Reti di Calcolatori
Homework 2: Strumenti per valutazione prestazioni di rete

Marco Cagnazzo

10 maggio 2023

Comando `ping`

L'applicazione `ping` è supportata da tutti i sistemi operativi comuni ed è ampiamente utilizzata come strumento elementare di diagnostica di rete. `ping` sfrutta il protocollo Internet Control Message Protocol (ICMP), che si appoggia direttamente sull'Internet Protocol (IP), senza coinvolgere alcun protocollo di livello trasporto.
In breve, l'applicazione `ping` invia una serie di pacchetti a una destinazione, la quale risponde con pacchetti della stessa dimensione. L'applicazione misura e visualizza il cosiddetto *Round-Trip Time* (RTT) per ogni pacchetto, ovvero il tempo che intercorre tra l'invio della richiesta e la ricezione della risposta. È possibile specificare la dimensione del pacchetto a livello di rete: chiamiamo tale quantità L (misurata in bit). Consideriamo il caso in cui il comando `ping` è ripetuto K volte per ogni valore di L e sia $RTT(L,k)$ il valore del round-trip time alla k-esima esecuzione del `ping` con un pacchetto di L bit. Il singolo valore osservato di RTT corrisponde alla somma dei ritardi su ogni singolo link del percorso da sorgente a destinazione e da destinazione a sorgente. Il tempo necessario per attraversare l'i-esimo link è:

$$d_i = d_{i,\text{queue}} + d_{i,\text{proc}} + d_{i,\text{prop}} + d_{i,\text{trans}}$$

dove i diversi contributi si valutano come segue:

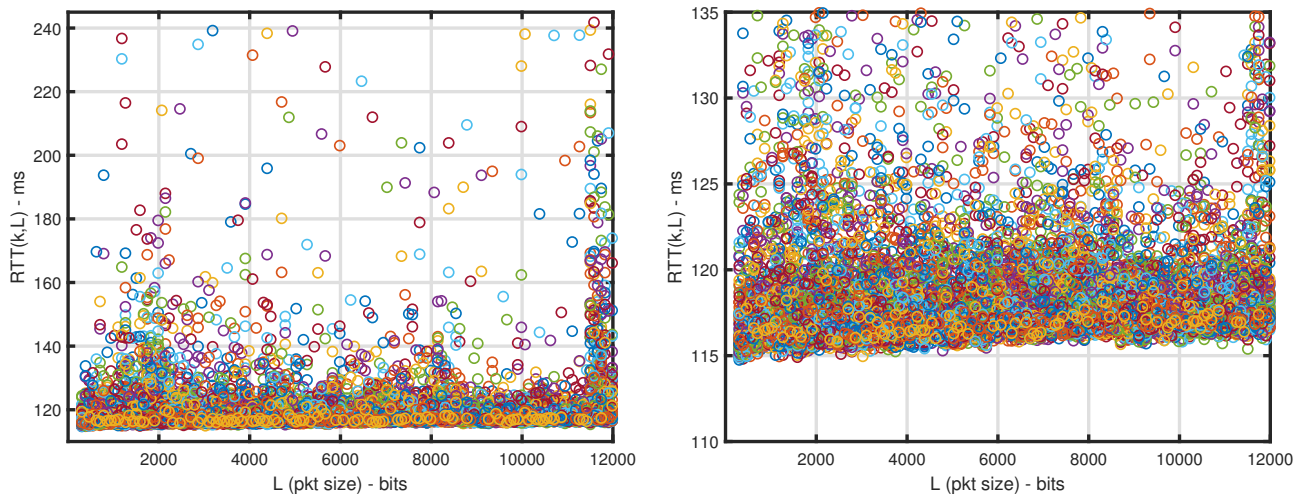
- $d_{i,\text{queue}}$ è il tempo speso dal pacchetto nel buffer di trasmissione del link $i$esimo. Alla k-esima istanza del comando `ping` tale tempo può variare e si indica quindi come $q_i(k)$
- $d_{i,\text{proc}}$ è il tempo utilizzato dagli algoritmi di routing per decidere l'instradamento del pacchetto ed è tipicamente trascurabile rispetto agli altri, ma anche difficile da modellare matematicamente
- $d_{i,\text{prop}}$ è il tempo di propagazione del segnale nel mezzo. Si può considerare costante nel tempo su un dato link i, lo indicheremo quindi con τ_i
- $d_{i,\text{trans}}$ detto tempo di trasmissione è la durata del segnale che rappresenta il pacchetto. Esso dipende dalla dimensione L del pacchetto a livello di rete e dal throughput S_i del link sempre al livello di rete. Scriveremo quindi $d_{i,\text{trans}} = \frac{L}{S_i}$

