Performance Modeling of Computer Systems and Networks

Prof. V. de Nitto Personè

AA 2019/2020

Progetto COVID-19

1. Si individui un sistema a scelta oggetto di studio;
2. Si individuino gli obiettivi dello studio;
3. Si costruisca un modello di simulazione seguendo i passi dell'algoritmo 1.1.1 e 1.1.2 del libro di testo[1];
4. Gli esperimenti di simulazione devono includere:
    a. Una fase di analisi transiente;
    b. Una fase di analisi dello stato stazionario; nel caso in cui il sistema non fosse stazionario o se l'obiettivo dello studio non prevede questo tipo di analisi, andrà mostrato soltanto l'evidenza che la stazionarietà non viene raggiunta;
5. I risultati devono essere mostrati sia in forma grafica sia in forma numerica mediante tabelle riassuntive degli stessi.

La simulazione deve essere svolta considerando come linee guida quanto riportato in [1].

A titolo esemplificativo si riportano i progetti di A. A. precedenti, nelle versioni per gruppo.
Casi di studio potrebbero essere derivati da:

- Esperienze personali, anche di studio all'interno di altri corsi;
- Esempi di sistemi descritti nei libri di testo[1,2]; ancora a titolo esemplificativo: la rete packet-switched dell'es. 17.2 affrontato a lezione, i casi reali riportati dallo stesso libro (ad es. nei capp. 10, 24).

Partecipanti al progetto

Il progetto può essere sviluppato individualmente o in gruppo di massimo 3 persone.
In caso di progetto di gruppo, lo studio deve includere un algoritmo migliorativo dell'obiettivo che corrisponda ad una evoluzione del modello, ripetendo i passi sopra descritti.

Griglia di valutazione

L'elaborato finale verrà valutato in base a:

- significatività e pertinenza del caso di studio in base al corso e a quanto richiesto;
- chiarezza e capacità di sintesi nella descrizione;
- completezza dello studio.

Elaborati da consegnare

Dovrà essere consegnata una relazione in formato pdf, che includa la descrizione del sistema, dell'obiettivo e dei risultati ottenuti, come visto durante il corso e nello stesso algoritmo di sviluppo del modello. Riguardo l'implementazione, andranno descritte le scelte fatte al riguardo delle parti essenziali e

---

[1] Lawrence M. Leemis, Sthephen K. Park, *Discrete-Event Simulation - A first course*, Pearson Education Prentice Hall, 2006

[2] M. Harchol-Balter, Performance Modeling and Design of Computer Systems, Cambridge, University Press, 2013

critiche della simulazione. Andrà consegnato anche il codice sviluppato dallo studente, omettendo le librerie di sistema.

Presentazione
La presentazione sarà individuale e dovrà durare, da parte dello studente, non più di 10 min. Durante la stessa il docente potrà chiedere di mostrare delle esecuzioni online.

[1] S. Kurkowski, T. Camp, M. Colagrosso, *MANET Simulation Studies: The Incredibles*, Mobile Computing and Communications Review, Volume 9, Number 4

Performance Modeling of Computer Systems and Networks

Prof. V. de Nitto Personè

Project

Mobile devices are still limited in computing capabilities and battery lifetime. The problem of enhancing user experience may find a solution in cloud computing.
Consider a "two-layer" cloud system, consisting of an edge cloud server (cloudlet) and a remote cloud server, where the cloudlet is at "one-hop" distance from a set of mobile devices. Applications running on these mobile devices autonomously select some of their tasks for offloading to an external server (e.g., because of performance or energy saving reasons), and send an **offloading request** to a **controller** located on the cloudlet. Upon receiving a request, the controller takes a decision about whether the task should be sent to the cloudlet or the cloud, with the goal of minimizing the mean response time.
Typically, tasks hosted by the cloud server benefit of a higher execution rate, but suffer for greater network delay. We assume that the remote cloud server has virtually unlimited resources, hence it is always able to guarantee absence of interference among any number of tasks allocated to it. On the other hand, the cloudlet has limited resources, so that it is able to guarantee absence of interferences among tasks allocated to it as long as their number does not exceed a given threshold N.
Let us assume users belong to two classes and denote with n_i the number of class i tasks in execution on the cloudlet. The controller can take the following simple decision upon task arrival events:

Algorithm 1

- arrival:
  if n1+ n2=N → send on the cloud
  else accept the task on the cloudlet

By considering that execution on the cloudlet of a class 1 task is more convenient than execution of a class 2 task, the controller can set a given threshold $S \leq N$ and can use the following access control algorithm:

Algorithm 2

- class 1arrival:
  if n1=N → send on the cloud
  else if n1+n2<S → accept
  else if n2 > 0 → accept the task on the cloudlet and send a class 2 task on the cloud
  else accept the task on the cloudlet
- class 2 arrival:
  if n1+n2≥S → send on the cloud
  else accept the task on the cloudlet

When a class 2 task is *interrupted* and sent on the cloud, a setup time s_{setup} has to be considered to restart the task on the cloud.

