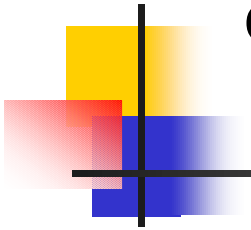
Politecnico di Bari
Facoltà di Ingegneria
Corso di laurea in Ingegneria Elettronica



Tesi di laurea

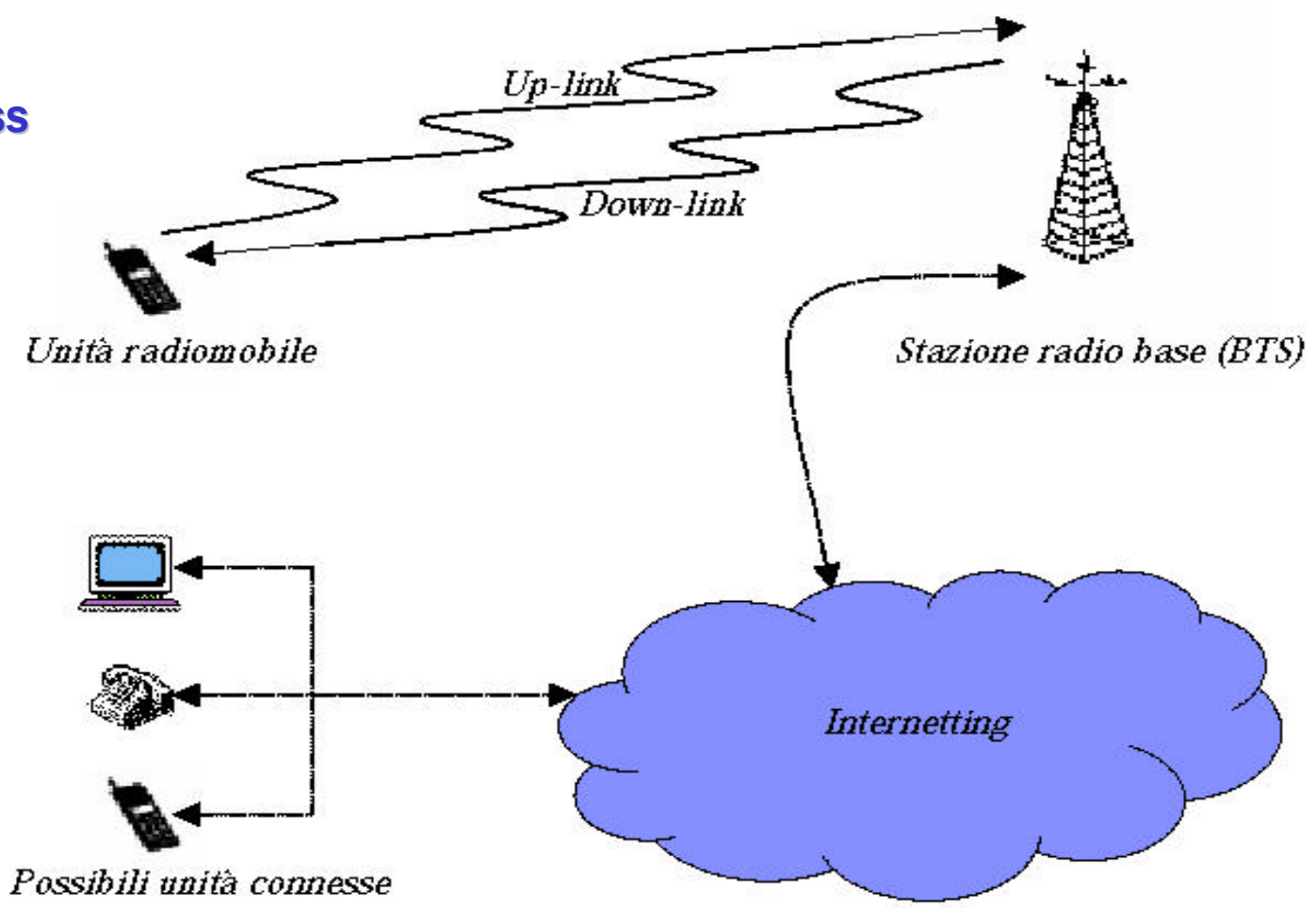
Simulazioni di protocolli di compressione dell' header per sistemi wireless basati su piattaforma TCP/IP

Relatore:
Chiar.mo Prof. Pietro Camarda

Laureando:
Sandro Petrizzelli

Scenario tipico per una rete wired-wireless

1. **Rete mista wired-wireless**
2. **Rete “all-IP”**



Perché comprimere l'header?

1. Per risparmiare banda

2. Per migliorare il FER

1. La riduzione dell'overhead dell'header equivale ad un risparmio di banda

$$Banda_{TOTsc} ? \frac{H ? P}{T} ? (H ? P) ? f_r ? H ? f_r ? P ? f_r ? Banda_{Header} ? Banda_{Utile}$$

$$OVERHEAD\% ? \frac{H}{H ? P} ?100 ? \frac{H}{H ? P} ?100 ? \frac{f_r}{f_r} ? \frac{Banda_{Header}}{Banda_{TOTsc}} ?100$$

2. A parità di caratteristiche di rumore (BER, Bit Error Rate), la compressione permette di trasmettere meno bit e quindi determina un FER (Frame Error Rate) inferiore

$$BER ? x$$

$$FER ? (P ? H) \cdot 8 \cdot BER$$

$$FER_C ? (P ? C) \cdot 8 \cdot BER$$

$$? FER\% ? \frac{FER ? FER_C}{FER} ? \frac{H ? C}{H ? P}$$

P il payload (in byte);

H la dimensione dell' header (in byte);

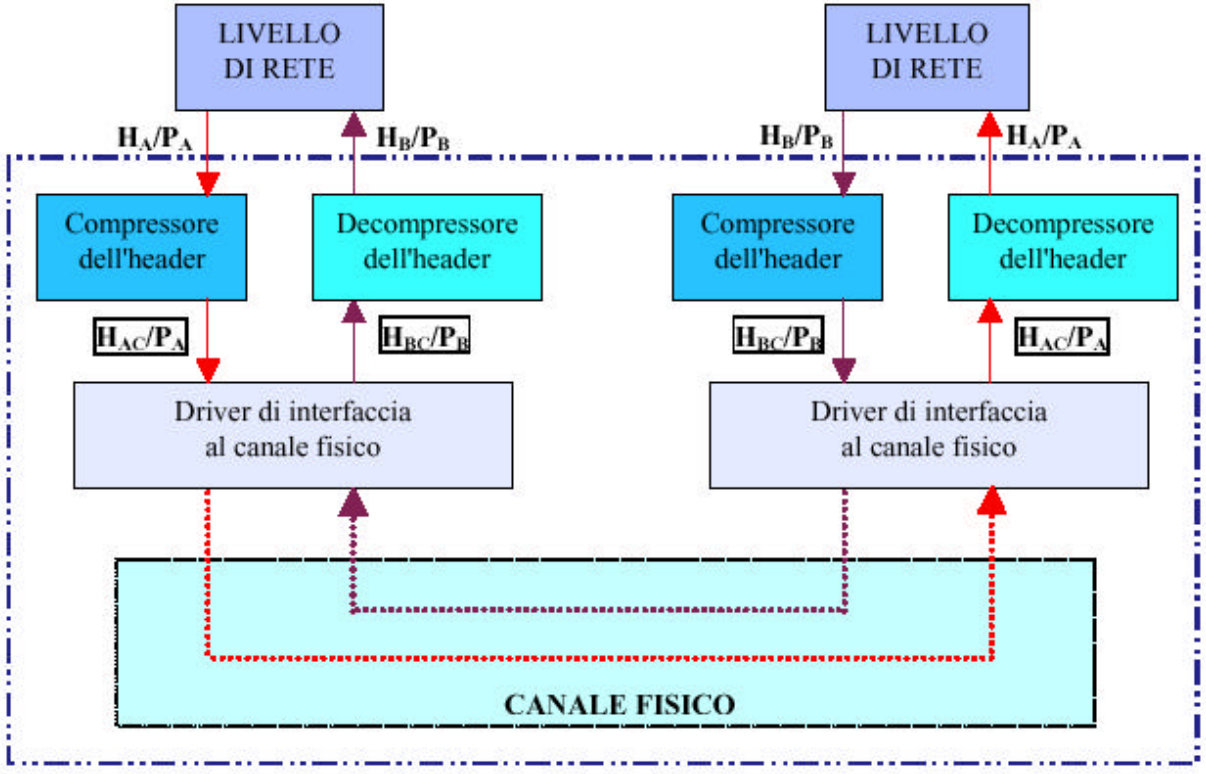
C la dimensione dell' header compresso (in byte);

