanno interrelazioni con immagini diverse, in quanto esplorano direttamente le dipendenze tra loro. Mentre nei fotogrammi P la codifica predittiva viene eseguita sulla base di (almeno) un'immagine di riferimento precedente, i B-Frame consistono in una combinazione di codifica bi-predittiva, cioè per la predizione vengono presi in considerazione campioni di immagini di riferimento multiplo.

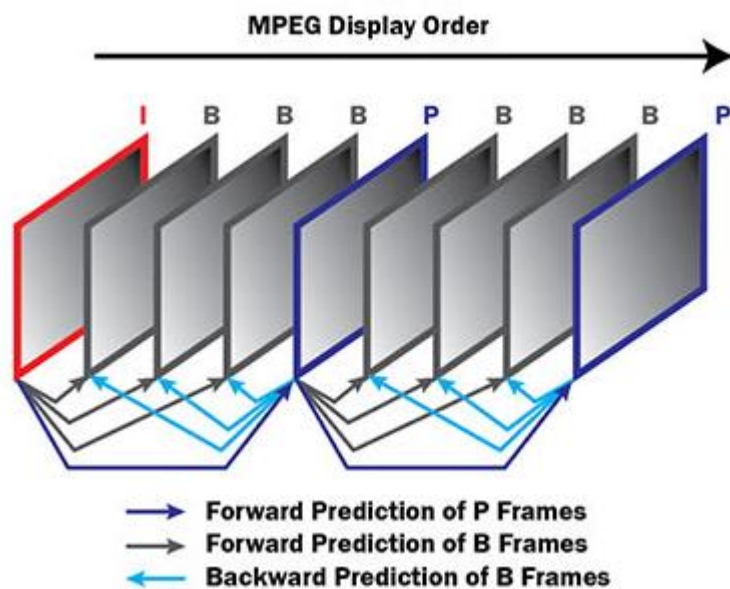


Figura 17. Funzionamento dei meccanismi predittivi

Ref: <https://www.rgb.com/streaming-video-and-compression>

Nella Figura 17 viene dimostrato come viene codificato un video tenendo presente la efficienza e la dimensione del prodotto finale. Da notare che i fotogrammi tra 2 Intra Frame (Key Frames) costituiscono il GOP (Group Of Pictures) il quale ci può svelare la qualità del flusso.

Inoltre, lo standard H.264 richiede che il primo fotogramma sia un'unità di accesso IDR (Instantaneous Decoding Refresh), che corrisponde all'unione di un fotogramma I con diverse informazioni critiche non correlate ai dati (ad esempio l'insieme dei parametri di codifica). In generale, la struttura **GOP** (Group Of Pictures) specifica la disposizione di questi fotogrammi all'interno di una sequenza video codificata. Certamente, la singolare dipendenza e le caratteristiche predittive di ogni tipo di fotogramma implicano caratteristiche del flusso video codificato divergenti.

Negli standard scalabili precedenti (ad es. MPEG-2, H.263 e MPEG-4 Visual), la scalabilità temporale veniva fondamentalmente eseguita segmentando i livelli in base ai diversi tipi di frame. Ad esempio, un video composto da un formato tradizionale "IBBP" (un I frame seguito da due frame B e un frame P) veniva utilizzato per costruire tre livelli temporali: lo

strato base (L0) con I frame, il primo strato di miglioramento (L1 ) con frame P e il secondo strato di miglioramento (L2) con frame B.

Questo approccio diadico (formato di decomposizione 2:1) si è dimostrato funzionale, anche se comporta uno squilibrio con l'uso della banda (cioè il bit rate totale richiesto dai frame I è significativamente più grande di quello dei frame P e B [11].

Al contrario, in H.264/SVC, la base della scalabilità temporale si trova nella struttura GOP, poiché divide ogni frame in livelli di scalabilità distinti (combinando congiuntamente i tipi di frame I, P e B).

Per quanto riguarda il codec, la definizione GOP può essere riformulata come *la disposizione dei frame del bitstream codificato tra due immagini successive dello strato di base temporale* [13]. E' importante ricordare che i fotogrammi dello strato di base temporale non devono essere necessariamente un I frame. In realtà, solo la prima immagine di un flusso video è strettamente obbligata ad essere codificata come I frame e ad essere inclusa nell'IDR.

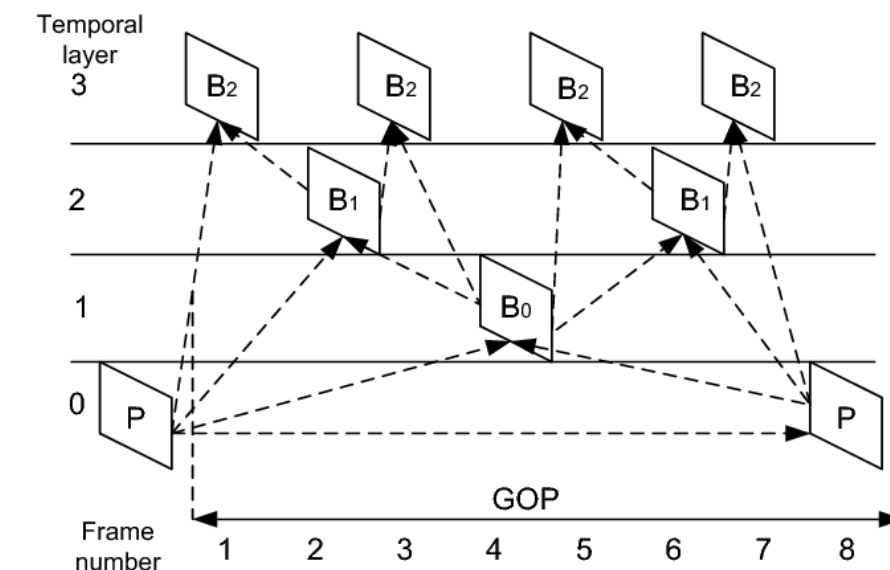


Figure 18(a) Decomposizione gerarchica la quale alla fine restituisce un GOP di 8 fotogrammi

Ref: [13]



Al fine di aumentare la flessibilità del codec, lo standard H.264/SVC definisce una struttura distinta per la previsione temporale, dove i fotogrammi di riferimento per ogni sequenza video sono riorganizzati in uno schema gerarchico ad albero. Questo schema ad albero migliora la distribuzione delle informazioni tra fotogrammi consecutivi e consente sia una scalabilità temporale diadica che non. La Figura 18(a) esemplifica questa decomposizione temporale gerarchica per una relazione di frame rate 2:1 in un video codificato a quattro strati. In questo esempio, lo strato di base L0, che è costituito da fotogrammi I o P, permette di ricostruire una immagine per GOP.

Il primo strato di valorizzazione L1, solitamente composto da B Frame, estrae un'immagine aggiuntiva per il GOP in aggiunta a quella di L0.

Il secondo strato di valorizzazione L2, composto da B Frame, estrae ulteriormente due immagini aggiuntive per GOP insieme a quelle degli strati precedenti. Infine, il terzo strato di miglioramento L3 permette di recuperare otto immagini.

Inoltre, H.264/SVC suggerisce l'inclusione di un filtro di pre-elaborazione prima del modulo di previsione del movimento, che può migliorare la distribuzione delle informazioni sui dati ed eliminare le ridondanze tra livelli consecutivi. L'algoritmo proposto è denominato **MCTF**. Questo filtro aggiuntivo, se applicato sui dati originali, esegue l'elaborazione della decomposizione allineata al movimento e ha come risultato, la correlazione tra i livelli filtrati migliorata, mentre aumenta la complessità complessiva del codificatore [14].

### 4.2.1 Valutazione di Analisi

La struttura dei frame imposta al GOP (Group of Pictures) è essenziale non solo per la scalabilità temporale offerta da questo codec scalabile, ma anche per le caratteristiche del flusso video risultante. Infatti, la modifica della dimensione del GOP influisce direttamente sul numero di strati temporali contenuti nel flusso di bit codificato.

Ad esempio, in un approccio temporale diadico, un flusso video codificato con dimensione GOP pari a 16 genera cinque strati temporali:

**T0** (1 fotogramma per GOP),  
**T1** (2 fotogrammi per GOP),  
**T2** (4 fotogrammi per GOP),  
**T3** (8 fotogrammi per GOP) e  
**T4** (16 fotogrammi per GOP).

Tuttavia, codificando lo stesso video con dimensione pari a 8 rende quattro strati temporali:

**T0** (1 fotogramma per GOP),  
**T1** (2 fotogrammi per GOP),  
**T2** (4 fotogrammi per GOP) e  
**T3** (8 fotogrammi per GOP).

Infine, la definizione di una dimensione GOP di 4 produce solo tre strati temporali:

**T0** (1 fotogramma per GOP),  
**T1** (2 fotogrammi per GOP) e  
**T2**. (4 fotogrammi per GOP)

Si può quindi concludere che la flessibilità di una soluzione scalabile nel tempo (in termini di numero di strati) è direttamente proporzionale al numero di strati per GOP. Tuttavia, l'aumento della dimensione del GOP ha alcuni effetti collaterali impliciti: influenza l'efficienza complessiva della codifica, in quanto impone una variazione del numero di fotogrammi I, P e B. Per dimostrare questo effetto, sono stati valutati diversi esperimenti cambiando il parametro GOP mentre il bitrate di uscita viene mantenuto costante. La Figura 19 mostra i risultati ottenuti in termini di qualità per lo strato superiore e di base.