Consider the following parameters:

- the mean arrival rates for the two classes are:
  λ_1= 4 task/s, λ_2= 6.25 task/s
- services on cloud can be assumed exponential, with mean rates:
  μ_{1cloud}= 0.25 task/s and μ_{2cloud}= 0.22 task/s
  while services on cloudlet are hyperexponential with p=0.2 for both classes and
  μ_{1clet}= 0.45 task/s and μ_{2clet}= 0.27 task/s.

Note that, as we stated above, the service time includes the transmission time. For this reason, we assume $\mu_{iclet} > \mu_{icloud}$ for both classes. Moreover, execution on the cloudlet of a class 1 task is more convenient than the execution of a class 2 task.
For Algorithm 2, we assume an exponential time with mean $E[s_{setup}]$= 0.8 s.

a.1. Design a next-event simulator for the system above with access control *Algorithm 1* and N=20.
a.2. Determine if the system is stationary or not and design the experiments accordingly.

a.3. Evaluate:
    1. the *system* response time and throughput both global and per-class;
    2. the per-class effective[3] cloudlet throughput;
    3. the per-class cloud throughput;
    4. the *class* response time and mean population, both for the cloudlet and for the cloud;
    5. Analyze the transient system statistics and in case of existence the steady-state statistics with the appropriate approach respectively.

b.1. Define a queueing model for the whole system.
b.2. Derive an analytical solution[4] and validate the simulation results.

c.1. Implement *Algorithm 2* and determine a threshold S to minimize the response time. (Hints: you can start with S=N/2 and try 2 different values for S.)
c.2. Evaluate indices 1., 2., 3., 4., in a.3 and
    6. the percentage of class 2 *interrupted* tasks and their response time.

    Please explain how you manage with class 2 interrupted tasks, their service and response time.
c.3. Compare all results with those obtained with *Algorithm 1*.
c.4. Modify queueing model in step b.1. to include *Algorithm 2*.
c.5. Derive an analytical solution[2] and validate the simulation results.

The steady state or transient statistics should be computed with a 95% confidence level. Please, consider as a guideline the paper "MANET Simulation Studies: The Incredibles", S. Kurkowski, T. Camp, M. Colagrosso, *Mobile Computing and Communications Review*, Volume 9, Number 4.

The project must be delivered at least one week before the oral test.
The students should prepare a presentation of the obtained results. The time should not exceed 10 minutes per student in a group (20 minutes for a student alone).

Documents to be delivered:

- The written report should include: conceptual and analytical models, the methodology used for estimating the statistics, the results, both graphical and numerical form, with the relative comments
- the listing code.

Evaluation grid

- clarity
- pertinence
- completeness.

[3] For "effective" throughput we mean the completed tasks rate.
[4] You can consider exponential distributions for cloudlet too.

Performance Modeling of Computer Systems and Networks

Prof. V. de Nitto Personè

Project - 9CFU

Consider a system that provides web services and which is built according to the widely used multi-tiered paradigm. Typically, the multi-tier application consists of a web server, an application server, and a back-end database. The web server and the application server reside usually within the same physical server, which is called front server. Access to a web service occurs in the form of a *session* consisting of many individual requests. After a new session connection is generated, client requests circulate among the front and database server before they are sent back to the client. After a request is sent back, the client spends a time before sending the following request. A session completes after the client has generated a series of requests.
For a customer trying to place an order, or a retailer trying to make a sale, the real measure of a web server performance is its ability to process the *entire* sequence of requests needed to complete a transaction.
The *useful throughput* of the system is measured as a function of the number of completed sessions. Aborted requests of already accepted sessions are highly undesirable because they compromise the server's ability to process all requests needed to complete a transaction and result in wasted system resources. The percentage of refused new connection requests in respect of the total arrival flow is called new session *drop ratio*. The percentage of refused requests in respect of the total requests of existing sessions (on-line requests) is called *aborted ratio*.