FER ? (P ? H) ·8 ·BER;

FER_C è il FER in presenza di compressio ne dell' header;

? FER% è il migliorame nto del FER.

Architettura TCP/IP con *unità di compressione/decompressione*



Regione di comprimibilità dell'header: delimita la sezione entro cui gli header, fino al livello rete, non esplicano alcuna funzione

P_A=Payload generato dal processo A

H_A=Pila di header fino al livello rete

H_{AC}/ P_A=pacchetto con header compresso, da inviare sul canale

Algoritmo ROHC (RObust Header Compression)

Riferimenti principali

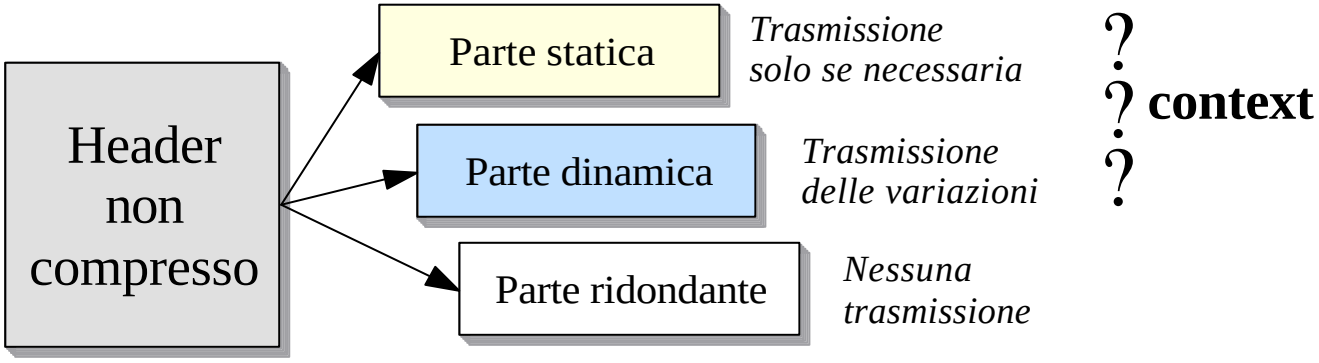
- **RFC 3095** (luglio 2001) – ROHC: Framework and four profiles (RTP, UDP, ESP, and uncompressed)
  - **Profilo 1 - RTP/UDP/IP**
    Per la compressione di flussi di tipo RTP/UDP/IP. Definisce anche le linee guida per ogni eventuale nuovo profilo da integrare nello schema ROHC
    - **Profilo 2 - UDP/IP**
    - **Profilo 3 – ESP/IP**
    - **Profilo 0 – Flussi non compressi**

Profilo 4 - TCP/IP

Per la compressione di flussi di tipo TCP/IP

Algoritmo ROHC
Generalità

- La compressione sfrutta la **ridondanza** esistente tra i campi dell'header sia nel singolo pacchetto sia soprattutto tra pacchetti dello stesso flusso
- Ai fini della compressione, ognuno degli header gestiti dall'algoritmo viene suddiviso in tre parti: **statica**, **dinamica**, **ridondante**



- L'insieme delle parti statica e dinamica, integrato da informazioni utili addizionali, costituisce il **context.** Ogni context è identificato da un numero detto **CID**
- Se i context di decompressore e compressore sono gli stessi (**sincronizzazione**), tutte le decompressioni risultano corrette (a meno di errori residui sul singolo pacchetto)

Algoritmo ROHC

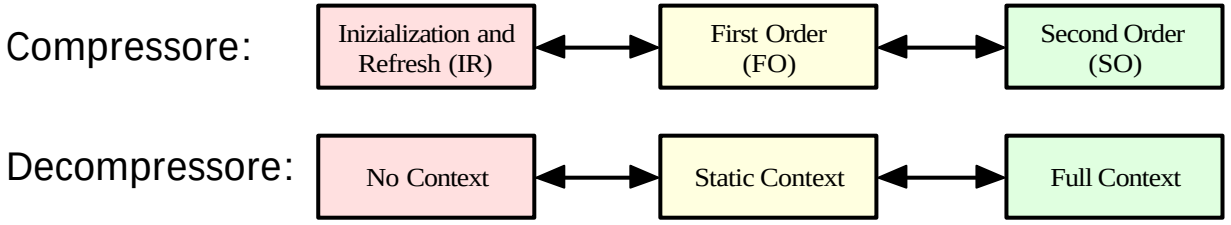
Principi cardine per la compressione ROHC

- **Campo “leader” e “regolarizzazione” del flusso**
  - In ogni flusso deve esistere un campo dinamico (**campo leader**) sulla base del quale ottimizzare la compressione, caratterizzato da una variazione costante fra pacchetti consecutivi ($? = 1$)
  - Quando le variazioni dei campi dinamici, tra pacchetti consecutivi, risultano proporzionali alle variazioni del campo leader, il flusso si dice **regolarizzato** e si invia soltanto la variazione del campo leader, opportunamente codificata
  - Così si ottimizzano sia l’**efficienza** di compressione sia la **robustezza**
  - Nel caso delle connessioni TCP, tale campo non è presente, per cui è stato appositamente introdotto: **SNR** (Sequence Number ROHC, 2 byte)
- **Tecniche di codifica dei campi dinamici**
  - Sono fondamentali per aumentare l’efficienza di compressione e, allo stesso tempo, prevenire le perdite di sincronizzazione del decompressore
  - Si usa la tecnica **W-LSB**, ossia la variante più “robusta” della tecnica LSB
- **Feedback**
  - Qualora sia disponibile un **canale di feedback**, il compressore può beneficiare dell’invio di informazioni di feedback da parte del decompressore per mantenere la sincronizzazione e aumentare il rapporto di compressione
  - Si usa un canale virtuale con piggypacking (soluzione efficiente ma meno affidabile)