Numeriamo da 1 a n i link lungo il percorso **di andata e ritorno** dei pacchetti. Per esempio, se il percorso è $A \to B \to C$, ci saranno 4 link: da A a B, da B a C, da C a B e da B ad A.
Alla k-esima (su K) esecuzione del `ping` con un pacchetto di L bit, si osserva un RTT:

$$RTT(L,k) = \sum_{i=1}^{n} d_i = \sum_{i=1}^{n} \left(\frac{L}{S_i} + q_i(k) + \tau_i \right) = \left(\sum_{i=1}^{n} \frac{1}{S_i} \right) L + \sum_{i=1}^{n} q_i(k) + \sum_{i=1}^{n} \tau_i \tag{1}$$

I valori osservati di RTT dipendono dunque dalla lunghezza L del pacchetto, ma a parità di L, ogni esecuzione della misura può dar luogo a valori osservati diversi a causa dei ritardi di accodamento che sono variabili, come si può osservare in figura 1(a) dove abbiamo riportato gli tutti RTT osservati con pacchetti di lunghezza da 10 a 1450 byte inviati ($K = 100$) al server `atl.speedtest.clouvider.net`
Poniamo ora

$$a = \sum_{i=1}^{n} \frac{1}{S_i} \qquad Q(k) = \sum_{i=1}^{n} q_i(k) \qquad T = \sum_{i=1}^{n} \tau_i$$



(a) (b)

Figura 1: Valori di RTT misurati su pacchetti da 10 a 1450 byte. Per ogni valore di L sono stati effettuate $K = 100$ misure. A destra uno zoom sulle ascisse da 110 a 135.

l'RTT si esprime come una funzione affine della lunghezza del pacchetto L:

$$RTT(L,k) = aL + Q(k) + T$$

Pertanto, l'RTT tendenzialmente cresce con la lunghezza dei pacchetti, ma nella pratica questa relazione viene mascherata dalla variabilità del ritardo di accodamento $Q(k)$. Tuttavia, ci si può disfare dell'impatto dei ritardi di accodamento considerando **il valore minimo di RTT** che si misura su una serie di trasmissioni con pacchetti di **dimensione costante**. Infatti si ipotizza che prima o poi il pacchetto possa trovare tutte le code vuote ad ogni nodo. In formule, assumeremo che, eseguendo un numero K sufficientemente grande di volte il `ping` con L costante, si abbia:

$$RTT_{\min}(L) = \min_{k\in\{1,2,\ldots,K\}} RTT(L,k) \approx aL + T \tag{2}$$

In effetti, dalla Fig. 1(b), si osserva che l'andamento del minimo dell'RTT per ogni valore di L è approssimativamente una retta. In Fig. 2(a) si mostra l'andamento dell'RTT minimo in funzione della dimensione del pacchetto e si traccia anche la migliore approssimazione lineare. Stimando la pendenza di tale retta, per esempio col metodo dei minimi quadrati, si ottiene una stima di $a = \sum_{i=1}^{n} \frac{1}{S_i}$. A sua volta, da questo valore, si possono ottenere delle stime del throughput in due ipotesi:

- Se tutti i link (andata e ritorno) hanno throughput uguali ad un certo valore S, si ottiene:

$$a = \sum_{i=1}^{n} \frac{1}{S_i} = \frac{n}{S} \qquad\qquad S = \frac{n}{a}$$

- Se invece esiste un link con un throughput **molto minore** di tutti gli altri (detto *bottleneck*), e supponendo che tale throughput sia lo stesso all'andata ed al ritorno si ha

$$a = \sum_{i=1}^{n} \frac{1}{S_i} \approx \frac{2}{S_{\text{bottleneck}}} \qquad\qquad S_{\text{bottleneck}} \approx \frac{2}{a}$$

Specifiche

In questo HW useremo il modello matematico qui descritto per stimare il throughput di una connessione. Le specifiche dell'homework sono le seguenti.

