Università degli studi di Trento
Dipartimento di Ingegneria e Scienza dell'Informazione
Design of Networks and Communication Systems

TCB-BBR: Analisi delle prestazioni

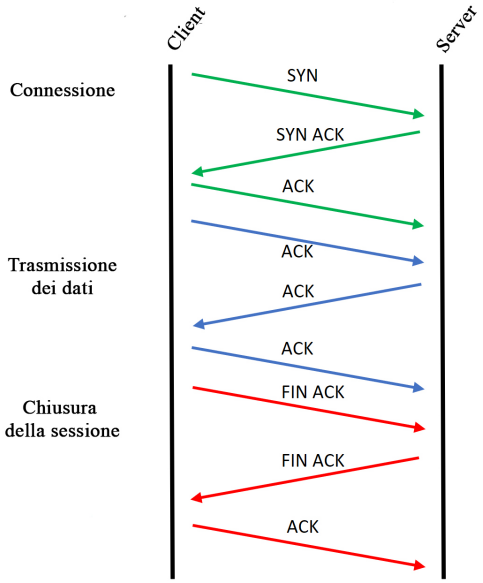
Irene Buffa
Nicola Fiorello
Daniele Mattedi

18 gennaio 2020

Indice

1 TCP (Transmission Control Protocol)

TCP è un protocollo di livello trasporto, e fa parte della cosiddetta suite di protocolli Internet; assieme al protocollo IP, è il più importante tra quelli che caratterizzano la suite intera. La sua funzione, sostanzialmente, è quella di effettuare il controllo della trasmissione, rendendo affidabili le comunicazioni basate su di un protocollo che, di per sé, non lo è. TCP è orientato alla connessione, e permette quindi ai terminali che comunicano di monitorare lo stato della trammissione.
Sono molte le caratteristiche che contraddistinguono TCP, come ad esempio la possibilità di formattare i dati in segmenti di lunghezza variabile o di effettuare multiplazione dei dati (ossia di trasmettere simultaneamente informazioni da sorgenti diverse). Tra tutte, però, una funzione di TCP è di particolare interesse per lo studio delle prestazioni: la verifica del flusso di dati al fine di evitare saturazioni della rete.



Flusso di rete in un ciclo TCP

2 Controllo di Congestione in TCP

Il controllo della congestione è una funzionalità di TCP che permette di evitare la saturazione della rete. Consiste nel limitare la quantità di dati trasmessi sotto forma di pacchetti che non sono ancora stati riscontrati, permettendo in sostanza di adattare il flusso di dati all'attuale stato della rete. Lo stato di congestione in ogni istante viene determinato dai terminali stessi, che basano la loro valutazione su informazioni relative alla trasmissione dei pacchetti; non è quindi necessario raccogliere informazioni circa la situazione di rete dai suoi nodi o da dispositivi terzi.
Si incorre nel problema di congestione nel momento in cui il carico offerto ad una rete è maggiore di quello gestibile. Per evitare di inserire un nuovo pacchetto nella rete in questa situazione TCP si occupa di manipolare dinamicamente la dimensione delle finestre di trasferimento dati. Per fornire una trasmissione internet efficace è necessario tenere conto della capacità del ricevente, che è nota al mittente come Receiver Windows (RcvWin), e della capacità di rete; quest'ultima è descritta dalla Congestion Windows (CongWin). Gli algoritmi dediti al controllo della congestione fanno riferimento a queste variabili.

2.1 Funzionamento e Congestion Window

Il controllo di congestione opera diminuendo il tasso di immissione di nuovi segmenti ogni qualvolta si riscontri una perdita di pacchetti (associata allo scadere di un timeout di ritrasmissione). Se invece il mittente riceve degli ACK positivi la velocità di immissione dei pacchetti viene aumentata, aumentando inoltre la finestra di congestione. Il continuo adattamento del flusso di dati in base alle condizioni di saturazione della rete permette quindi di delineare, in ogni momento, la dimensione ottimale della cosiddetta "finestra di trasmissione". Esso è gestito in tre modalità differenti da TCP: Slow Start (l'evoluzione della CongWind ha un andamento esponenziale), Congestion Avoidance (evoluzione lineare) e Fast Recovery (dopo la ricezione di un triplice ACK non si ritorna in Slow Start ma, se si sono ricevuti dei riscontri, il recupero è più veloce).
Più accurato è l'algoritmo che si occupa del dimensionamento della finestra, più questa sarà simile alla finestra ottimale, ossia quella che permette ad ogni istante la massima trasmissione dei dati senza perdita di alcun pacchetto.