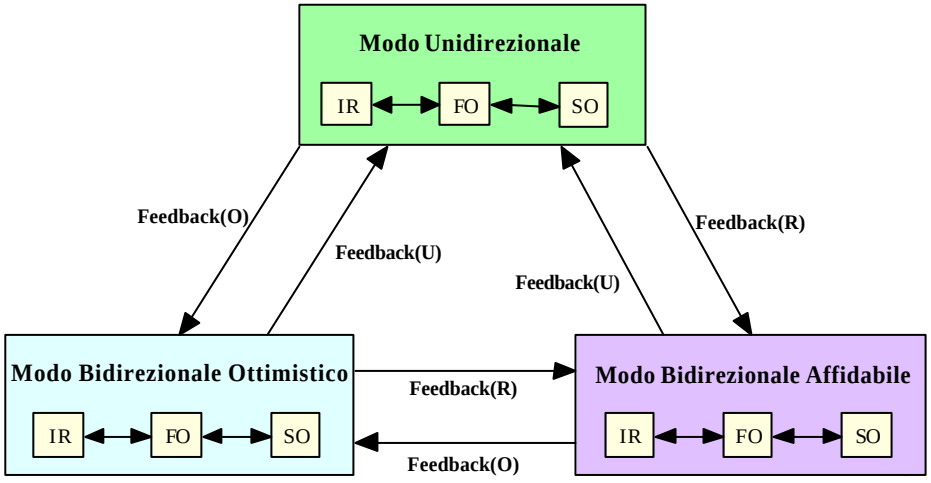
Algoritmo ROHC

Macchine a stati e Modalità operative

- La compressione dell'header con l'algoritmo ROHC può essere vista come l'interazione tra due **macchine a stati**, ciascuna con 3 stati:



- I due organi ROHC possono inoltre funzionare in 3 diverse ***modalità operative***:
  - **Unidirezionale**
  - **Bidirezionale Ottimistica**
  - **Bidirezionale Affidabile**



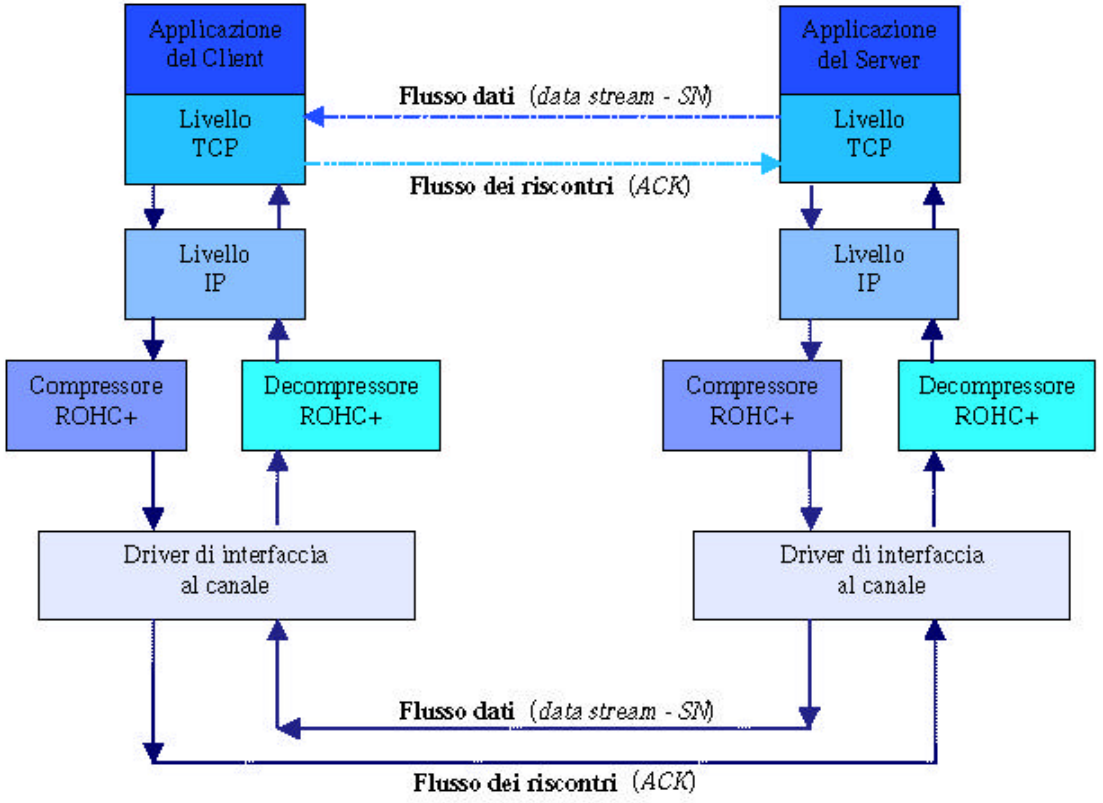
Algoritmo ROHC

Flussi TCP: *Data stream* e *ACK stream*

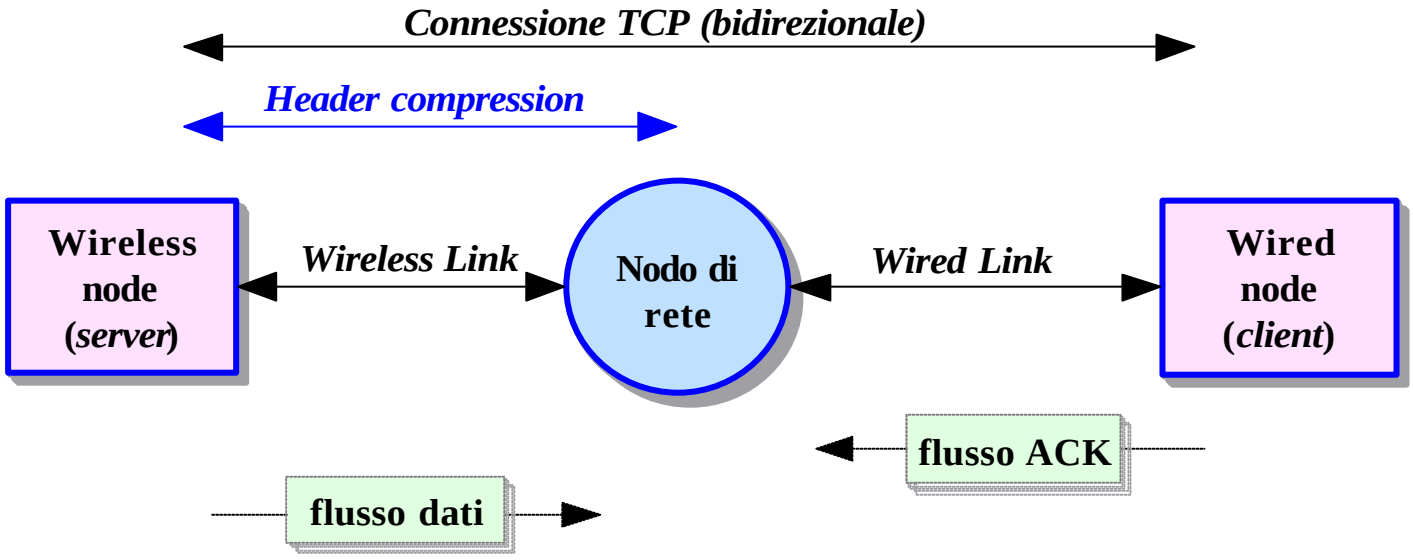
- Il protocollo TCP fornisce un **servizio orientato alla connessione ed affidabile**. Ogni connessione TCP è quindi costituita da due flussi, in direzione opposta, compressi separatamente:
  - Data stream
  - ACK stream