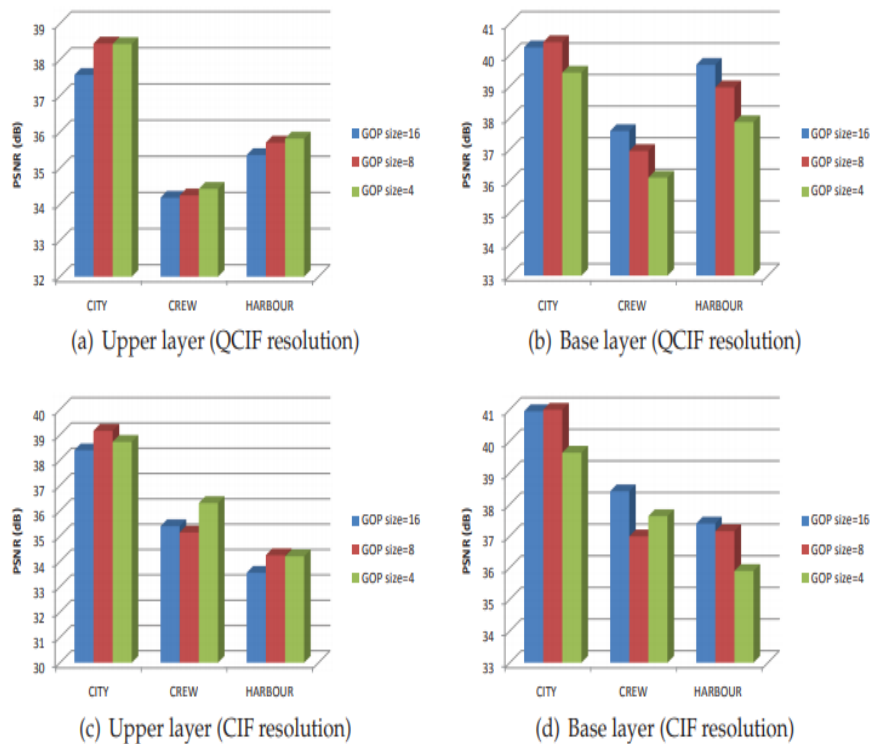


Figura 19. Qualità del flusso decodificato con settaggi diversi del GOP

Ref: [31]

Osservando le Figure 19(a) e 19(c) si può notare che non c'è alcuna differenza significativa di qualità nel video finale recuperato quando si aumenta la dimensione del GOP. Tuttavia, il comportamento della qualità dello strato di base varia leggermente a seconda sia dei campioni video particolarmente utilizzati che delle risoluzioni selezionate, come si può vedere nelle Figure 19(b) e 19(d). Un incremento della dimensione GOP comporta un incremento della qualità dello strato di base per le sequenze video CREW-QCIF, HARBOUR-QCIF e HARBOUR-CIF mentre, per esempio, tale relazione, nel campione video CREW-CIF, non è così evidente. Questa variabilità delle prestazioni qualitative può essere in parte indotta dalle particolarità del modulo di previsione scalabile (H.264/SVC ILP). Teoricamente, un incremento della dimensione del GOP dovrebbe implicare un miglioramento della qualità, in quanto il numero di fotogrammi B aumenta contribuendo al contempo ad una codifica efficiente.

Al contrario, la complessità dell'encoder è chiaramente influenzata dal parametro GOP size, cioè l'aumento del numero di strati (e quindi dei B Frame) implica requisiti più elevati per il modulo di previsione. Un tale aumento della complessità di codifica (misurato in termini di tempo di esecuzione della codifica), è rappresentato nella Figura 20. Ad esempio, un incremento di circa il 20% del tempo di codifica si ottiene confrontando le dimensioni GOP di 4 e 16 per la sequenza video CITY con risoluzione QCIF.

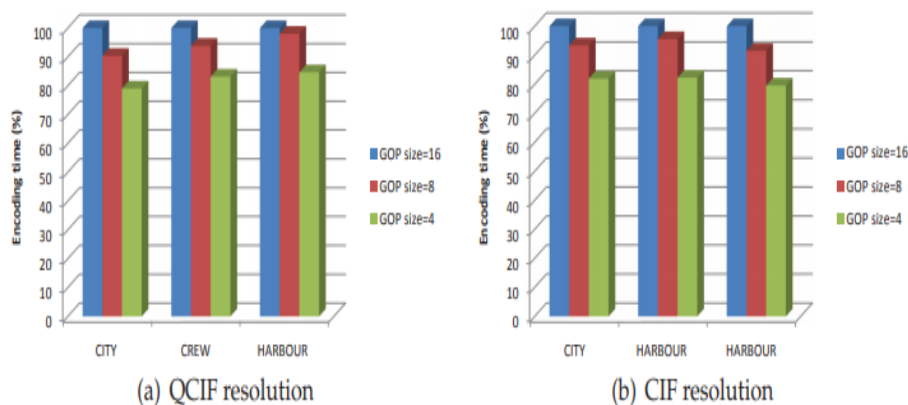


Figura 20. Tempo di Codifica relazionato alla grandezza del GOP

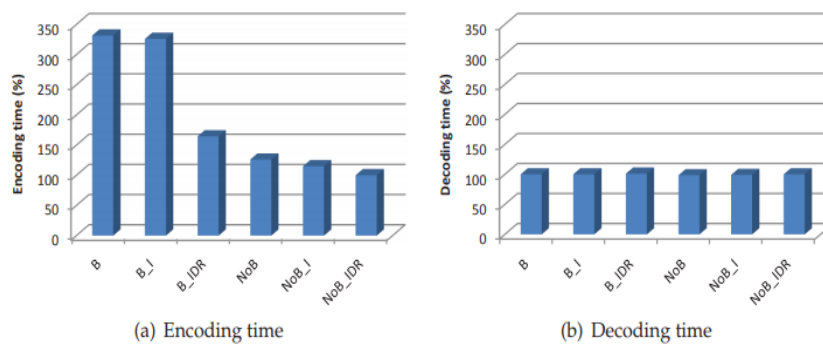
Ref: [31]

È inoltre interessante analizzare i vantaggi dell'utilizzo di dimensioni GOP più elevate per la scalabilità temporale, in quanto un incremento della dimensione GOP aumenta il numero di livelli temporali disponibili e, in ultima analisi, aumenta la flessibilità del flusso video. Come già menzionato nella Sezione 4.2, tre tipi di fotogrammi sono generalmente considerati per codificare un'immagine video: I, P e B frame.

La differenza tra questi tipi di fotogrammi risiede principalmente nei riferimenti utilizzati per la codifica predittiva. Certamente, la singolare dipendenza e le caratteristiche predittive di ogni tipo di fotogramma portano a caratteristiche del flusso video codificato divergenti. Inoltre, la disposizione dei fotogrammi all'interno di un GOP ha un impatto diretto sulle prestazioni del codec. In questo contesto, la Figura 21 mostra come diverse strutture GOP influenzano la complessità della codifica e della decodifica, pur mantenendo una qualità video simile.

**Definizione:**

- **B**: un frame P iniziale e 15 frame B consecutivi formano la struttura GOP.
- **B\_I**: il GOP è composto da un I frame iniziale e 15 frame B consecutivi.
- **B\_IDR**: il GOP corrisponde ad un frame IDR iniziale, seguito da 15 B frame.
- **NoB**: solo i frame P (16) in tutto il GOP.
- **NoB\_I**: il GOP è composto da un I frame iniziale, seguito da 15 frame P.
- **NoB\_IDR**: un frame IDR iniziale seguito da 15 fotogrammi che formano la struttura GOP.



**Figura 21. Tempi di Codifica / Decodifica in base al GOP**

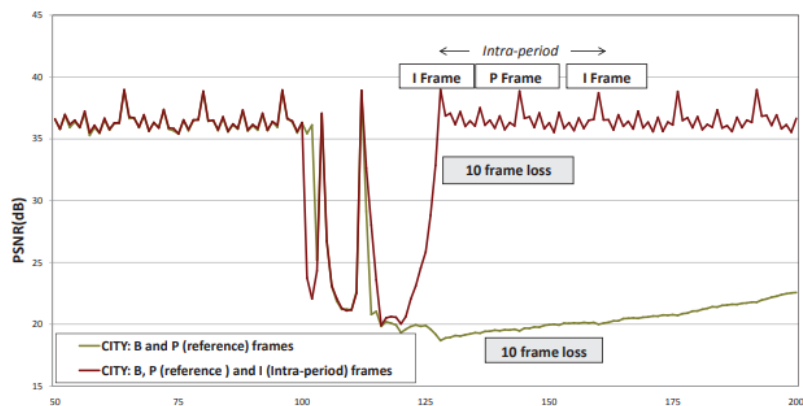
**Ref: [31]**

Questo esperimento sottolinea l'influenza dei frame B all'interno di un GOP, poiché impongono un significativo aumento della complessità di codifica. Tuttavia, la loro inclusione non fornisce alcun vantaggio comparabile, in quanto la qualità rimane quasi uguale. Per quanto riguarda l'influenza dei Frame I e dei IDR ulteriori test indicano che la qualità, la complessità e il comportamento del bit rate sono simili.