2.2 Algoritmi e versioni di TCP

Esistono diverse versioni di TCP, che basano il loro funzionamento su diversi algoritmi di Controllo della congestione. Alcuni di quelli storicamente più diffusi sono TCP-RENO e TCP-CUBIC, che saranno considerati come modelli di riferimento.

- Caratteristiche di TCP-RENO

  Questo algoritmo si caratterizza per una reazione distinta ai due eventi che, agli occhi di TCP, segnalano lo smarrimento di un pacchetto. Allo scadere del timeout viene associata una perdita particolarmente significativa, in seguito alla quale RENO fa ripartire la fase di Slow Start dal valore minimo di 1MSS. La ricezione invece di 3 ACK duplicati è considerata una perdita meno drastica, in seguito alla quale, RENO, ritrasmette velocemente il segmento contenente i dati già trasmessi, senza attendere lo scadere del timer ad essi associato.

- Caratteristiche di TCP-CUBIC

CUBIC, successore di TCP-BIC, può vantare maggiore rapidità di convergenza rispetto a RENO. La caratteristica che contraddistingue questo algoritmo è la funzione di crescita che, a differenza delle versioni standard di TCP, è una cubica. Nelle reti con lunghe latenze questo porta maggiore scalabilità e stabilità.
In un grafico rappresentante l'andamento di TCP-CUBIC si potrebbe individuare una prima fase caratterizzata da una notevole crescita (porzione concava della funzione) fino a raggiungere la dimensione della finestra ottenuta prima dell'ultimo evento di congestione. La porzione complessa è necessaria invece per individuare la larghezza di banda massima utilizzabile: la pendenza sarà inizialmente contenuta, per poi accentuarsi sempre di più (proprio seguendo l'andamento tipico di una funzione cubica). CUBIC trascorre molto tempo in una fase di stabilità, in attesa di ricercare maggiore banda, ovvero nelle regione compresa tra la porzione di crescita concava e quella convessa.

2.3 BBR (Bottleneck Bandwidth and Round-trip time)

Un nuovo algoritmo, già in uso su diverse piattaforme (ad esempio, su alcuni servizi di Google) è TCP-BBR, che sembra essersi guadagnato, in seguito a test svolti da Google e altri, la reputazione di una soluzione eccellente, migliore di qualsiasi altro suo predecessore.

- Caratteristiche di TCP-BBR

  BBR è l'acronimo di "Bottleneck Bandwidth and Round-trip time". Esso porta con se una novità nel campo degli algoritmi di congestione: non è più un controllo di congestione basato sulla perdità di pacchetti. Infatti, come scrive la stessa Google, il controllo della congestione basato sulla perdita di pacchetti è problematico nelle rete di oggi, caratterizzata da larghezze di banda ben maggiori delle reti presenti fin dagli anni '80. Nel caso di router con buffer piccoli si incorre in problemi di bufferbloat, con buffer grandi invece spesso si hanno bassi throughput causati da perdite interpretate erroneamente.
  Al contrario BBR considera quanto velocemente la rete stia fornendo dati, utilizzando un modello che comprende sia la bottleneck bandwidth, sia il suo RTT. BBR effettua continue stime di questi valori, garantendo così un controllo della congestione che reagisce ad una effettiva congestione e non alla perdita di pacchetti o al ritardo di coda.

3 Test e confronti

3.1 Valutazione delle prestazioni

Gli scenari di rete, quando si parla di Internet, sono molti e possono essere profondamente diversi tra loro: il numero degli host che condividono il medesimo canale, la qualità del canale stesso, la quantità di informazioni da trasmettere e molti altri fattori influenzano le prestazioni del protocollo di trasmissione.
Per condurre una valutazione oggettiva, quindi, bisogna partire dal presupposto che un algoritmo non può essere sempre ottimo, ma rivela le sue massime potenzialità in determinate condizioni. I test per confrontare gli algoritmi, quindi, andranno svolti variando i parametri che caratterizzano lo scenario.