La compressione di ciascun flusso ha prestazioni strettamente legate alle variazioni di un **campo chiave**:

- TCP Sequence Number
- TCP ACK Number



Simulazioni (Network Simulator)



✍ **Scenario 1 (UMTS)**:

wireless link: banda 384 kb/s, ritardo 30 ms, BER 10^{-6}-10^{-3}

wired link: banda 2 Mb/s, ritardo 30 ms, affidabile

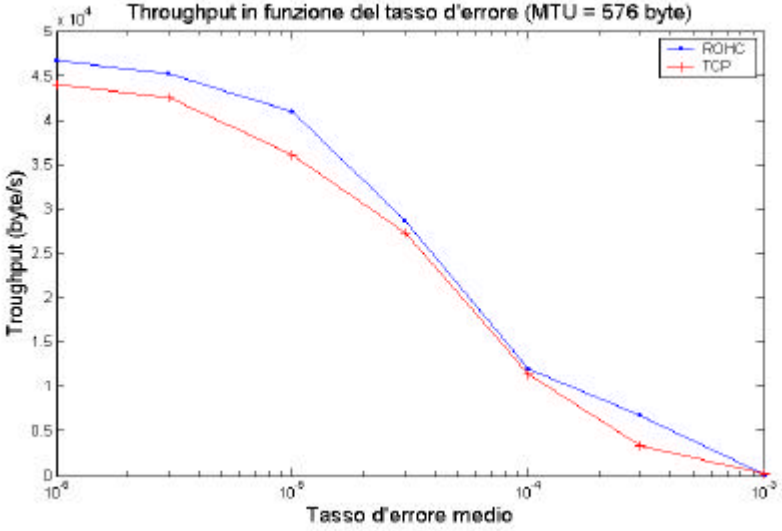
✍ **Scenario 2 (LAN wireless):**

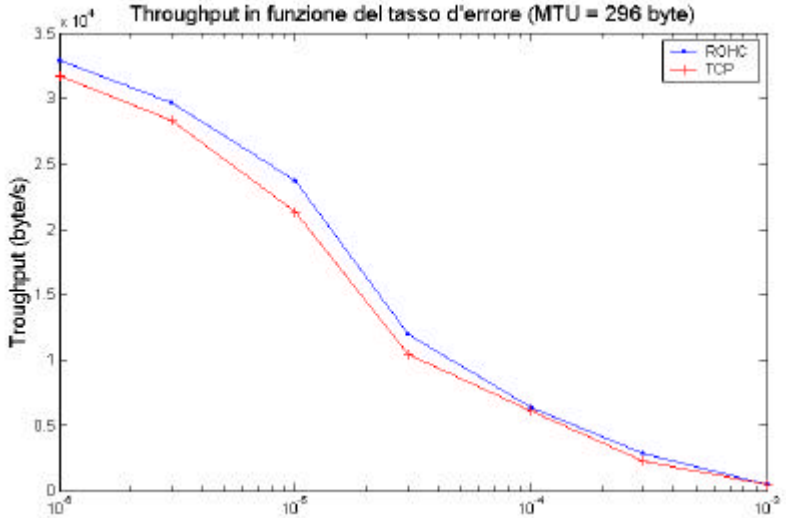
wireless link: banda 2 Mb/s, ritardo 0.1 ms, BER 10^{-6}-10^{-3}

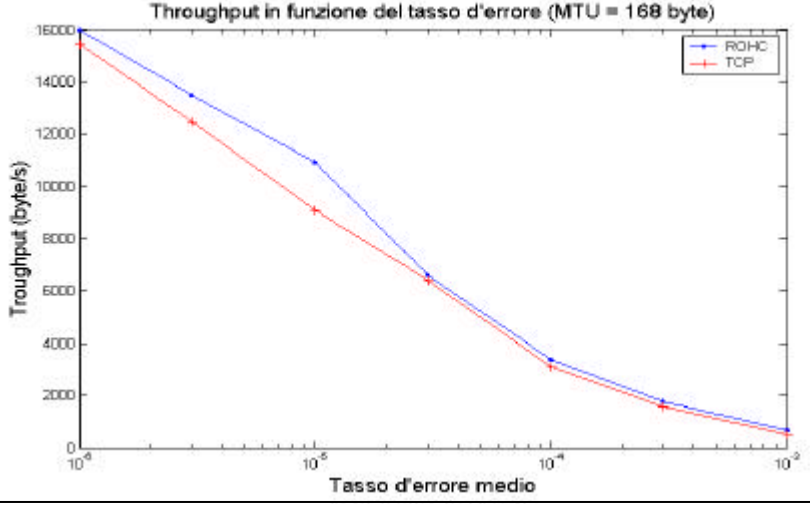
wired link: banda 10 Mb/s, ritardo 45 ms, affidabile

Simulazioni scenario 1

Throughput
(U-mode, MTU=576,296,168byte)





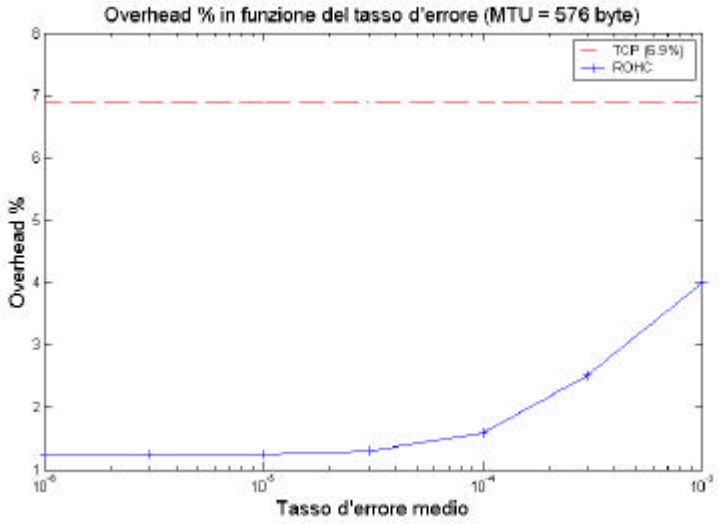


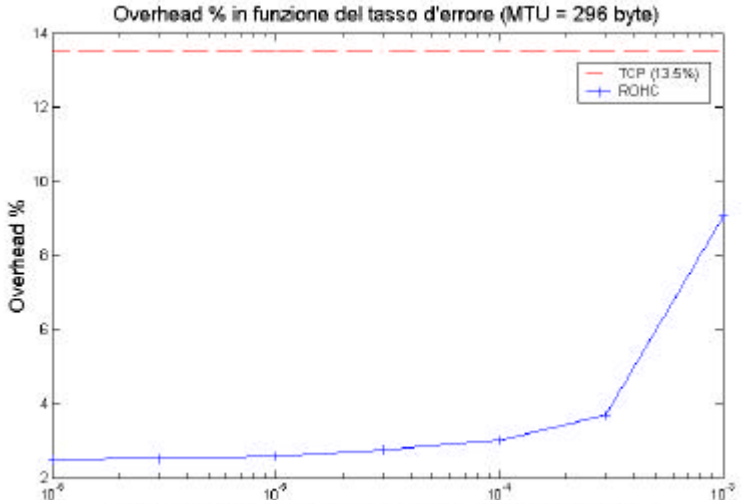
$$Throughput(\text{byte}/\text{s}) \; ? \; \frac{P_{recv}}{T}$$