*

Consider a queueing network model for the system above:

1. describe the topology, the system state and the model to realize the session.
2. Assume the following parameters:
   - inter-arrival times of new session requests exponentially distributed with rate 35 requests/s;
   - session length *Equilikely*(5, 35);
   - E[Z]=7 s exponentially distributed (client average think time);
   - $E[D]_{\text{front server}}$=0.00456 s exponentially distributed;
   - $E[D]_{\text{back-end server}}$=0.00117s exponentially distributed.
3. Evaluate if the stability condition holds.
4. According to what stated at point 3, compute the following steady state or transient statistics with a 95% confidence level: the system response time, the system useful throughput and the response time for each center.
5. Evaluate the linear correlation between the *delay* and the *wait* for the front server; show the bivariate scatterplot and the correlation coefficient.
6. Implement an *overload management mechanism* based on monitoring the CPU utilization of the web server as follows: if the observed utilization is above 85%, then for the next time interval the admission controller rejects <u>all new session requests</u> and only serves requests from already admitted sessions. Once the observed utilization drops below the 75%, the admission controller changes its policy for the next time interval and begins admitting and processing new sessions again.
   a. Describe the method used to estimate the web server utilization
   b. Evaluate the appropriate statistics (transient or steady state) with a 95% confidence level for the system including the response time, the useful throughput, the drop and the aborted ratio.
7. Verify both the simulation models by means of analytical models.

The project must be delivered at least one week before the oral test.
The students should prepare a presentation of the obtained results. The time should not exceed 10 minutes per student in a group (20 minutes for a student alone).

<u>Documents to be delivered</u>:

- The written report should include: the conceptual model, the methodology used for estimating the statistics, the results, both graphical and numerical form, with the relative comments
- the listing code.

<u>Evaluation grid</u>

- clarity
- pertinence
- completeness

BOZZA di progetto

Si consideri una organizzazione che fornisce ai suoi dipendenti N postazioni di accesso ad internet attraverso 4 server di rete. Gli utenti potranno quindi collegarsi a un sito web sottoponendo richieste di pagine statiche. Il web server una volta elaborata la richiesta, invierà la risposta all'utente. E' interesse sia del gestore del sito web che dell'organizzazione stessa, riuscire a mantenere il tempo di risposta entro valori adeguati in modo da "soddisfare" gli utenti.
Allo scopo di studiare l'andamento delle prestazioni, si costruisce il seguente modello a rete con M centri:

- M=8 centri esponenziali, N=30 utenti; in particolare:
  - Centro 1, IS, tempo di servizio medio $E(t_{s1})=12$ s;
  - Centri 2, 3, 4, 5, FIFO, servente singolo, tempo di servizio medio $E(t_{si})=0.8$ s, i=2, 3, 4, 5;
  - Centro 6, FIFO, servente singolo, tempo di servizio medio $E(t_{s6})=0.3$ s;
  - Centro 7, FIFO, servente multiplo, m=3, tempo di servizio medio $E(t_{si})=1.5$ s, i=1, 2, 3;
  - Centro 8, FIFO, servente singolo, tempo di servizio medio $E(t_{s8})=0.3$ s;

- Parametri di routing:
  - $p_{12}= p_{13}= p_{14}= p_{15}=0.25$;
  - $p_{ii}=0.15$ per i=2, 3, 4, 5;
  - $p_{26}= p_{36}= p_{46}= p_{56}=0.85$;
  - $p_{67}= p_{81}=1$
  - $p_{77}=0.15$, $p_{78}=0.85$.

Il centro 1 modella le postazioni di accesso da cui gli utenti possono collegarsi al web. I centri 2, 3, 4 e 5 modellano quattro server di rete dell'organizzazione. I centri 6 e 8 modellano il ritardo dovuto alla rete per la "richiesta" e per la "risposta" rispettivamente. Infine il centro 7 modella un multiprocessor utilizzato come web server. Il suo tempo medio di servizio modella l'intero processo di reperimento dell'informazione richiesta, incluso l'accesso al data base.

Sviluppare un simulatore del sistema per determinare l'andamento del tempo medio di risposta rispetto al centro 1 per valori di popolazione crescente n= 1, 2, 3, …, N.
E' necessario migliorare il tempo medio di risposta del 20%. Il gestore del web server ha la possibilità di agire secondo le seguenti modifiche:

a) inserire una Task Assignment Policy per migliorare le prestazioni;
b) modificare lo scheduling astratto dei centri introducendo scheduling SB;
c) Incrementare del 25% la capacità di servizio del web server.

Studiare l'effetto delle modifiche separatamente, e l'effetto delle modifiche combinate rispetto all'obiettivo.

Il tool dovrà fornire gli indici locali ai centri e l'indice globale richiesto.