3.2 Ambiente di simulazione

Le simulazioni sono state effettuate mediante software 'Oracle VM VirtualBox', X server 'Xming', terminale 'Putty' e macchina virtuale (OVF format, 32-bit, Mininet 2.2.2) con kernel Ubuntu aggiornato a 4.12.5 installati su un computer con sistema operativo Windows 10 a 64bit. In tutti i test eseguiti è stata implementata, mediante 'Mininet', una semplice rete composta da due host, uno switch ed un controller collegati tra loro con link Ethernet.

Per garantire una miglior accuratezza dei dati raccolti si è deciso di svolgere, nei casi più delicati, più prove nelle stesse identiche condizioni di simulazione. I risultati delle tre prove sono stati poi mediati algebricamente al fine di produrre un risultato più affidabile e verosimile.

Per caratterizzare di volta in volta l'ambiente di simulazione, al fine di riprodurre una serie quanto più esaustiva di scenari, si è agito sull'impostazione dei seguenti valori:

- Percentuale di dati persi (Loss)
- Ritardo temporale dei pacchetti (Delay)
- Limite di banda (Bandwidth)

I risultati raccolti nelle varie simulazioni sono raccolti nel capitolo 5 'Dati' e consultabili per aver un'idea più dettagliata degli esiti ottenuti al variare delle condizioni.

3.3 Test 1: Prestazioni al variare della percentuale di Loss

Il primo test è stato effettuato per studiare l'andamento dei tre protocolli di controllo della congestione di TCP presi in considerazione (Reno, Cubic e BBR) al variare della percentuale di pacchetti persi. L'analisi è stata effettuata sempre con un tempo fisso di trasmissione pari a 60 secondi, mentre la perdita percentuale è stata variata da 0, 1, 2 e 5. Per ogni conformazione di rete sono stati eseguiti tre diversi test per raggiungere un risultato medio più attendibile di una singola prova.

Risultati ottenuti

Secondo quanto riportato dagli studi teorici BBR mostra un andamento sensibilmente migliore, rispetto alle altre versioni di TCP, con una situazione di rete congestionata, quindi nonostante la perdita di pacchetti BBR dovrebbe riuscire a mantenere una quantità di dati trasferiti maggiore rispetto a CUBIC e RENO.
Le prove da noi effettuate hanno confermato quanto atteso a livello teorico; in assenza di perdite la quantità di dati trasferiti nelle tre versioni di TCP sono tra loro confrontabili essendo con lo stesso ordine di grandezza, come si può notare nel grafico 1 dove sia il rapporto tra BBR e CUBIC che quello con RENO è approssimativamente pari a 1.
Aggiungendo delle perdite, il risultato ottenuto è concorde con quanto ci si aspettava dagli studi, quindi i dati trasferiti da BBR sono sensibilmente maggiori rispetto a quelli inviati mediante le altre versioni di TCP prese in considerazione con si vede dal grafico 2. BBR riesce infatti a trasferire 100 volte più dati rispetto a CUBIC e RENO permettendo un traffico accettabile anche con delle perdite.
Sono stati successivamente eseguiti dei nuovi test per analizzare le prestazioni di TCP con una perdita elevata (10%); in questo caso i risultati di RENO e CUBIC sono tra loro discordanti. RENO si stabilizza ad un rapporto costante di dati trasferiti rispetto a BBR, mentre CUBIC continua a perdere sempre in rapporto con BBR.