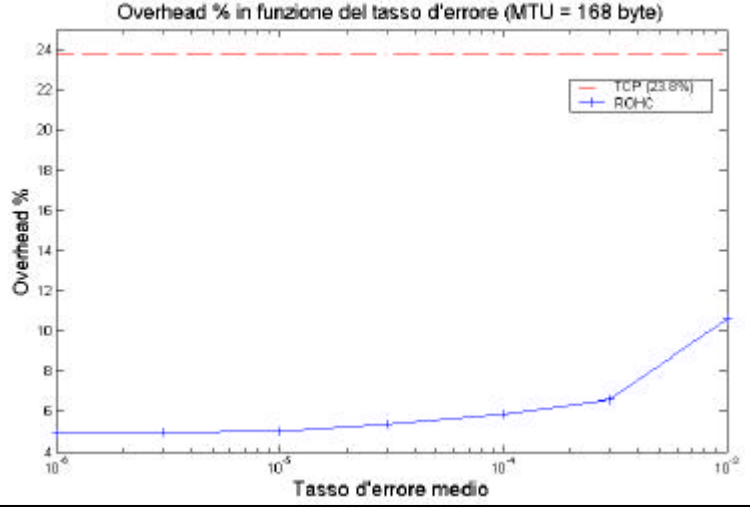
Simulazioni scenario 1

Overhead

(U-mode, MTU=576,296,168 byte)



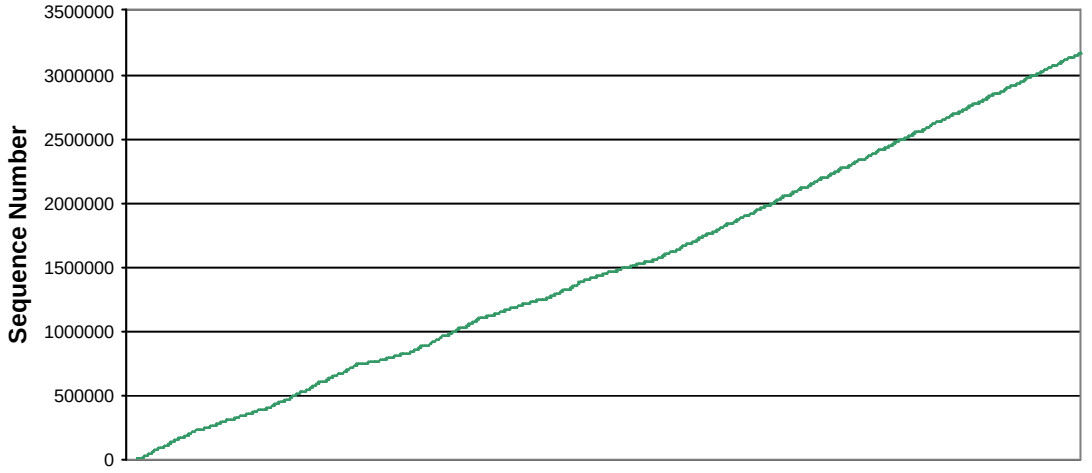




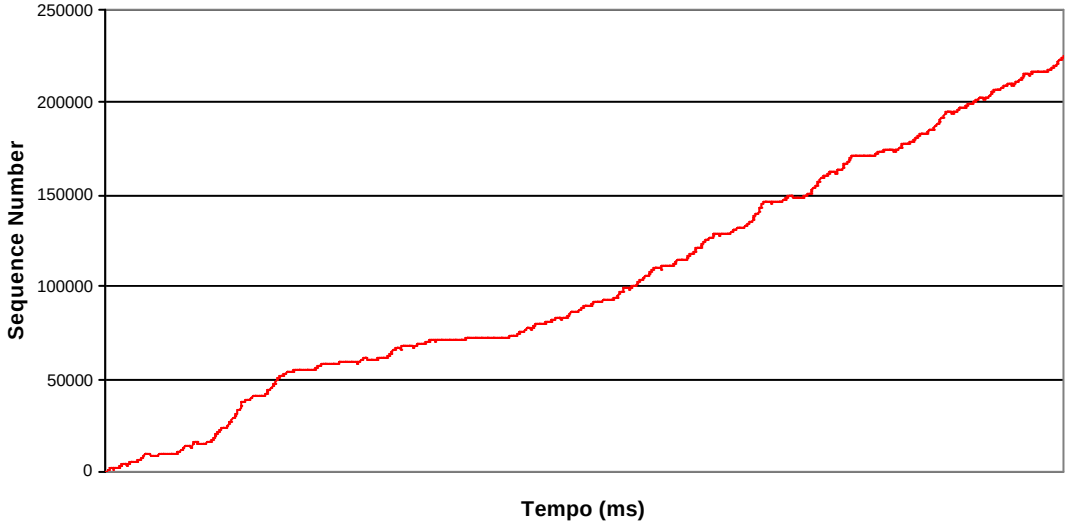
$$Overhead(\%) = \frac{H_{fwd}}{H_{fwd} + P_{fwd}} \cdot 100$$

Simulazioni scenario 1

Andamento nel tempo del Sequence Number (U-mode, MTU=296byte)