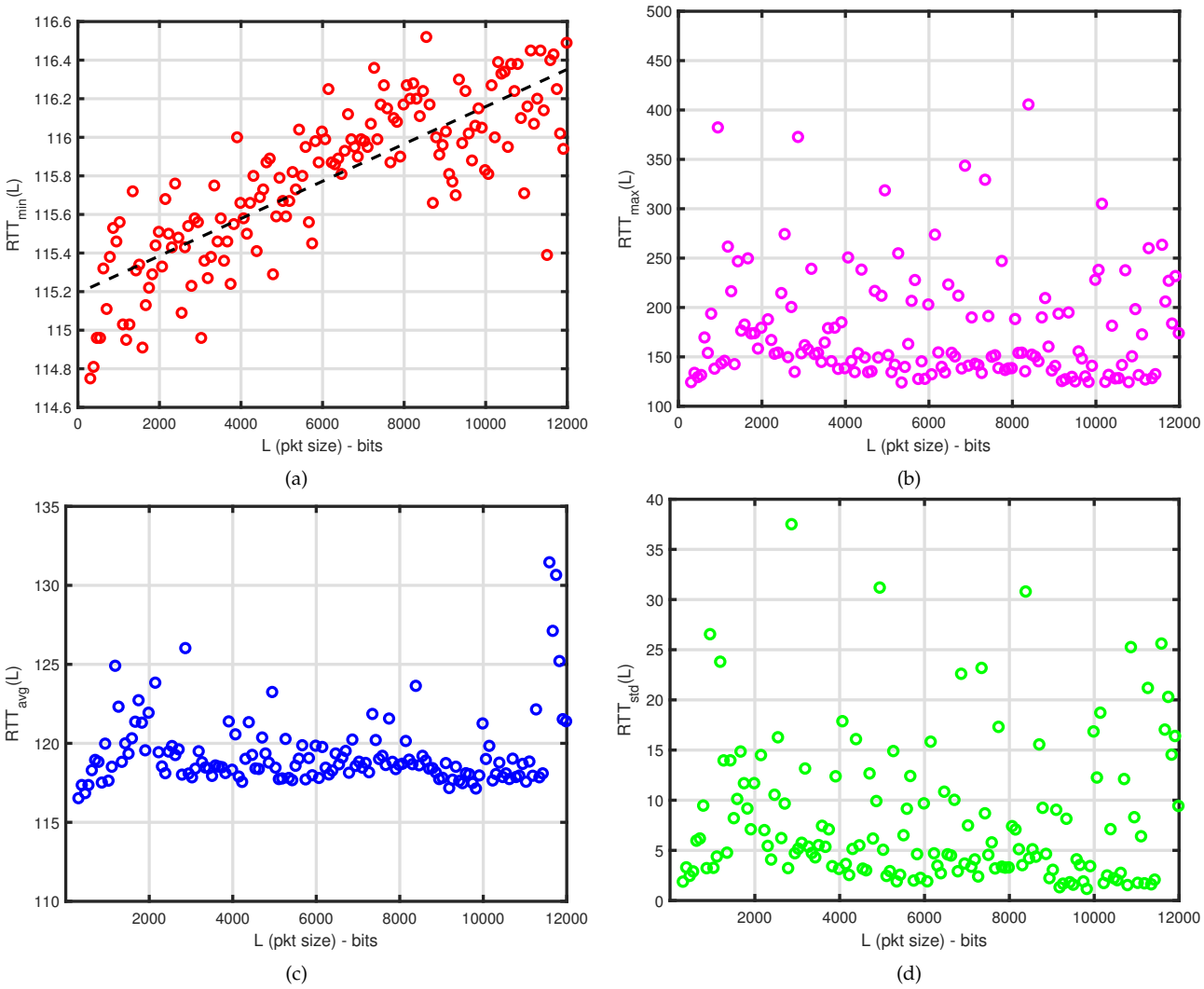


Figura 2: Statistiche dell'RTT in funzione della dimensione del pacchetto: (a) RTT minimo (b) massimo (c) media (d) deviazione standard

1. Comprensione del comando `ping`: Si usi l'help dell'applicazione `ping` e si individuino le opzioni più utili per valutare le prestazioni di una connessione.
   Nota 1: le opzioni possono variare in base al sistema operativo. In particolare, sui computer con Windows può essere utile installare una versione più flessibile di quella disponibile con il S.O., per esempio `psping`
   Nota 2: alcune opzioni potrebbero essere disponibili solo se si possiedono i diritti di amministratore sulla macchina.