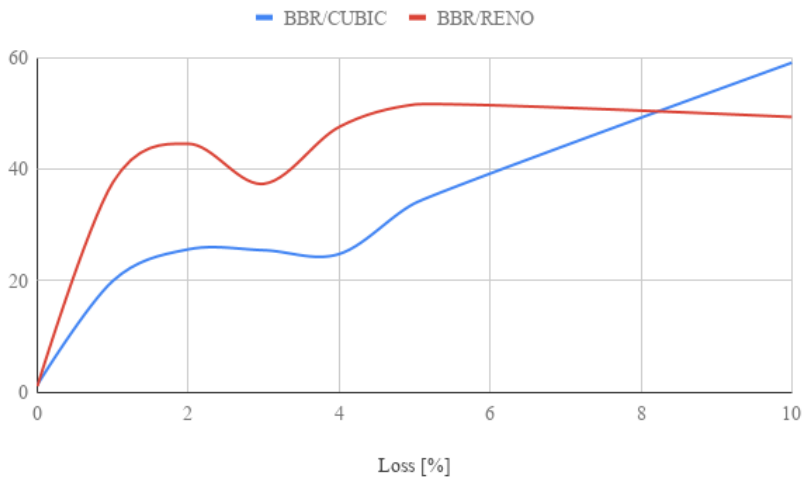


Grafico 1: *Rapporto dei dati trasferiti da BBR in relazione agli altri algoritmi.*

Quest'ultima prova è stata eseguita una sola volta, quindi il risultato ottenuto è meno affidabile rispetto ai test precedenti.
Tutte le versioni di TCP hanno, all'aumentare della percentuale di perdita, un andamento confrontabile con un esponenziale inverso; quello di BBR risulta più 'dolce', cala più lentamente rispetto agli altri due che sono tra loro molto simili.

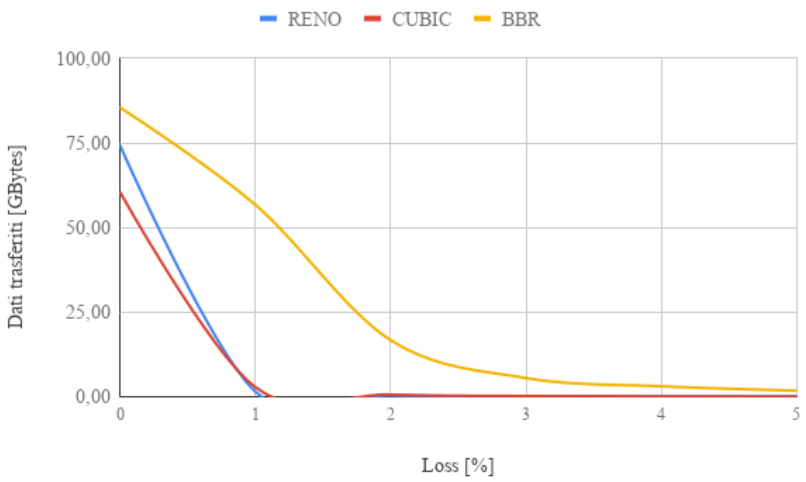


Grafico 2: *Dati trasferiti dai diversi algoritmi al variare della percentuale di Loss.*

3.4 Test 2: Prestazioni al variare dei parametri di Delay, Bandwidth e Loss

La seconda simulazione effettuata è volta a valutare le prestazioni delle versioni di TCP considerate al variare della banda, del ritardo o della perdita.
Alla banda, quando impostata manualmente, è stato assegnato un valore pari a 10.00 Mbit, al ritardo di 10 ms e la perdita al 5% dei pacchetti trasmessi.
Le prestazioni sono state valutate per ogni combinazione possibile delle tre variabili, in modo da poter attentamente valutare sotto quali condizioni BBR ha un andamento effettivamente vantaggioso rispetto a CUBIC e RENO.

Risultati ottenuti

Si nota immediatamente che BBR si distingue nella quasi totalità delle prove per una prestazione complessiva superiore agli altri due algoritmi; una buona parte di queste delinea inoltre un vantaggio quantitativamente schiacciante di BBR sui concorrenti.