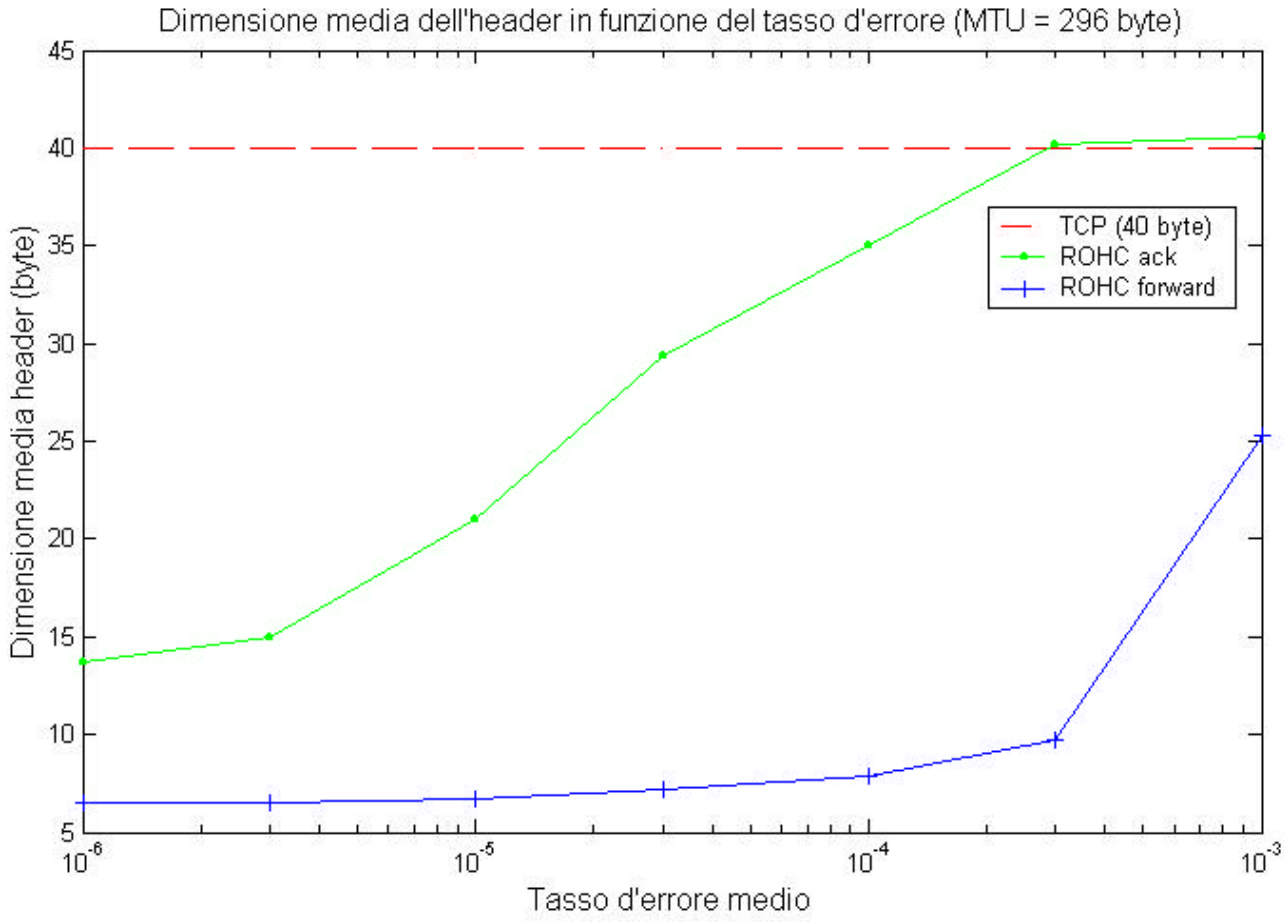
BER=10^{-6}



BER=$3*10^{-4}$

Simulazioni scenario 1

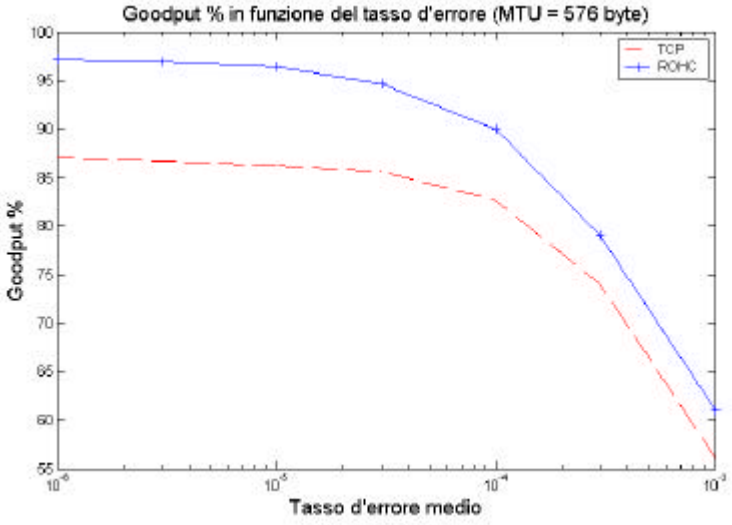
Dimensione media dell'header (U-mode, MTU=296byte)

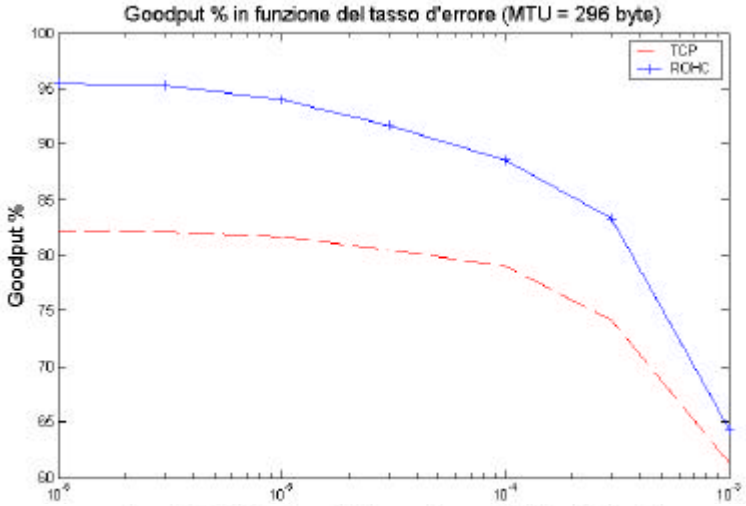


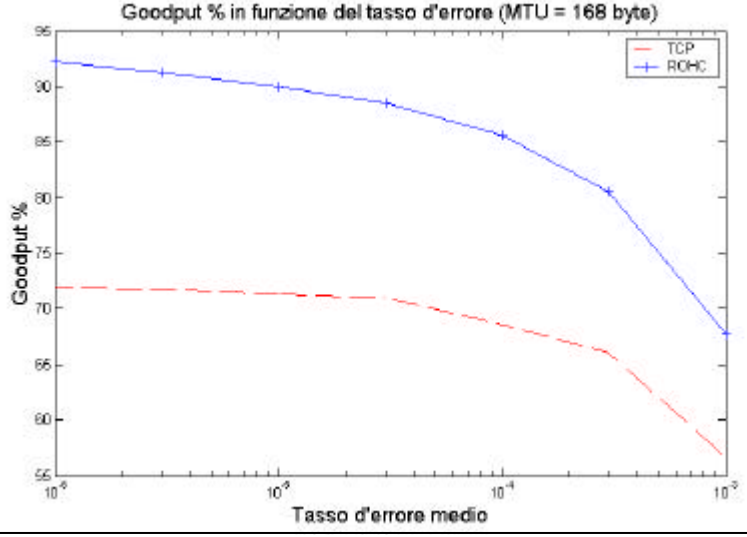
Simulazioni scenario 1

Goodput

(U-mode, MTU=576,296,168byte)