Oltre alle implicazioni sul bitrate video, la determinazione della frequenza di key frames ha un ruolo importante anche quando si tratta di perdite di pacchetti in applicazioni di streaming video reali, che possono essere dovute a fenomeni diversi, come ad esempio congestione, perdite di comunicazione wireless o handover [16]. Come esemplificato nella Figura 22, il recupero della qualità video è direttamente influenzato dalla struttura

del GOP e, in particolare, dal ricevimento di key frames. A causa delle caratteristiche dei fotogrammi I, prima si riceve uno di essi prima si recupera la qualità video.

Sulla base di questa logica e facendo riferimento all'esempio riportato, il recupero della qualità video per H.264/SVC quando sono presenti key frames è molto più veloce (linea marrone nella figura 22) di quella corrispondente a non-key frames. (linea verde nella figura 22). E' importante notare che con la ricezione di un frame intra-type, la qualità del video ricevuto viene recuperata quasi immediatamente, mentre le dipendenze intrinseche dei frame P e B comportano un recupero di qualità più lento in caso di perdite. In altre parole, a causa dell'uso di una struttura di codifica predittiva, una perdita di frame non solo influisce sull'attuale GOP, ma può avere un impatto anche sui quelli precedenti e successivi.



**Figura 22. Perdita e Recupero di un Key-Frame**

Ref: [31]

Tuttavia, e in aggiunta al fatto che i key frame forniscono una risincronizzazione dei frame rapida, la velocità di recupero di qualità della sequenza video non dipende solo dal GOP ma anche dalle sue particolari caratteristiche. Infatti, per quanto siano simili (ad esempio, movimento semistatico nella sequenza CITY), i fotogrammi P e B codificati forniscono poche informazioni l'uno rispetto all'altro. Pertanto, in questi tipi di sequenze di movimento, è difficile riprendersi dalla perdita dei fotogrammi precedenti, a meno che non siano inclusi gli intraframe [16].

Di conseguenza, si ritiene fondamentale determinare con attenzione la frequenza di questo tipo di fotogrammi - siano I o IDR - che rappresenta un compromesso tra le dimensioni del file e la velocità di recupero: una frequenza di inclusione più alta accelera il recupero di qualità video in ambienti lossy con una penalità nella dimensione del file. In sintesi, il problema su a cosa dare priorità, ovvero, al bit rate dell'ambiente o alla velocità di recupero della qualità video, è una decisione da prendere in base al rispettivo scenario. Allo stesso modo, la selezione tra i fotogrammi I e IDR (o qualsiasi combinazione di entrambi) dovrebbe essere lasciata aperta ad ogni particolare applicazione.

### **4.3 Scalabilità Spaziale**

Con il termine si va ad identificare il processo con il quale i bit stream vengono codificati in modo tale da rappresentare una risoluzione specifica. In origine, questa codifica veniva fatta indipendentemente dalle varie layer. Quindi non si prendeva in considerazione quanto prodotto dalla Base Layer se il bitstream veniva codificato ad una risoluzione maggiore. Tutto andava fatto da capo. Questo processo però essendo molto costoso in termini di risorse, e' stato rivisitato.

Sebbene H.264/SVC contempli anche rapporti non diadici, la configurazione più comune adottata e' la quella di 2:1 [15]. Gli approcci dei precedenti encoder scalabili consistono essenzialmente nel riutilizzare le informazioni di previsione del movimento dagli strati inferiori per ridurre le dimensioni del flusso globale. Purtroppo, la qualità dell'immagine ottenuta con questa metodologia è piuttosto limitata. Per migliorare la sua efficienza, l'encoder H.264/SVC introduce un modulo di previsione più flessibile e complesso chiamato Inter-Layer Prediction (ILP) il quale ha lo scopo di massimizzare il riuso di parti già codificate in precedenza: tali parti verranno ovviamente adattate alla risoluzione rispettiva, così che il processo di codifica risulti più efficiente e anche meno costoso.

## In cosa consiste ILP:

Ci sono 3 metodi per raggiungere l'obiettivo di efficienza:

**-Inter-Layer Motion Prediction:** i vettori di movimento degli strati inferiori possono essere utilizzati dagli strati superiori di miglioramento. In alcuni casi, i vettori di movimento e le informazioni ad essi collegate devono essere ridimensionati in modo da adeguare i valori agli strati superiori [12].

- **Inter-Layer Intra Texture Prediction:** h.264/SVC supporta la previsione della texture per i blocchi dello stesso strato di riferimento. Il blocco può essere utilizzato per altri blocchi negli strati superiori. Questo modulo sovra-campiona la risoluzione della texture dello strato inferiore fino alla risoluzione dello strato superiore, calcolando successivamente la differenza tra di loro. In altre parole, se due o più intra-frame hanno molte cose in comune, invece di codificare il nuovo frame, si elabora così che possa essere adattato alla nuova risoluzione.

- **Inter-Layer Residual Prediction:** come conseguenza di diverse osservazioni del processo di codifica, è stato identificato che quando due strati consecutivi hanno informazioni di movimento simili, ci sono ridondanze. Su questa base, in H.264/SVC, il metodo di previsione dei residui interstrato può essere utilizzato dopo il processo di compensazione del movimento per esplorare le ridondanze nel dominio spaziale residuo. In sintesi, se in due fotogrammi il background rimane invariato (esempio: una persona che cammina), avremo delle ridondanze. Con la residual prediction si può migliorare la compressione proprio perché vengono sfruttate quest'ultime.



### 4.3.1 Valutazione di Analisi

Con la scalabilità spaziale, i livelli all'interno dello stesso flusso video codificato contengono distinte risoluzioni video. Per supportare questa scalabilità, il movimento, la texture e le informazioni residue da i livelli precedenti (dopo il ridimensionamento alla nuova risoluzione), possono essere riutilizzati dall'encoder H.264/SVC. Quando la relazione tra i layer è di 2:1, l'algoritmo di ridimensionamento è piuttosto semplice, poiché in questo caso l'operazione di ridimensionamento di un layer si riduce ad una semplice operazione di bit-shift. Tuttavia, H.264/SVC supporta anche qualsiasi altro rapporto di risoluzione tra i livelli successivi per i quali sono necessarie operazioni matematiche più complesse.

Per determinare i reali requisiti della codifica di scalabilità spaziale di H.264/SVC, diversi esperimenti pratici sono stati eseguiti variando i rapporti di risoluzione tra strati. Nel primo caso, sono stati utilizzati un livello di risoluzione QCIF di base e un livello di miglioramento della risoluzione CIF (scenario diadico) vengono utilizzati. Nel secondo esperimento, lo strato di miglioramento è regolato a 240 x 112 pixel, mantenendo lo stesso livello di base.

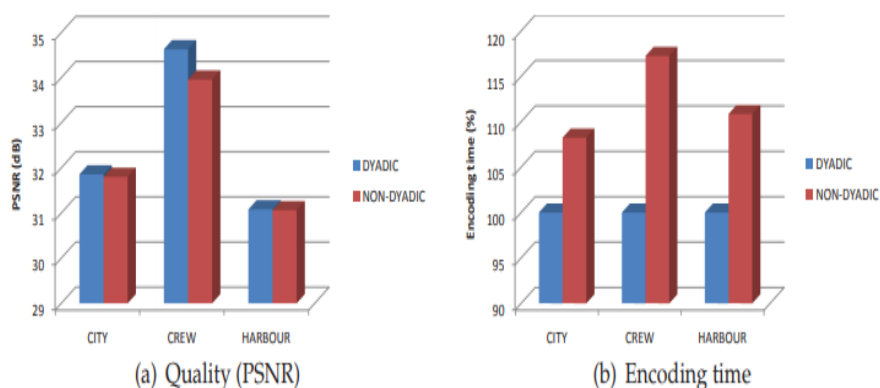


Figura 23. Approccio diadico relazionato alla Qualita' e al tempo di codifica

Ref: [31]

Da un lato, la Figura 23(a) mostra il confronto di qualità per entrambi gli esperimenti, dove si può notare un grado di qualità leggermente superiore per lo scenario diadico. Questo fenomeno è spiegato notando che una relazione 2:1 non produce alcuna distorsione di ridimensionamento e che non tiene in considerazione i fattori per i rapporti di risoluzione non integrali. D'altro canto, quando si affrontano casi non diadici, la complessità dell'encoder aumenta significativamente, come mostrato nella Figura 23(b). In altre parole, configurazioni diadiche possono essere elaborate con un tempo di codifica notevolmente inferiore a quello delle configurazioni non diadiche. L'approccio non diadico aumenta il carico di codifica fino a circa il 18% per la sequenza video di CREW.

## 4.4 Scalabilità Qualitativa

Qui il bit stream viene codificato ad una singola risoluzione spaziale ma a diverse qualità, ovvero si parte da una base (BASE LAYER) che vale per tutti i dispositivi e dipendendo dalla qualità della linea, si possono aggiungere delle migliorazioni (Enhancement Layers). Lo standard H.264/SVC supporta tre distinte scalabilità SNR:

- **Coarse Grain Scalability (CGS):** in questa architettura (Figura (a)), ogni strato ha un'azione di procedura di previsione indipendente (tutti i riferimenti hanno lo stesso livello di qualità). CGS può essere visto come un caso particolare di scalabilità spaziale, in quanto vengono utilizzati meccanismi di codifica simili, ma la risoluzione spaziale è costante. Più specificamente, in modo simile alla scalabilità spaziale, CGS implementa meccanismi di previsione interstrato (ILP), come la previsione delle modalità di macroblocco e la previsione del segnale residuo. Il CGS si differenzia dalla scalabilità spaziale in quanto le operazioni di sovracampionamento non sono performati. Inoltre, quando viene utilizzato ILP per questo tipo di scalabilità, i dati vengono ridimensionati con un parametro QP minore a quelli degli strati precedenti. In realtà, questo concetto supporta bitrate specifici e quindi non è adatto a tutti i dispositivi. Una peculiarità di CGS è che non è possibile fare lo switch di layer casualmente. Quest'ultimo si può fare solo in determinati punti pre-

impostati. Quindi, per incrementare la adattabilità e il processo di codifica, viene introdotta una variazione del CGS chiamata MGS.