   Si verifichi quindi l'esistenza e il formato delle opzioni che consentono di attivare le seguenti funzionalità:

   - Definire il numero K di pacchetti spediti per ogni sessione ping.
   - Specificare la dimensione L_{byte} del payload del pacchetto. Si noti che la dimensione del pacchetti IP che trasporta il messaggio è maggiore in quanto include i 20 byte di header IP e gli 8 byte di header ICMP. Si ricordi inoltre che L è misurato in bit. Quindi $L = 8 \cdot (L_{\text{byte}} + 28)$
   - Definire il Time-To-Live (TTL). Questo è un campo dell'header del protocollo IP. Viene impostato dal mittente e decrementato di una unità ogni qual volta il pacchetto viene inoltrato da un nodo intermedio. Se il nodo che riceve il pacchetto con un TTL di 1 non è il destinatario finale, il pacchetto viene scartato. Pertanto, il TTL determina il numero massimo di link da attraversare. Se il pacchetto viene scartato prima di raggiungere la destinazione l'applicazione riporta un messaggio di errore.

2. Si eseguano delle sessioni di `ping` verso uno dei server della lista data più in basso. Può essere conveniente usare un qualsiasi linguaggio di script (`bash`, `perl`, ..., ma anche Matlab o Python) per gestire automaticamente le sessioni e per raccogliere ed elaborare i risultati. Si determini quanto segue:

   - il numero di link attraversati. A questo fine, si sfrutti l'opzione che permette di variare il campo TTL (Time To Live), e lo si decrementi finché l'applicazione non segnala errore. Si confronti il risultato con quanto si può ottenere con il comando `traceroute` (Linux) o `tracert` (Windows). Attenzione, nell'equazione (1) n include anche il percorso di ritorno, quindi è il doppio del valore trovato con TTL o con `traceroute`
   - l'andamento del RTT minimo, massimo e medio, e la deviazione standard del RTT, in funzione della dimensione totale del pacchetto L. La dimensione del payload deve variare tra 10 e 1472 byte (dimensione massima che permette al pacchetto di attraversare una rete Ethernet senza essere frammentato). La scelta del numero di valori di L da testare in questo intervallo rimane allo studente. Per ogni valore di L, si eseguano K istanze. La figura 2 mostra un esempio dei risultati attesi: l'andamento di RTT minimo, massimo, medio e deviazione standard per il server `atl.speedtest.clouvider.net`
   - usando un metodo numerico (per esempio, `Polynomial.fit` in numpy o `polyfit` in Matlab) stimare il coefficiente a a partire dagli RTT minimi (Eq. (2)), e usarlo per stimare il throughput nel caso di link identici ed il throughput minimo nel caso di bottleneck.

   Suggerimento: si consiglia di salvare l'output del comando ping su un file per poi poter procedere all'elaborazione usando un foglio di calcolo oppure le espressioni regolari. A tal scopo si può ridirigere l'output del comando direttamente su un file con l'opzione ">". Ad esempio, il comando:
   `ping www.repubblica.it > risultati_repubblica.txt`
   crea un file testuale "risultati_repubblica.txt" contenente l'output del comando ping.

   Lista possibili server.

`atl.speedtest.clouvider.net`	Atlanta
`nyc.speedtest.clouvider.net`	New York City
`lon.speedtest.clouvider.net`	London
`la.speedtest.clouvider.net`	Los Angeles
`paris.testdebit.info`	Paris
`lille.testdebit.info`	Lille
`lyon.testdebit.info`	Lyon
`aix-marseille.testdebit.info`	Aix-Marseille
`bordeaux.testdebit.info`	Bordeaux

Consegna

Si consegneranno due file:

- Un file zip con tutto il codice prodotto per effettuare i test
- Un file pdf che descriva il lavoro svolto:
  1. Descrizione su come lanciare il codice (sistema operativo, linguaggio di scripting etc.)
  2. Descrizione dei parametri dell'esperimento: server, numero di istanze K, dimensioni dei pacchetti, altre eventuali opzioni del ping
  3. Stima del numero di link attraversati
  4. Andamento dell'RTT minimo, medio, massimo e della deviazione standard in funzione della dimensione del pacchetto
  5. Stima di R e di $R_{\text{bottleneck}}$
  6. Discussione dei risultati ottenuti.