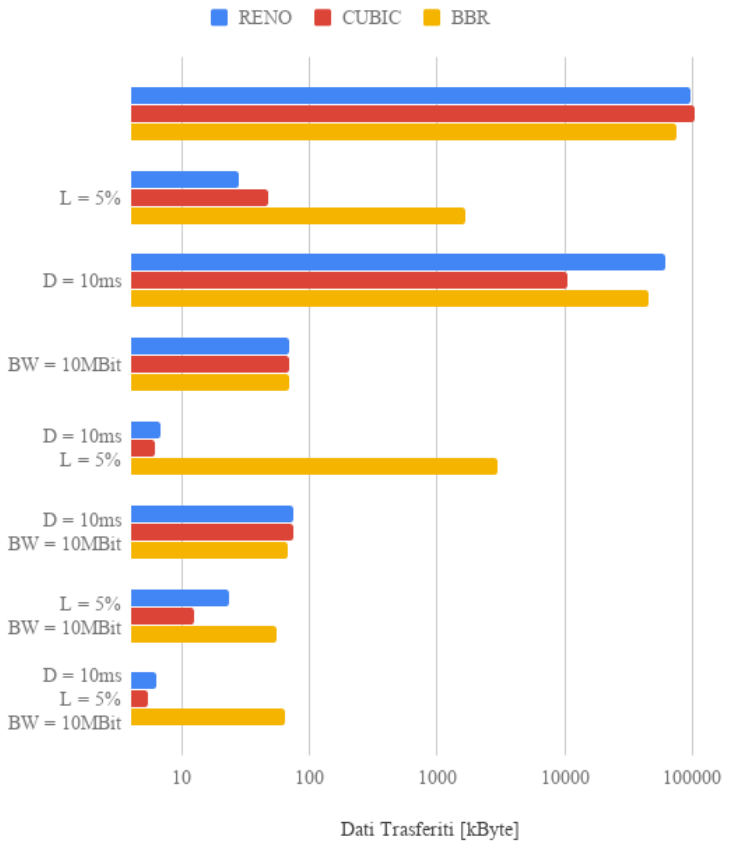


Grafico 3: *Dati trasferiti dai diversi algoritmi al variare dei parametri di simulazione.*

Se si volessero isolare i punti di forza di BBR sarebbe facile accorgersi che, in primis, le prove svolte con un tasso di perdita percentuale di dati favoriscono il neonato algoritmo rispetto alle storiche alternative: ogni qualvolta viene simulata la perdita del 5% dei pacchetti trasmessi, BBR fornisce una prestazione vincente che, comparata con CUBIC e RENO, risulta crescere rispetto a queste ultime in maniera esponenziale. É difficile invece decidere se la limitazione in Bandwidth e il ritardo temporale siano di fatto elementi che giocano a favore o contro BBR: in alcuni casi sembrano accentuare il suo vantaggio (apparentemente derivante però dal fattore di perdita), mentre altre prove danno l'idea che Bandwidth e Delay non influiscano sulle prestazioni o che addirittura tendano a livellarne i valori per i tre algoritmi.

3.5 Test 3: Prestazioni con Delay costante e con Bandwidth costante

Nella terza prova è stato impostato un ritardo costante di 1 secondi (misura adeguata rispetto alla durata del test di 60 secondi): sono state poi fatti variare altri parametri di partenza (inserimento di un limite di banda di 10 Mbit e di una perdita del 5%), ottenendo un totale di quattro prove (grafico 1). Infine (grafico 2) i tre algoritmi sono stati confrontati partendo da delay di 1 secondo e bandwidth pari a 10 Mbit, proponendo prove con differenti ritardi: delay a 0, 1, 2 e 5 secondi.

Risultati ottenuti

Dal grafico 1 emerge un risultato molto interessante. Le prestazioni di BBR non sono all'altezza degli altri due algoritmi nei casi in cui la rete è interessata solo da ritardo e bandwidth limitata: in tal caso gli algoritmi Reno e Cubic (le prestazioni sono paragonabili, Reno trasferisce una quantità di dati di un 5% maggiore rispetto a Cubic) risultano nettamente migliori dell'algoritmo progettato a Mountain View (quasi del 100% in più con una bandwidth di 10 Mbit, nel caso di Bandwidth non specificata i dati trasferiti da BBR sono circa di un quinto quelli di Reno). Nel caso di perdite la situazione si capovolge, mostrando una