- **Medium Grain Scalability (MGS):** l'approccio MGS (Figura (b)) aumenta l'efficienza più flessibile, dove entrambi i tipi di strato (base e miglioramento) possono essere referenziati. Questo permette di fare lo switch tra layer in qualsiasi momento senza essere limitati da premesse. La differenza più significativa con GSC è che qualunque strato di miglioramento può essere scambiato garantendo così la scalabilità desiderata. Tuttavia, questa strategia può indurre un effetto di deriva (cioè può introdurre un de-sincronismo tra encoder e decoder) se viene ricevuto solo lo strato di base (si avranno dei fotogrammi nulli). Per risolvere questo problema, la specifica MGS propone l'utilizzo di immagini chiave periodiche (Key Frames), che risincronizzano immediatamente il modulo di previsione.

- **Fine Grain Scalability (FGS):** questa versione (Figure (c)) della scalabilità SNR si prefigge i seguenti obiettivi:

- 1.) fornire un adattamento continuo del bit rate di uscita in relazione alla reale larghezza di banda della rete.
- 2.) Impiegare una tecnica avanzata di bitplanning in cui diversi tipi di livelli sono responsabili del trasporto di sottoinsiemi distinti di bit corrispondenti a ciascun dato.

Lo schema consente di troncare i dati in qualsiasi punto arbitrario al fine di supportare il progressivo affinamento dei coefficienti di trasformazione. In questo tipo di scalabilità, solo lo strato di base lancia le tecniche di previsione del movimento. Questo implica una grande flessibilità ma essendo molto tassativo in termini di compressione, FGS è stato rimosso (per il momento) dalla definizione di SVC.

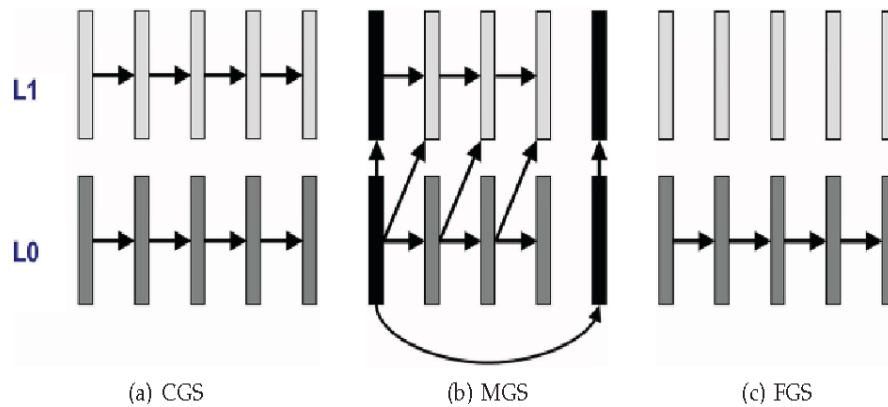


Figure (a). L'indipendenza degli strati non permette gli switch casuali. Essi si possono fare solo tramite determinati punti

Figure (b). Variazione di CGS che permette switch casuali e che garantisce maggiore flessibilita' chiamata MGS

Figure (c). Il metodo piu' complesso tra i tre menzionati che permette la scalabilita' tramite il bitplanning

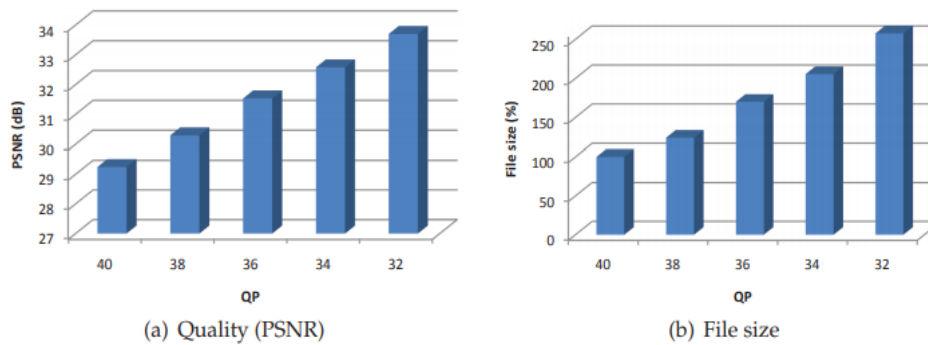
Ref: [31]

#### 4.4.1 Valutazione di analisi

La scalabilità SNR (o scalabilità qualitativa) implica diverse tecniche per creare livelli di qualità diversi all'interno dello stesso flusso di bit codificato. A questo proposito, JSVM fornisce diverse opzioni per specificare la qualità desiderata non solo per ogni particolare livello, ma anche per l'intero flusso.

In primo luogo, questa sottosezione si concentra sul cosiddetto Parametro di Quantizzazione (QP), che è direttamente correlato al processo di quantizzazione della sequenza video originale. Successivamente, vengono analizzate le proprietà specifiche di due distinte modalità di scalabilità SNR di H.264/SVC, ovvero CGS e MGS. La modalità FGS non è stata inclusa in questi esperimenti in quanto, non consente la configurazione personale di

parametri rilevanti, come il numero di strati o il valore del passo di quantizzazione per strato.



**Figura 24. Maggiore e' il QP, maggiore risultano PSNR e Dimensione del file**

Ref: [31]

In generale, valori più bassi dei parametri di quantizzazione portano sia ad un migliore livello di PSNR che ad un bitrate più alto. Tuttavia, durante il processo di codifica, il valore QP non e' mantenuto esattamente uguale per tutti i fotogrammi all'interno del flusso dato, ma varia leggermente a seconda della posizione di ciascun Frame all'interno del GOP.

Il valore QP appropriato per ogni scenario o applicazione multimediale dovrebbe essere selezionato non solo tenendo conto della qualità desiderata, ma anche analizzando l'impatto pratico del QP sulla dimensione del file del bitstream codificato. Da un lato, la Figura 24 attesta la relazione diretta tra il parametro di quantizzazione selezionato, la qualità video e le dimensioni del file. La Figura 25 invece rappresenta la differenza nella qualità visiva che si verifica quando si assegnano diversi valori di QP al processo di codifica del campione video CREW.



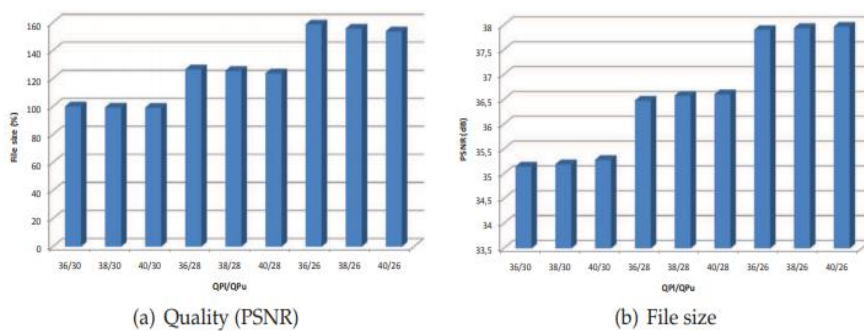
**Figura 25. Qualita' alterata con parametro QP alternato**

Ref: [31]

Una volta esplorata l'influenza del parametro QP, viene effettuata un'analisi più approfondita valutando la scalabilità della qualità intrinsecamente fornita.

Nel seguente test, due strati scalabili SNR sono incorporati nel flusso codificato (qualità inferiore per lo strato inferiore  $QP_L$ , e migliore qualità per lo strato superiore,  $QP_U$ ), poiché con JSVM è possibile assegnare un valore QP indipendente ad ogni strato scalabile.

Una delle basi di H.264/SVC è la capacità di beneficiare dei suoi meccanismi di previsione interstrato in modo da eseguire una codifica scalabile efficiente. Tuttavia, c'è una stretta dipendenza tra le scalabilità della qualità selezionata e la previsione interstrato nel flusso video risultante, come l'esperimento nella figura 26 mostra chiaramente.



**Figura 26. Dipendenza tra le scalabilità della qualità selezionata e la previsione interstrato nel flusso video risultante.**

Ref: [31]

In questo esempio, la qualità ottenuta negli strati superiori (definiti da  $QP_U$ ) dipende certamente dalla qualità degli strati inferiori come specificato da  $QP_L$ . Facendo riferimento alla Figura 26(a), anche se il  $QP_U$  impostato è lo stesso, la qualità video risultante è leggermente diversa in base alla qualità dello strato inferiore sottostante. La ragione di questo fenomeno gravita sul meccanismo di previsione interstrato: poiché gli strati di miglioramento affinano progressivamente la qualità degli strati inferiori, anche quando viene utilizzata lo stesso  $QP_U$ , il PSNR raggiunto dal

contenuto dipende approssimativamente dalla qualità degli strati inferiori, che è stabilita dal parametro QPL.