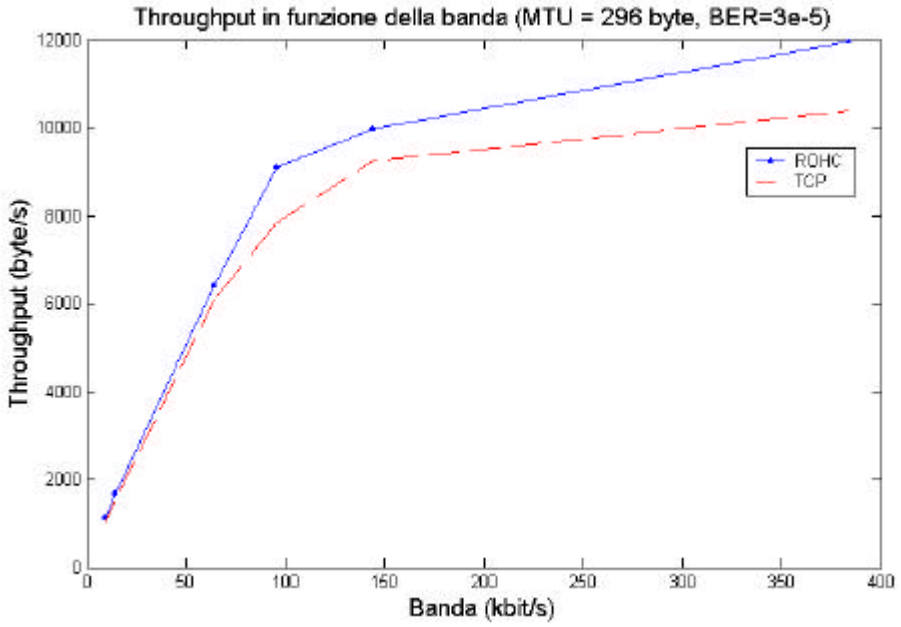


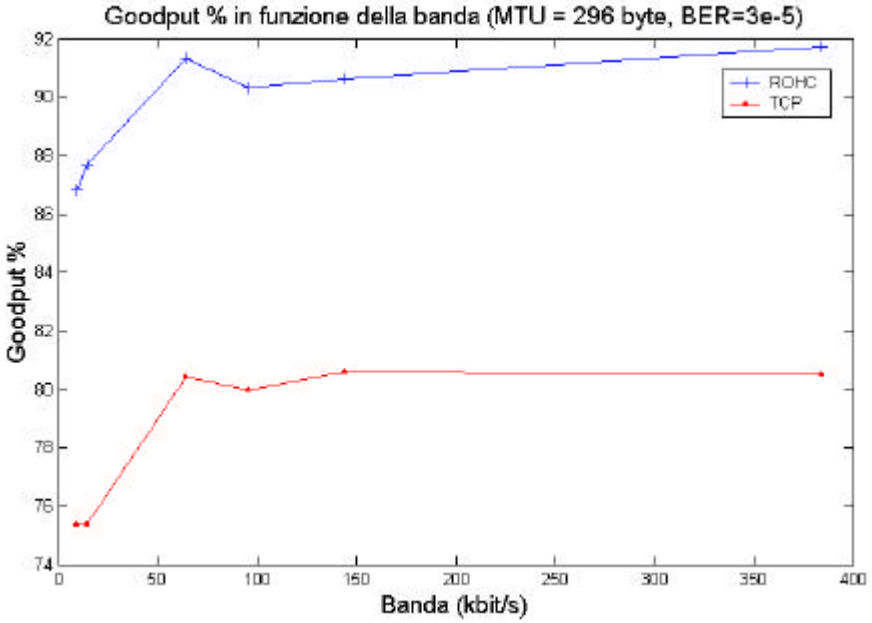
$$Goodput(\%) = \frac{P_{recv}}{(H_{fwd} + H_{ack}) \cdot P_{fwd}} \cdot 100$$

Simulazioni scenario 1

Throughput e goodput in funzione della banda

(U-mode, MTU=296byte, BER=3*10^{-5})





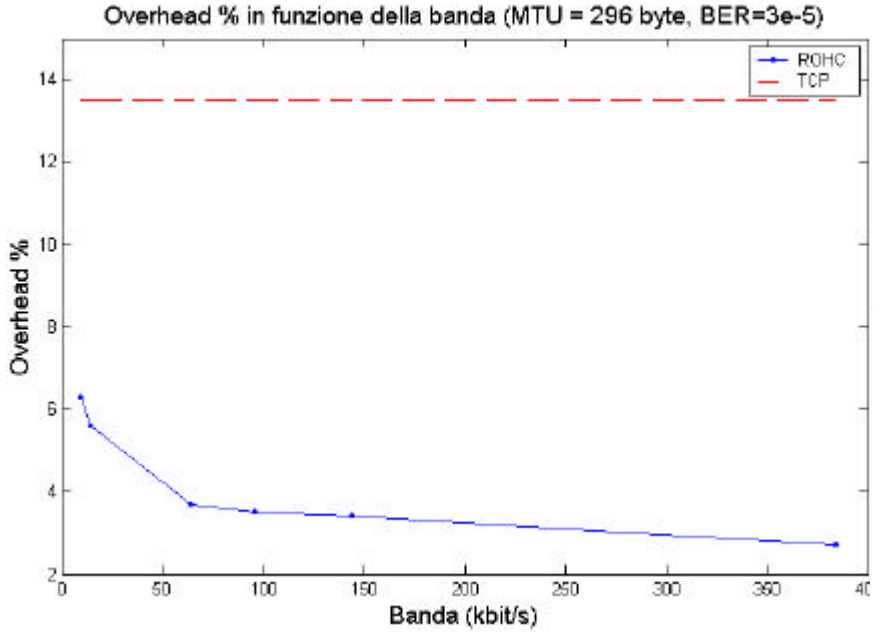
Banda sul wireless:

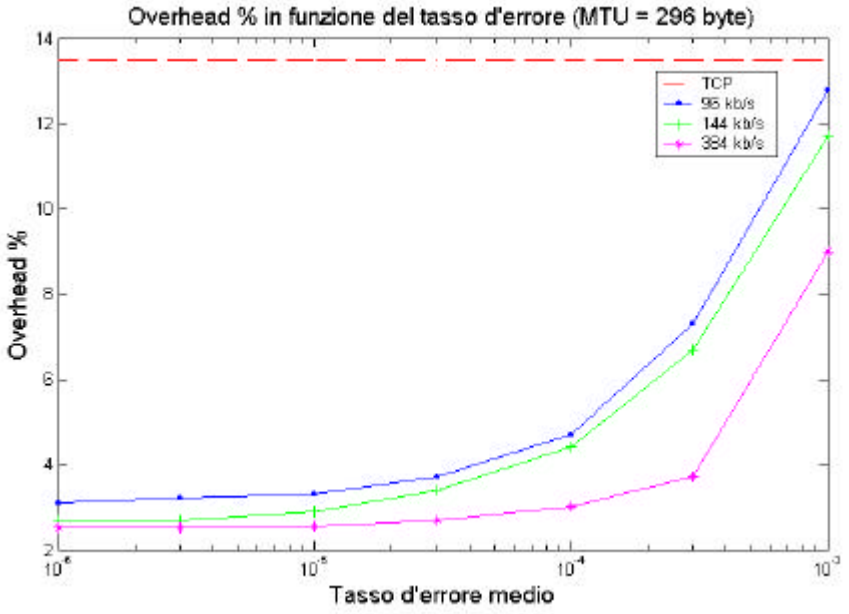
9.6 kb/s, 14.4 kb/s, 64 kb/s,

96 kb/s, 144kb/s, 384kb/s

Simulazioni scenario 1

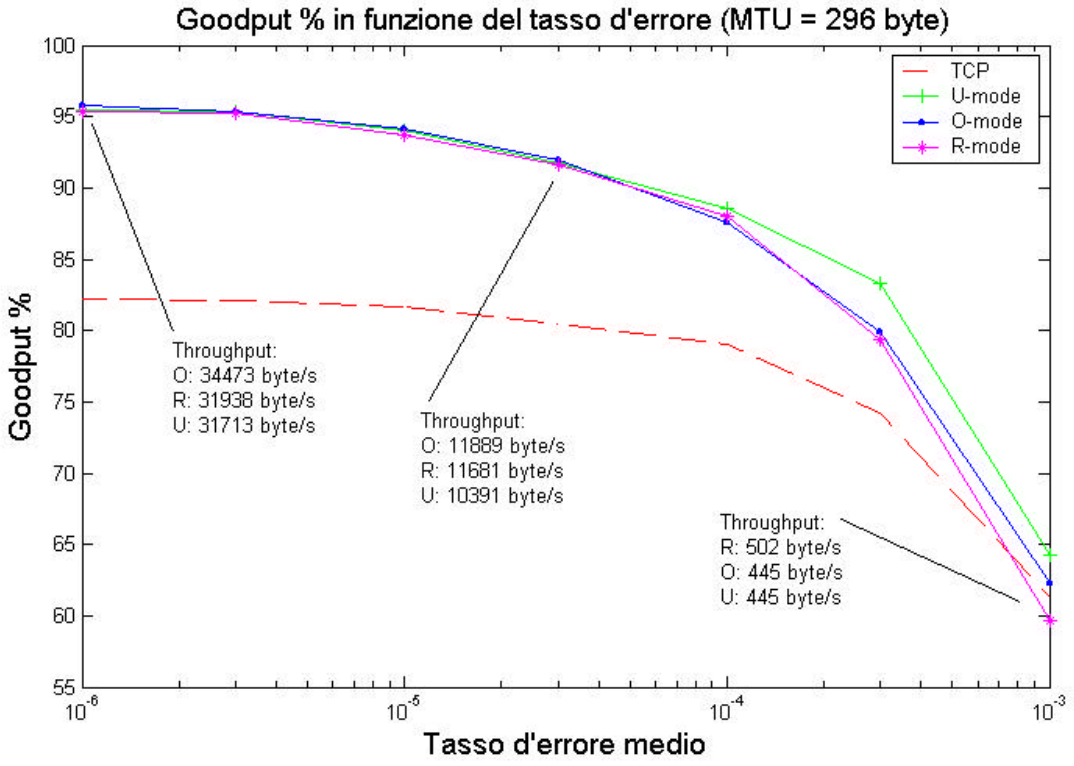
Overhead in funzione della banda (U-mode, MTU=296byte)

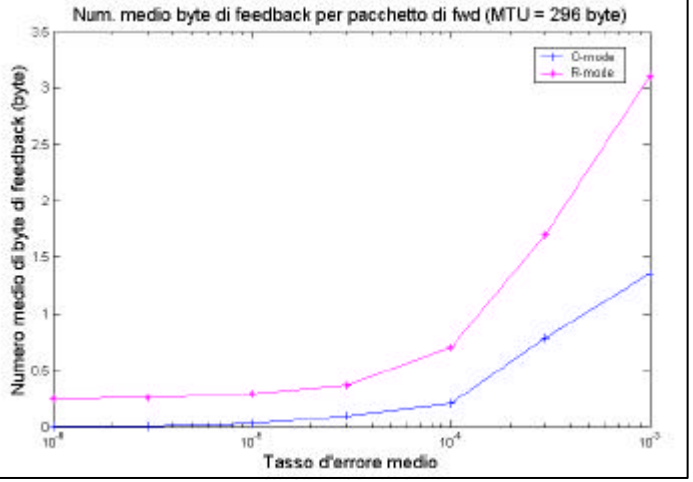




Simulazioni scenario 1

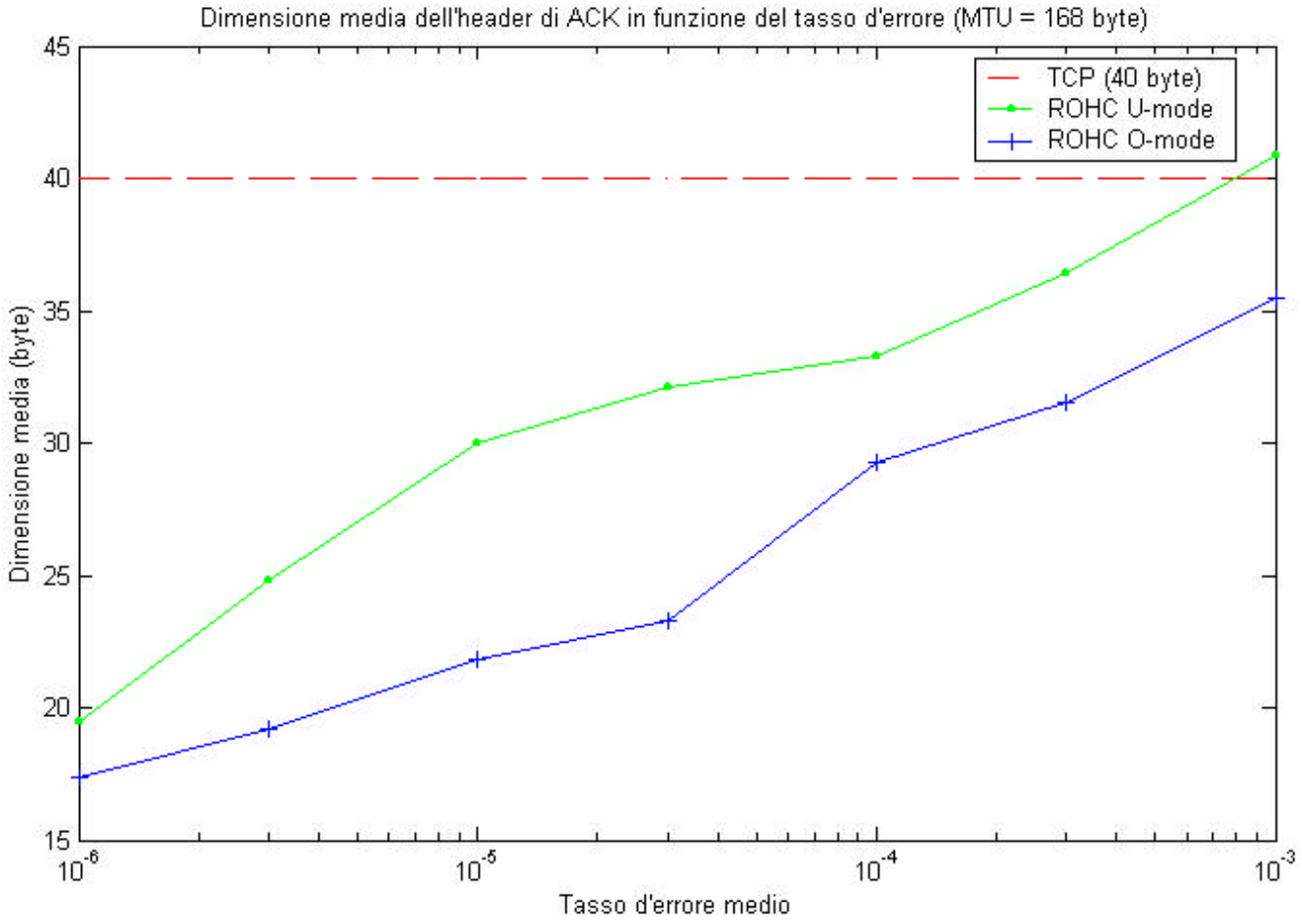
Throughput e Goodput
(U/O/R-mode, MTU=296byte)





Simulazioni scenario 1

Dimensione media header ACK
(U-mode, O-mode, MTU=168byte)



Canale wireless:

64 kb/s, 100 ms

Conclusioni

- Il comportamento dell'algoritmo appare decisamente **soddisfacente** per tassi d'errore non eccessivamente alti (BER<10^{-4})
  - Dimensione media header in condizioni favorevoli: ? 5 byte
  - Pochissime perdite di sincronizzazione (grazie alla tecnica W-LSB)
- Il **traffico degli ACK** compressi risente maggiormente degli elevati tassi d'errore (instabilità del campo ACK Number)
- Ci si attendono prestazioni ancora migliori con **IPv6** (sfruttando la notevole lunghezza dei campi statici)
- E' opportuno un meccanismo di **variazione dinamica** dei parametri di compressione, basato su opportune stime del canale sottostante

Fine presentazione

Sandro Petrizzelli