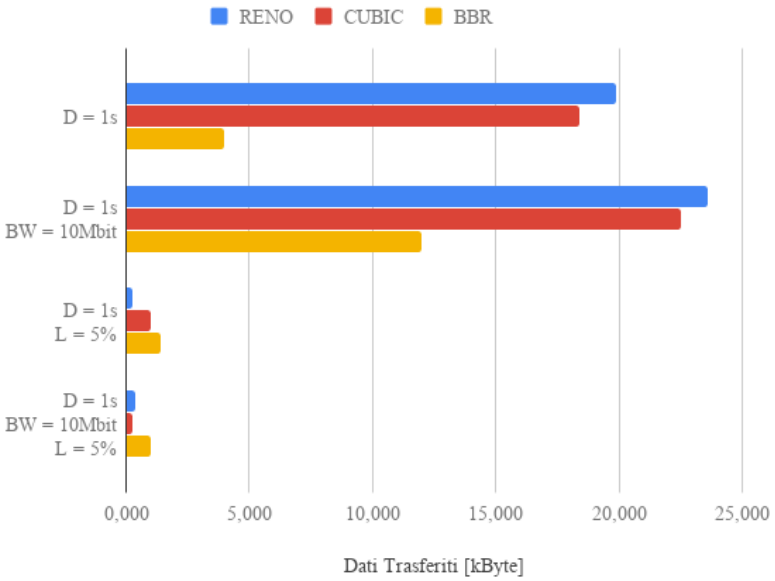


Grafico 4: *Dati trasferiti dai diversi algoritmi al variare dei parametri di simulazione.*

netta supremazia di BBR. Confrontando il caso con bandwidth non limitata e quello con limitazione di banda, il dato più interessante è il crollo di prestazioni di Cubic. Nel primo caso i risultati ottenuti era paragonabili, se pur inferiori, a quelli di BBR (Reno otteneva i risultati peggiori, circa 1/4 di cubic e 1/5 di BBR). Nel secondo invece Cubic scende in fondo alla classifica; Reno migliora sensibilmente (aumenta quasi del 100%), BBR invece cala leggermente. I test effettuati nel grafico 2 mostrano come l'algoritmo BBR sia in grado

di lavorare meglio dei concorrenti in condizioni di maggiori perdite. BBR infatti risulta migliore degli altri due versioni di TCP una volta inserite delle perdite.

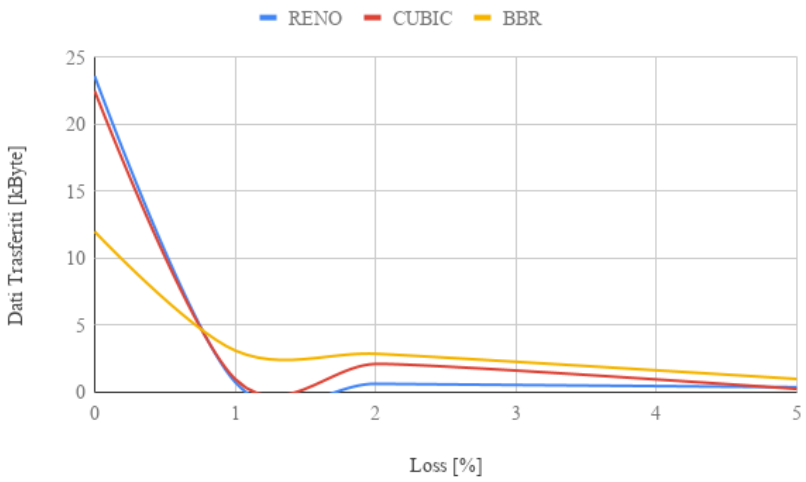


Grafico 5: *Dati trasferiti dai diversi algoritmi al variare della percentuale di Loss.*

3.6 Test 4: Prestazioni con limite elevato di banda

L'ultimo test eseguito mirava a valutare le prestazioni di TCP ponendo un limite di banda elevato (1 Gbit) e confrontando i tre algoritmi in due situazioni differenti: un caso caratterizzato da assenza di ritardo e un altro in cui il delay è di 1 secondo. Tutte le prove sono state inoltre caratterizzate da una perdita del 5%.

Risultati ottenuti

BBR è l'algoritmo che riesce a lavorare in modo migliore con limiti di banda elevati. Nel caso caratterizzato da assenza di ritardo le differenze tra i BBR ed i due concorrenti sono molto marcate: la sua capacità di sfruttare meglio la banda a propria disposizione lo porta infatti a scambiare una quantità di dati quasi cinqua volte superiore a Cubic, a sua volta capace di spostare il triplo dei dati di Reno.
Impostando il ritardo di 1 secondo le differenze si assottigliano notevolmente, pur mostrando ancora una netta supremazia di BBR. Anche in questo caso Cubic lavora meglio di Reno.