Ulteriori esperimenti sono stati condotti per analizzare le caratteristiche specifiche delle distinte modalità di scalabilità SNR di H.264/SVC:

- CGS
- MGS

Per entrambi gli esperimenti è stata utilizzata la stessa configurazione per il parametro di quantizzazione:  $QP_L=39$  per lo strato di base e  $QP_U=33$  per lo strato di miglioramento. Inoltre, e per semplificare l'analisi, entrambe le modalità sono state costrette a produrre lo stesso bitrate in uscita. I risultati di questi esperimenti sono presentati nella Figura 27, sia per la qualità video che per le metriche delle prestazioni di codifica. Per tutte le sequenze video valutate, l'approccio MGS produce risultati di migliore qualità, come evidenziato nelle figure 27(a) e 27(b). Questo interessante risultato è dovuto alla maggiore flessibilità dell'algoritmo di previsione interna di MGS (in quanto sono supportati più riferimenti), che contribuisce a ridurre gli errori di corrispondenza (cioè i dati residui). D'altra parte, entrambe le modalità di scalabilità presentano risultati simili in termini di prestazioni del codec (tempo di esecuzione della codifica).

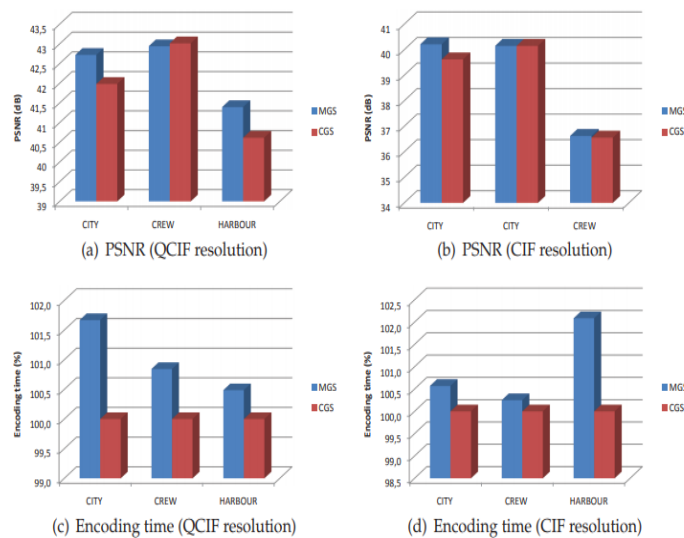


Figura 27. Risultati dei test con approccio MGS e CGS

Ref: [31]

## Capitolo 5

---

### Algoritmi usati da SVC

SVC, insieme alle sue scalabilità temporali, di qualità e spaziali differenziate, fornisce diverse altre caratteristiche innovative, che sono soggette a sperimentazione pratica attraverso questa sezione.

#### 5.1 Prediction Module

In generale, questo tipo di tecniche rappresentano quegli algoritmi che permettono di determinare i vettori che descrivono la correlazione tra due fotogrammi adiacenti in una sequenza video. In questo contesto, H.264/SVC permette di sintonizzare i parametri di ricerca del suo algoritmo di stima del movimento: è possibile decidere se utilizzare un algoritmo esaustivo di block-searching o un approccio ottimizzato per la velocità. Inoltre, è possibile modificare anche la gamma di ricerca della funzione di ricerca a blocchi.

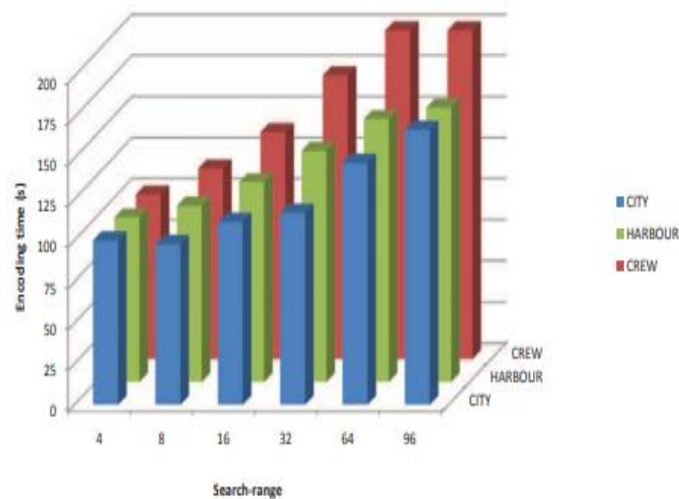
Tuttavia, l'esaustiva funzione di ricerca dei blocchi richiede un'elevata complessità computazionale nel processo di codifica, mentre la sua ripercussione sulla qualità e l'efficienza di codifica non è significativa. Queste affermazioni sono corroborate dai risultati degli esperimenti condotti, riportati nella Tabella 3. Si noti che questi risultati sono stati generati dalla codifica di sequenze video con risoluzione QCIF, dato che la complessità della codifica aumenta drasticamente con l'aumentare della complessità di codifica. Poiché la qualità della codifica video è paragonabile per entrambe le funzioni di ricerca, è altamente raccomandato selezionare l'algoritmo di ricerca rapida nei pratici encoder H.264/SVC.



Video sequence	Motion-search algorithm	Search-range	Decoding time (%)
CITY	Fast	Exhaustive	100%
CITY	Exhaustive	Exhaustive	6133,20%
CREW	Fast	Exhaustive	100%
CREW	Exhaustive	Exhaustive	3153,25%
HARBOUR	Fast	Exhaustive	100%
HARBOUR	Exhaustive	Exhaustive	6482,42%

**Tabella 3: Impatto dell'algoritmo di ricerca del movimento selezionato in H.264/SVC.**

Strettamente correlato alla compensazione del movimento, consentendo un'ulteriore compensazione di blocchi di 8x8, si può notoriamente aumentare la complessità dell'encoder. Come i risultati sperimentali nella Figura 29 certificano, abilitando ulteriori partizioni macroblocco di 8x8, richiede più risorse quando si codifica una data sequenza video, mentre ci sono pochi vantaggi nelle altre metriche considerate (dimensione e qualità del file).



**Figura 28. L'impatto dei parametri del campo di ricerca sulla codifica video H.264/SVC.**

Ref: [31]

### Conseguenza:

Per quanto riguarda i meccanismi di stima del movimento in H.264/SVC è altamente consigliato l'uso di algoritmi di ricerca rapida e piccoli campi di ricerca se l'applicazione di destinazione richiede di ridurre al minimo la complessità dell'encoder (Figura 28).

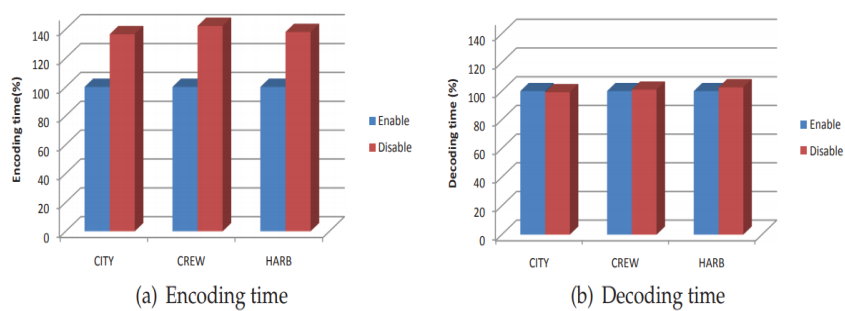


Figura 29. Impatto dell'attivazione di partizioni sub-macroblock 8 x 8.

Ref: [31]

## 5.2 Deblocking Filter

All'interno di questa sezione, vengono analizzati i benefici derivanti dall'applicazione dell'approccio con il deblocking filter nella codifica video H.264/SVC. I filtri vengono sfruttati nelle tecniche di codifica, applicandoli ai blocchi all'interno dei frame, il che porta a una migliore previsione in quanto riducono l'errore di decodifica tra i macroblocchi. Il filtro opera all'interno del ciclo di previsione di motion vectors, permettendo una migliore qualità per l'utente finale [13]. Un filtro di deblocking è un filtro video applicato al video compresso decodificato per migliorare la qualità visiva e le prestazioni di previsione, smussando gli 'spigoli' che possono formarsi tra macroblocchi quando si utilizzano tecniche di codifica a blocchi.

In questi esperimenti vengono valutati il filtro di deblocking in loop e il filtro di deblocking interstrato incluso nello standard H.264/SVC. A tal fine, nel software di riferimento JSVM sono stati considerati i seguenti casi:

- 1) non viene applicato alcun filtro (LF0);
- 2) il filtro viene applicato a tutti i bordi dei blocchi (LF1);
- 3) il filtraggio a due stadi dove i limiti delle sezioni sono filtrati nel secondo stadio (LF2); e, infine,
- 4) il filtraggio di deblocking a due stadi viene applicato al componente luma (black-and-white) (LF3).

La valutazione dei benefici e degli svantaggi di ciascuno dei casi di filtraggio sopra citati è stata effettuata, oltre alle metriche finora utilizzate (cioè tempo di codifica/decodifica, efficienza di codifica e PSNR), ricorrendo alla MSU Blocking Metric (MSU Video Quality Measurement Tool, 2010).

La MSU Blocking Metric misura l'effetto di blocco da fotogramma a fotogramma in una data sequenza video, rilevando i bordi degli oggetti con metodi euristici. **Un valore più alto del MSU Blocking Metric corrisponde ad una migliore qualità video.** Gli esperimenti per l'analisi del filtro di deblocking in-loop sono stati eseguiti su diverse sequenze video e configurazioni che combinano livelli temporali, spaziali e SNR.

La tabella 4 mostra i risultati degli esperimenti per un singolo strato spaziale (risoluzione QCIF) e due strati di qualità (un comportamento simile è stato ottenuto per altre combinazioni) [31].

Video Sequence	LF <sub>0</sub>	LF <sub>1</sub>	LF <sub>2</sub>	LF <sub>3</sub>
CITY	1222159	1175891	1175891	1174807
CREW	1051660	1356914	1356914	1362196
HARBOUR	1208833	1252369	1252369	1251459

**Tabella 4:** Impatto di selezionate tecniche di filtraggio di deblocking in loop sulle prestazioni di H.264/SVC (in termini di MSU Blocking Metric media).

Da questi test, si può trarre una conclusione interessante: le prestazioni del filtro di deblocking dipendono fortemente dalla specifica sequenza video e dalla combinazione dei strati scalabili. Da un lato, la qualità ottenuta applicando ciascuna delle tecniche di filtraggio testate, diverge sostanzialmente e dipende non solo dalla dinamica e dalle caratteristiche della sequenza video originale, ma anche dalla specifica combinazione di scalabilità nel processo di codifica H.264/SVC. D'altra parte, la complessità di codifica e decodifica di questi filtri mostra una chiara dipendenza da ogni sequenza video in ingresso.

Analogamente, il filtro di deblocking interstrato è stato valutato negli scenari sopra menzionati. La stessa analisi e procedura è stata fatta e, ancora una volta, i risultati ottenuti non sono stati conclusivi. Pertanto, la migliore tecnica di filtraggio non può essere determinata in anticipo e, per ogni applicazione multimediale o scenario, è necessario effettuare un'analisi approfondita per selezionare la tecnica di filtraggio del deblocking appropriata.

### 5.3 Pre-processing Filter

SVC supporta anche un filtro aggiuntivo di pre-elaborazione: l'opzione Motion-Compensated Temporal Filtering (MCTF).

Questo filtro è stato suggerito come una soluzione aggiuntiva per gestire meglio la somiglianza dei dati tra livelli consecutivi, aiutando principalmente la decomposizione temporale. Fondamentalmente, lo schema MCTF consiste in un filtro a 2 rubinetti basato su Haar o 5/3 trasformazioni wavelet [14], che devono essere applicate sull'ingresso originale prima di qualsiasi elaborazione dell'encoder. All'interno della piattaforma di riferimento JSVM, questo filtro è un modulo software indipendente (contrassegnato come "*MCTFPreProcessorStatic*"). Riceve come ingresso una sequenza video grezza (in formato YUV), generando un file di output filtrato. Al fine di integrare questo modulo MCTF nella

codifica, le sequenze video originali vengono prima filtrate e poi inviate all'encoder JSVM.

I risultati nelle figure 30(a) e 30(b) presentano la qualità video ottenuta con e senza filtro di pre-elaborazione MCTF. È indubbiamente dimostrato che il filtro produce un piccolo miglioramento della qualità video. Per quantificare ulteriormente l'impatto dell'inclusione del filtro MCTF nella procedura di codifica, al tempo di codifica JSVM viene aggiunto il tempo di filtraggio - il ritardo causato dal "*MCTFPreProcessorStatic*". I risultati comparativi sono presentati rispettivamente nelle figure 30(a), 30(b), 30(c) e 30(d) per le risoluzioni CIF e 4CIF, rispettivamente. Vi si osserva chiaramente come MCTF deteriora significativamente le prestazioni globali, aumentando il tempo di esecuzione totale al 300% in tutti i casi.

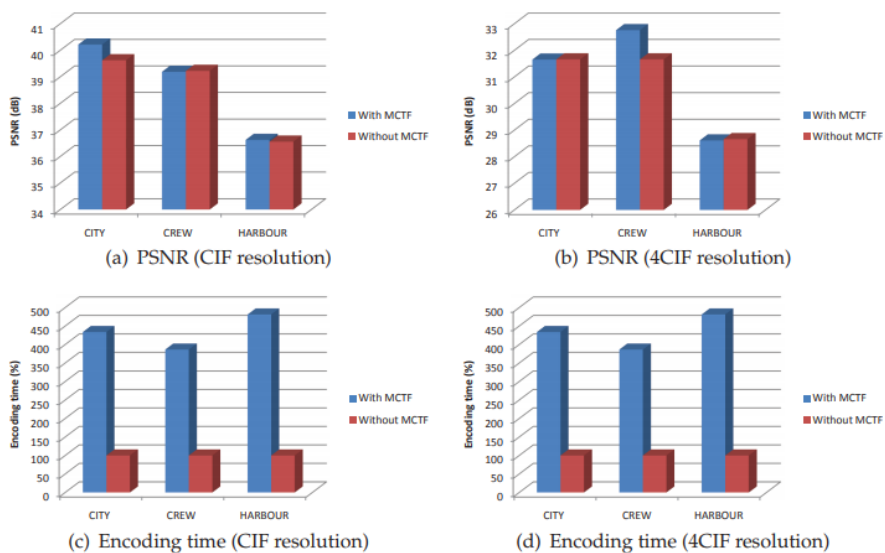


Figura 30. Impatto dell'attivazione del filtro di pre-elaborazione MCTF.

Ref: [31]

## Capitolo 6

---

### Esempi di Configurazione

#### 6.1 Configurazione Ideale Per Uno Streaming Qualitativo

Per questo scenario, che richiede qualità, è stata progettata una configurazione ibrida scalabile con scalabilità temporale (4 strati) e SNR (2 strati) [31]. Questa configurazione di alta qualità è stata progettata in modo da fornire un miglioramento della qualità rispetto alla configurazione di riferimento di base. I parametri chiave modificati per la configurazione di alta qualità proposta, sono l'utilizzo di soli frame B, un'area di ricerca estesa di 92 e la modalità MGS per fornire la scalabilità SNR.

In particolare, i valori di QP determinati per questa configurazione di alta qualità sono  $QP_U=25$  e  $QP_L=30$ . Si ricorda che questi parametri sono solo alcuni esempi particolari, e potrebbero richiedere ulteriori modifiche in altri scenari reali. I risultati pratici ottenuti dalla valutazione delle due configurazioni proposte (riferimento di base e di alta qualità) per le tre sequenze video a risoluzione CIF si possono notare nella Figura 31 dove, per motivi di equità nel confronto, il bitrate di uscita di tutte le configurazioni è stato regolato allo stesso valore (1 Mbps) per poter valutare solo variazioni nella qualità e nelle prestazioni. In primo luogo, è importante osservare la qualità ottenuta nella figura 31(a) quando si utilizza la configurazione di alta qualità suggerita, con guadagni fino a 2.5 dB in alcuni casi. Tuttavia, un impatto considerevole nel contesto di prestazioni di calcolo si ottiene per la configurazione Figura 31(b): il tempo di codifica aumenta di oltre cinque volte in alcuni casi.

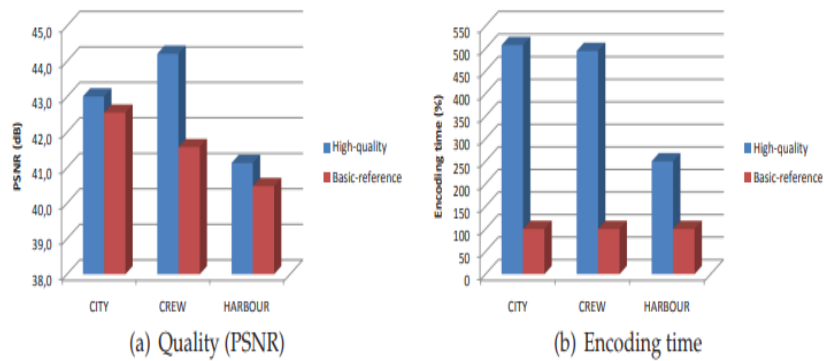


Figura 31. Confronto tra il riferimento di base e le configurazioni di alta qualità.

Ref: [31]

## 6.2 Configurazione Ideale Per Uno Streaming Performante

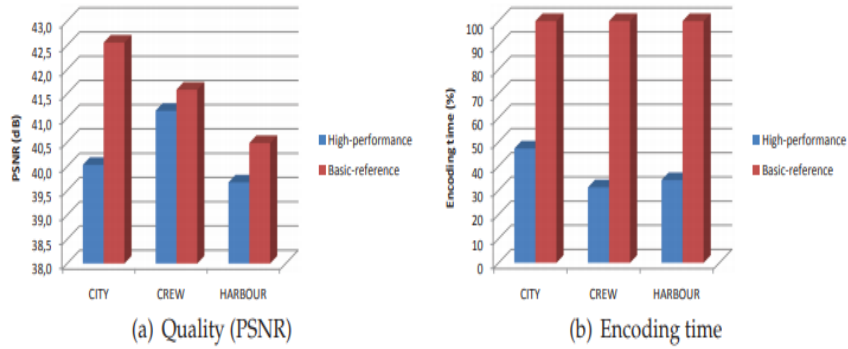
Per applicazioni che richiedono prestazioni in tempo reale, come la videoconferenza diffusa o sistemi di videosorveglianza, il tempo trascorso nella codifica di una sequenza video è critico. In questi casi, le prestazioni computazionali del codec sono considerate decisive, a condizione che la qualità del flusso video non si degrada drasticamente. Per queste applicazioni viene proposta la tecnologia che segue i parametri:

- dimensione GOP pari a 4 con struttura "IPPP" (un frame I e tre frame P per GOP senza includere i fotogrammi B),
- ILP in modalità di ricerca rapida con area di ricerca ridotta a 16 e passi di quantizzazione regolati a  $QP_U=36$  e  $QP_L=38$ .