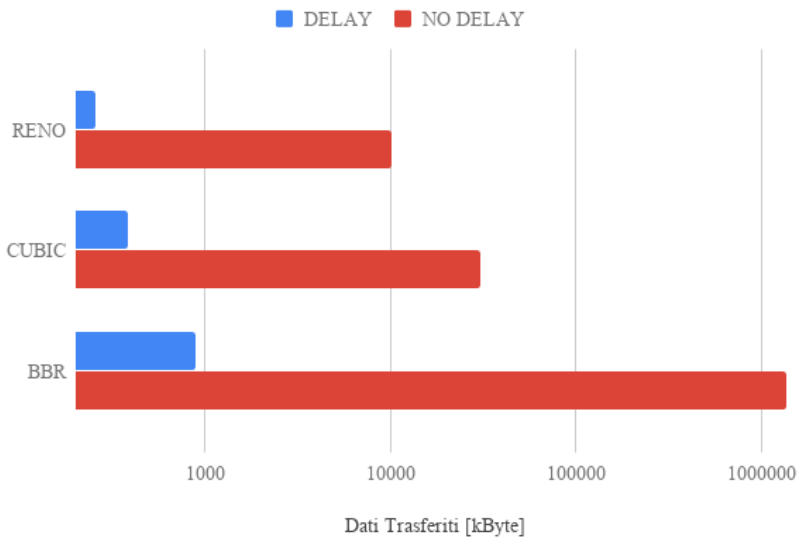


Grafico 6: *Prestazioni degli algoritmi con e senza Delay.*

4 Conclusioni

I risultati da noi ottenuti sono piacevolmente concordi con le aspettative. BBR mostra un andamento analogo, e a volte peggiore, rispetto ai suoi colleghi nelle situazioni ideali, quindi senza perdite; quando invece la rete si avvicina alla realtà, quindi con un Loss diverso da zero, BBR mostra prestazioni nettamente migliori nei confronti di RENO e CUBIC. Secondo le simulazioni da noi effettuate quindi BBR si dimostra un'ottima alternativa alle classiche versioni di 'Congestion Control'. La continua stima di bandwidth disponibile effettuata da BBR gli permette di inviare, a differenza dei suoi concorrenti, un numero molto più elevato di dati anche quando la rete mostra delle perdite. In caso di mancata ricezione di un pacchetto RENO e CUBIC necessitano di molto tempo prima di riprendere a trasmettere con un'elevata capacità, mentre l'essere adattivo di BBR gli permette di regolare in autonomia la quantità di dati da inviare per non sovraccaricare il collegamento o per non sottostimarlo.

Appurato che la percentuale di Loss è di fatto il dato che influenza maggiormente le prestazioni di BBR, si può affermare che le condizioni medie in cui la rete internet opera sono a favore del nuovo algoritmo. S'intende che, mediamente, gli utenti che usufruiscono della rete globale navigano in condizioni di Loss simili a quelle in cui, secondo gli esperimenti svolti, BBR fornisce prestazioni migliori. Una stima precisa di quale sia il Loss medio nella navigazione Internet è molto complicata da compiersi, ma facile osservare che l'ordine di grandezza rientra appunto nel range a favore di BBR ($1\% \div 8\%$).

5 Dati

5.1 Test 1

Tempo sim. [s]	Loss [%]	RENO		CUBIC		BBR	
		Bandwidth [Gbit/s]	Dati trasf. [GBytes]	Bandwidth [Gbit/s]	Dati trasf. [GBytes]	Bandwidth [Gbit/s]	Dati trasf. [GBytes]
60	0	12,300	86,100	10,600	74,100	10,300	72,100
		9,360	65,400	7,960	55,600	13,600	94,900
		10,300	71,700	7,450	52,100	12,800	89,500
		10,653	74,400	8,670	60,600	12,233	85,500
60	1	0,200	1,400	0,373	2,620	7,660	53,600
		0,221	1,550	0,425	2,990	7,730	54,000
		0,226	1,580	0,415	2,910	8,960	62,700
		0,216	1,510	0,404	2,840	8,117	56,767
60	2	0,055	0,398	0,098	0,708	2,780	19,400
		0,052	0,375	0,092	0,668	2,360	16,500
		0,049	0,354	0,081	0,582	2,050	14,300
		0,052	0,376	0,091	0,653	2,397	16,733
60	5	0,004	0,030	0,006	0,044	0,170	1,200
		0,005	0,040	0,008	0,061	0,277	1,930
		0,005	0,036	0,008	0,056	0,328	2,300
		0,005	0,035	0,007	0,053	0,258	1,810