Anche in questo caso, questi valori specifici sono una conseguenza delle linee guida generali di progettazione.

Confrontando sia il riferimento di base che le configurazioni ad alte prestazioni in termini di qualità (Figura 32(a)), si può osservare che la degradazione del PSNR varia a seconda della sequenza video codificata, cioè il PSNR per la sequenza video CREW è quasi uguale per entrambe le configurazioni, mentre il PSNR per le sequenze video CITY e HARBOUR

diminuisce approssimativamente fino a 1 e 2 dB rispettivamente. Tuttavia, questo inconveniente trova la sua controparte nell'evidente miglioramento delle prestazioni di calcolo mostrate in Figura 32 (b), dove si conclude che il tempo di codifica per la configurazione ad alte prestazioni è almeno due volte più veloce della soluzione di riferimento di base per tutte le sequenze video.



**Figura 32. Confronto tra il riferimento di base e le configurazioni ad alte prestazioni.**

Ref: [31]



## CAPITOLO 7

---

### Confronto Tecnologico

Nell'ultimi anni si sta notando un forte interesse da parte dell'industria a fornire video tramite il protocollo HTTP. A differenza del tradizionale video on demand (VoD) dove il video viene consegnato ad un preciso bitrate e la rete sottostante deve garantire un certo livello di servizio, il video trasmesso tramite HTTP usa canali di comunicazione pubblici e quindi si adatta alle diverse condizioni di larghezza di banda e di rete durante le operazioni di riproduzione. I servizi che utilizzano DASH in una prima fase catturano dei segmenti e dopo li riproducono affinché il contenuto sia finito. Tale pre-fetch protegge i servizi dalle fluttuazioni della larghezza di banda disponibile e aumenta anche la capacità del sistema di trasmissione [26]. Poiché il contenuto è pre-recuperato tramite HTTP, non vi sono perdite di pacchetti ed inoltre, non vi è neppure la necessità di codificare il video ad un bitrate costante (CBR), come è necessario in servizi come la trasmissione e lo streaming video in tempo reale. Un flusso codificato con un bitrate medio su un segmento di una data durata fornisce una migliore qualità e non influisce sulla riproduzione quando viene consegnato utilizzando DASH.

#### Codifica dei contenuti

##### A. Multi bitrate Video

Quando le condizioni di rete cambiano durante una sessione, la riproduzione video deve adattarsi alle mutevoli condizioni. Questo adattamento dinamico è supportato dalla codifica dei segmenti video a bitrate multipli e dal recupero dei segmenti al bitrate appropriato. La lunghezza dei segmenti gioca un ruolo importante nel determinare quanto velocemente il sistema possa adattarsi. Segmenti più corti permettono al sistema di adattarsi più velocemente. Tuttavia, poiché un intra (I) frame è necessario quando si cambiano i segmenti, i segmenti più corti tendono a diventare più inefficienti a causa del loro costo. Un candidato da prendere in considerazione quando si supportano più bitrate è lo Scalable Video

Coding (SVC) [13]. SVC codifica il video come strato di base assieme a livelli di miglioramento, con ogni livello che aumenta la qualità rispetto al livello precedente. Esso è progettato per supportare più bitrate e risoluzioni in modo efficiente. Tuttavia, il supporto della scalabilità aumenta il sovraccarico rispetto ad MPEG-DASH dato che SVC utilizza circa il 20% in più di bit per ottenere la stessa qualità (questo aumento dipende dal contenuto). Mentre i livelli multipli di un video codificato utilizzando SVC possono supportare bitrate adattivi, per DASH ogni livello deve essere codificato ad un specifico bitrate.

### Analisi e confronto costi

Assumendo che i costi sono modellati in base a quanto segue:

Video di 90 minuti sono codificati a bitrate multipli utilizzando SVC e DASH. Per le rappresentazioni di DASH sono stati usati 16 bitrate da 500 Kbps a 8000 Kbps. Per SVC, 16 strati sono codificati con strato di base codificato a 500 Kbps e ogni strato di miglioramento codificato per fornire una qualità equivalente del bitstream DASH. Il modello non considera separatamente le risoluzioni video o la scalabilità spaziale siccome il bitrate è il fattore chiave nella valutazione del costo.

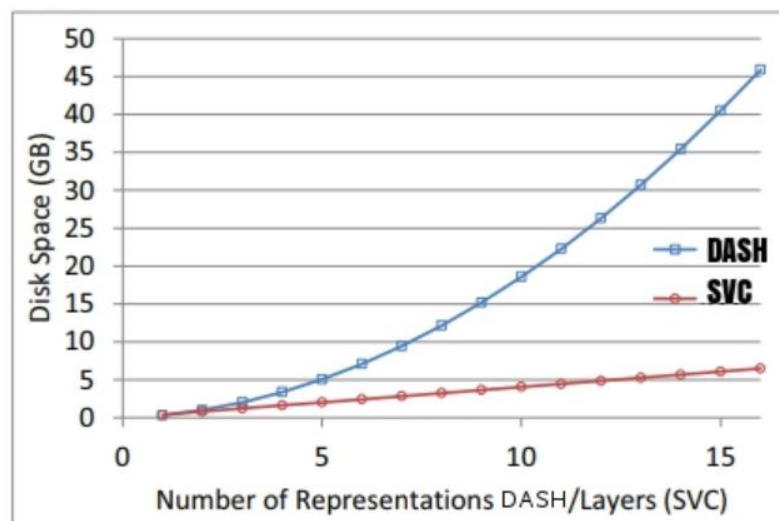


Figura 33. Requisiti di memorizzazione tra SVC e DASH player

Ref: [32]

La Figura 33 mostra i requisiti di memorizzazione di SVC e DASH in funzione del numero di bitrate supportati dal servizio. E' facile notare che maggiore e' il numero di rappresentazioni, maggiore risulta essere lo spazio necessario per la memorizzazione dei contenuti. Questo pero' puo' cambiare se prendiamo in considerazione la frequenza di uso.

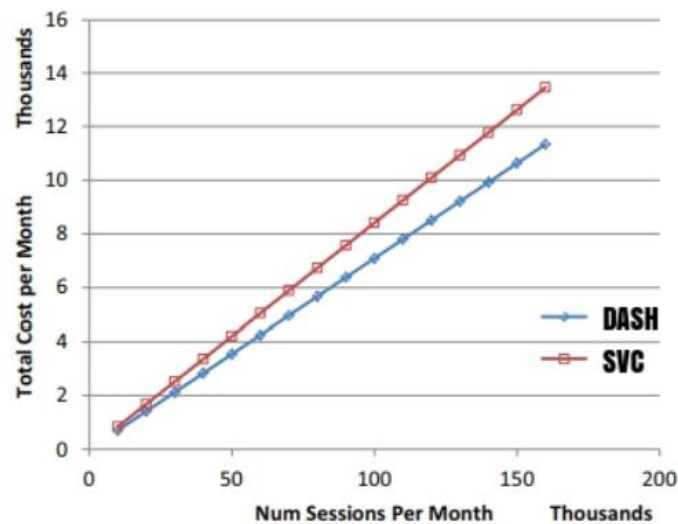


Figura 34 Il costo in base alla frequenza di uso.

Ref: [32]

Il costo di SVC aumenta con l'aumentare del numero di sessioni. Questo modello presuppone che vengano utilizzate 5 rappresentazioni da 500 Kbps a 2500 kbps e che le sessioni siano distribuite uniformemente tra i bitrate. Poiché il costo di una sessione è dominato dai costi di larghezza di banda, le sessioni SVC costano di più e quindi il costo delle sessioni SVC rimane basso se il numero di sessioni rimane basso. Nel caso però in cui la frequenza delle sessioni risulta moderata i risultati sono molto simili tra loro come si può notare nella figura 35

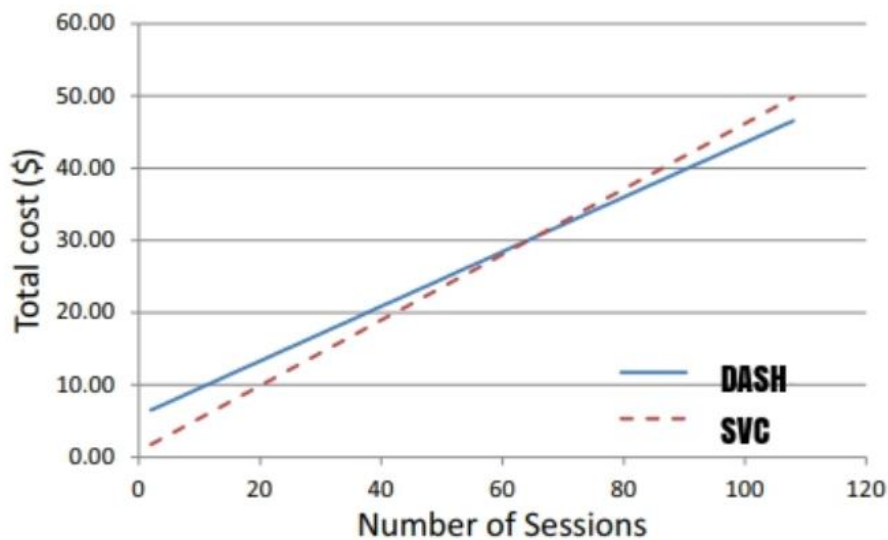


Figura 35. Riducendo il numero di sessioni richieste entrambe le tecniche danno risultati simili

Ref: [32]

Come mostrato nella figura 35, se il numero totale di sessioni è inferiore a 64, il costo dell'SVC rimane inferiore al costo delle sessioni DASH. SVC è efficace per i servizi di conversazione in tempo reale in cui la larghezza di banda tra i partecipanti determina il bitrate erogato [27]. Per le applicazioni on-demand (topologia uno-a-uno), come quelle previste per DASH, il ricevitore richiede il bitrate desiderato dal server e, quindi, l'overhead del video scalabile è un costo inutile.

Mentre SVC ha requisiti di archiviazione inferiori rispetto a DASH, i costi totali sono inferiori per DASH. Il costo della trasmissione video è dominato dal costo della larghezza di banda. Quello dello storage invece diventa un fattore da prendere in considerazione quando il numero di sessioni è molto piccolo, come nel caso dei film meno popolari in un catalogo. Questo rappresenta un'interessante opportunità per ridurre i costi di un servizio video utilizzando SVC per i video con meno sessioni.

## CONCLUSIONI

---

In conclusione, decidere con certezza quale sia la migliore tra le due tecnologie elencate in questa proposta risulta complicato. In realtà, non ci sono vincitori per il momento. Questo trova riscontro nel fatto che tutto dipende dall'obiettivo degli sviluppatori. Ogni protocollo e tecnologia per la trasmissione adattiva di dati (MPEG-DASH, H.264/SVC) ha i suoi svantaggi e i suoi pregi. Ovviamente ci sono fattori esterni che possono indurre a prendere una scelta invece che un'altra come per esempio lo scopo dello sviluppo (applicazione per streaming, playback etc) e ai dispositivi ai quali e' mirata la predetta applicazione.

Sembra che quando il nostro obiettivo risulti essere il pubblico non mirato (quantità > qualità), implementare un'applicazione basandosi su SVC, (per il momento), risulta essere la scelta migliore data la flessibilità che quest'ultimo offre in termini di bitrate. Inoltre, il fatto che SVC codifichi il flusso di dati su varie layer rende possibile ai 'vecchi' dispositivi di decodificare il flusso e visualizzare il contenuto anche se con la qualità minore rendendo le risorse accessibili a una vasta gamma di dispositivi a prescindere dall'architettura. Come si intuisce da questa affermazione un'applicazione del genere può essere usata su larga scala e da dispositivi molto diversi tra di loro (tecnologicamente parlando). In base alle potenzialità di ogni dispositivo il proprio decoder decodifica fino a quando non supporta il flusso di dati.

MPEG-DASH essendo ancora abbastanza nuovo ha una supportabilità limitata a nuovi decoder e quindi dispositivi pre-MPEG-DASH non possono favorire delle sue funzioni. Questo ovviamente restringe il campo d'azione e quindi per uno scopo di utilizzo globale può risultare penalizzante. Sebbene per il momento la situazione rimanga stabile, gli sviluppatori sono forti sostenitori dell'idea che MPEG-DASH sia il futuro proprio per la sua versatilità nell'adattarsi a qualunque situazione di banda e alla capacità nel gestire le risorse per massimizzare l'efficienza durante la riproduzione del flusso di dati.

## RINGRAZIAMENTI

---

.....*Ringrazio il Professore Bedogni Luca per avere portato pazienza con la quantita' di email che gli sono state inviate da parte mia e tutti i miei amici che mi hanno supportato durante questo corso. Buona Fortuna a tutti!*

.....

*Dedicato alla mia Famiglia che non ha smesso di credere in me neanche un momento.*

## BIBLIOGRAFIA & RIFERIMENTI

---

- [1] Christopher Muller, Stefan Lederer, Christian Timmerer, 2012, An Evaluation of Dynamic Adaptive Streaming over HTTP in Vehicular Environments
- [2] Gannes, Liz, 10 June 2009, The Next Big Thing in Video: Adaptive Bitrate Streaming
- [3] Douglas E. Comer, 2006, Internetworking with TCP/IP - Principles, Protocols and Architecture
- [4] Junchen Jiang, Vyas Sekar, and Hui Zhang. 2012. Improving Fairness, Eciency, and Stability in HTTP-based Adaptive Video Streaming with FESTIVE. In Proc. ACM Int. Conf. on Emerg. Net. Exper. and Techn. (CoNEXT). 97–108.
- [5] Z. Li, X. Zhu, J. Gahm, R. Pan, H. Hu, A. C. Begen, and D. Oran. 2014. Probe and Adapt: Rate Adaptation for HTTP Video Streaming At Scale. IEEE J. Sel. Areas Commun. 32 (April 2014).
- [6] Te-Yuan Huang, Ramesh Johari, Nick McKeown, Mahew Trunnell, and Mark Watson. 2014. A Buer-based Approach to Rate Adaptation: Evidence from a Large Video Streaming Service. In Proc. ACM SIGCOMM.
- [7] K. Spiteri, R. Uргаonkar, and R. K. Sitaraman. 2016. BOLA: Near-optimal bitrate adaptation for online videos. In IEEE INFOCOM.
- [8] A. Beben, P. Wisniewski, J. Mongay Batalla, and P. Krawiec. 2016. ABMA+: Lightweight and Efficient Algorithm for HTTP Adaptive Streaming. In Proc. Int.ACM Conference on Multimedia Systems (MMSys)
- [9] Huang, H.-S., Peng, W.-H. & Chiang, T. (2007). Advances in the scalable amendment of h.264/avc, IEEE Communications Magazine 45(1):68

- [10] Ohm, J.-R. (2005). Advances in scalable video coding, Proceedings of the IEEE 86(1): 42–56.
- [11] Rieckl, J. (2008). Scalable video for peer-to-peer streaming, Master's thesis, University of Wien.
- [12] Husemann, R., Roesler, V. & Susan, A. (2009). Introduction of a zonal search strategy for svc inter-layer prediction module, VLSI-SOC 2009, Florianopolis, Brazil.
- [13] Schwarz, H., Marpe, D. & Wiegand, T. (2007). Overview of the scalable video coding extension of the H.264/AVC standard, IEEE Transactions on Circuits and Systems for Video Technology 17(9): 1103–1120.
- [14] Schafer, R., Schwarz, H., Marpe, D., Schierl, T. & Wiegand, T. (2005). Mctf and scalability extension of h.264/avc and its application to video transmission, storage and surveillance, Proceedings of the SPIE, pp. 343–354.
- [15] Segall, A. & Sullivan, G. (2007). Spatial scalability within the h.264/avc scalable video coding extension, IEEE Transactions on Circuits and Systems for Video Technology 17(9): 1121–1135.
- [16] Unanue, I., Del Ser, J., Sanchez, P. & Casasempere, J. (2009). H.264/svc rate-resiliency tradeoff in faulty communications through 802.16e railway networks, Ultra Modern Telecommunications and Workshops, 2009. ICUMT '09. International Conference on, pp. 1–6.
- [17] Kasarapu Ramani, LAP LAMBERT Academic Publishing (January 25, 2018), Media Presentation Description over MPEG-DASH Paperback – January 25, 2018
- [18] Michael B. Spring, 2002, Information Technology Standards. MIT Press
- [19, 20, 21] <https://www.internetvideoarchive.com/documentation/video-api/progressive-download-vs-adaptive-bitrate/>



[22] H. Liu, Y. Wang, Y. R. Yang, A. Tian, and H. Wang. Optimizing Cost and Performance for Content Multihoming. In Proc. SIGCOMM, 2012.

[23] X. Liu, F. Dobrian, H. Milner, J. Jiang, V. Sekar, I. Stoica, and H. Zhang. A Case for a Coordinated Internet Video Control Plane. In SIGCOMM, 2012

[24] <https://bitmovin.com/dynamic-adaptive-streaming-http-mpeg-dash/>

[25] L. Popa, A. Ghodsi, and I. Stoica. HTTP as the narrow waist of the future internet. In Proc. HotNets, 2010.

[26] H. Kalva and B. Furht, “Techniques for Improving the Capacity of Video-on-Demand Systems,” Proceedings of the 29th Hawaii International Conference on System Sciences HICSS-29), Vol. 2, Jan 3-6 1996.

[27] A. Eleftheriadis, R. Civanlar, and O. Shapiro, “Multipoint videoconferencing with scalable video coding,” in Journal of Zhejiang University – Science A, vol. 7, no. 5, pp. 696–705, May 2006.

[28] L. A. Grieco and S. Mascolo. Performance evaluation and comparison of westwood+, new reno and vegas tcp congestion control. ACM Computer Communication Review, 34, 2004.

[29][https://www.researchgate.net/publication/316680315\\_A\\_Comparative\\_Case\\_Study\\_of\\_HTTP\\_Adaptive\\_Streaming\\_Algorithms\\_in\\_Mobile\\_Networks](https://www.researchgate.net/publication/316680315_A_Comparative_Case_Study_of_HTTP_Adaptive_Streaming_Algorithms_in_Mobile_Networks)

[30]<https://www.sciencedirect.com/science/article/pii/S0141938216300269>

[31] <https://www.researchgate.net/publication/221912993>

[32] <https://www.researchgate.net/publication/232075102>