5.2 Test 2

Tempo sim. [s]	Delay [ms]	Bandwidth [Mbit]	Loss [%]	RENO		CUBIC		BBR	
				Bandwidth [Gbit/s]	Dati trasf. [GBytes]	Bandwidth [Gbit/s]	Dati trasf. [GBytes]	Bandwidth [Gbit/s]	Dati trasf. [GBytes]
60	0	No limit	0	13,800	96,500	15,700	110,000	12,300	85,800
				13,400	93,300	15,200	106,000	10,300	72,000
				14,800	103,000	13,500	94,400	9,320	65,100
				14,000	97,600	14,800	103,467	10,640	74,300
60	0	No limit	5	0,003	0,025	0,005	0,037	0,223	1,580
				0,004	0,027	0,008	0,057	0,298	2,100
				0,004	0,031	0,006	0,048	0,182	1,350
				0,004	0,028	0,006	0,047	0,234	1,677
60	10	No limit	0	8,850	61,800	1,500	10,500	6,570	45,900
				8,940	62,500	1,460	10,200	6,590	46,100
				8,700	60,800	1,500	10,400	6,490	45,300
				8,830	61,700	1,487	10,367	6,550	45,767
60	0	10	0	0,010	0,069	0,010	0,069	0,010	0,069
				0,010	0,069	0,010	0,069	0,009	0,069
				0,010	0,069	0,010	0,069	0,010	0,069
				0,010	0,069	0,010	0,069	0,010	0,069
60	10	No limit	5	0,001	0,006	0,001	0,006	0,415	2,910
				0,001	0,007	0,001	0,006	0,433	3,050
				0,001	0,007	0,001	0,006	0,425	2,970
				0,001	0,007	0,001	0,006	0,424	2,977
60	10	10	0	0,010	0,078	0,010	0,074	0,009	0,068
				0,010	0,074	0,010	0,074	0,009	0,068
				0,010	0,074	0,010	0,075	0,009	0,068
				0,010	0,075	0,010	0,074	0,009	0,068
60	0	10	5	0,003	0,023	0,002	0,014	0,008	0,055
				0,003	0,023	0,002	0,013	0,008	0,056
				0,003	0,024	0,001	0,011	0,008	0,055
				0,003	0,023	0,002	0,013	0,008	0,055
60	10	10	5	0,001	0,006	0,001	0,005	0,009	0,064
				0,001	0,007	0,001	0,006	0,009	0,063
				0,001	0,006	0,001	0,005	0,009	0,064
				0,001	0,006	0,001	0,005	0,009	0,063

5.3 Test 3

Tempo sim. [s]	Delay [s]	Bandwidth [Mbit]	Loss [%]	RENO		CUBIC		BBR	
				Bandwidth [Mbit/s]	Dati trasf. [MBytes]	Bandwidth [Mbit/s]	Dati trasf. [MBytes]	Bandwidth [Mbit/s]	Dati trasf. [MBytes]
60	1	0	0	2,310	19,900	2,270	18,400	0,493	4,000
60	1	10	0	2,680	23,600	2,820	22,500	1,390	12,000
60	1	0	5	0,007	0,256	0,051	1,000	0,129	1,380
60	1	10	1	0,058	0,768	0,105	1,000	0,355	3,120
60	1	10	2	0,036	0,640	0,212	2,120	0,296	2,880
60	1	10	5	0,013	0,384	0,020	0,256	0,116	1,000

Tempo sim. [s]	Delay [s]	Bandwidth [Mbit]	Loss [%]	RENO		CUBIC		BBR	
				Bandwidth [Mbit/s]	Dati trasf. [MBytes]	Bandwidth [Mbit/s]	Dati trasf. [MBytes]	Bandwidth [Mbit/s]	Dati trasf. [MBytes]
60	1	1.000	5	0,007	0,256	0,035	0,384	0,131	0,896
60	0	1.000	5	1,340	10,200	4,130	30,400	194,000	1.370,000