Performance Modeling of Computer Systems and Networks

Prof. Vittoria de Nitto Personè

Finite-Horizon and Infinite-Horizon Statistics

Università degli studi di Roma Tor Vergata
Department of Civil Engineering and Computer Science Engineering

1

Ci troviamo al punto in cui:
Progettiamo gli esperimenti, li eseguiamo, ed analizziamo l'output

Algorithm 1.2: using the resulting model

7. **Design simulations experiments**
   - What parameters should be varied?
   - perhaps many combinatoric possibilities
8. **Make production runs**
   - Record initial conditions, input parameters
   - Record statistical output
9. **Analyze the output**
   - Random components → statistical analysis (means, standard deviations, percentiles, histograms etc.)
10. Make decisions
    - The step9 results drive the decisions → actions
    - Simulation should be able to correctly predict the outcome of these actions (→ further refinements)
11. Document the results
    - summarize the gained insights in specific observations and conjectures useful for subsequent similar system models

2

La base teorica è composta da:

non quelle che generiamo noi

noi facciamo simulazioni, non abbiamo VARIABILI random, ma VARIATE random, di cui non conosciamo la media.

Central limit theorem

If $X_1, X_2, \ldots, X_n$ is an iid sequence of random variables (RVs) with

- common mean μ
- common standard deviation σ

and if $\overline{X}$ is the (sample) mean of these RVs

then $\overline{X}$ approaches a *Normal*(μ, $\sigma / \sqrt{n}$) as $n \to \infty$

Theorem 2

If $x_1, x_2, \ldots, x_n$ is an independent random sample from a "source" of data with unknown mean μ, if $\bar{x}$ and s are the mean and standard deviation of this sample, and if n is large, it is approximately true that

$$t = \frac{\bar{x} - \mu}{s/\sqrt{n-1}}$$

is a *Student*$(n-1)$ random variate

se facciamo 300 simulazioni, per ciascuna faccio tempo risposta medio per ogni simulazione, ne ho 300.
Se conosco la media teorica so a cosa tendono, ma spesso non la conosciamo!

Theorem 3

If $x_1, x_2, \ldots, x_n$ is an independent random sample from a "source" of data with unknown mean μ

- if $\bar{x}$ and s are the sample mean and sample standard deviation
- n is large

Then, given a confidence parameter α with $0.0 < \alpha < 1.0$, there exists an associated positive real number t^* such that

$$Pr\left(\bar{x} - \frac{t^* s}{\sqrt{n-1}} \leq \mu \leq \bar{x} + \frac{t^* s}{\sqrt{n-1}}\right) \cong 1 - \alpha$$

l'incognita cade nell'intorno di questi valori, costruiti a partire dalla variabile di Student

3

Discrete Simulation
Interval Estimation

Vogliamo calcolare intervallo di confidenza per la media teorica, non cerchiamo un "punto preciso", mi va bene dire "nel 95% delle volte cade in questo intervallo", con l'intervallo più piccolo possibile.

Algorithm

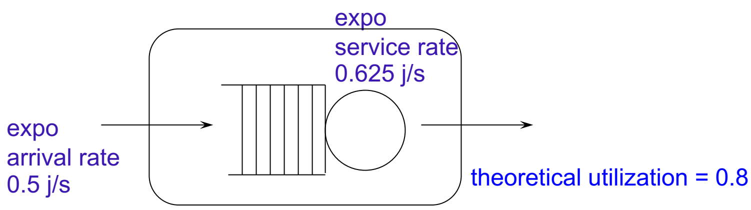
To calculate an interval estimate for the unknown mean μ of the population from which a random sample $x_1, x_2, \ldots, x_n$ was drawn:

- pick a level of confidence $1- \alpha$ (tipically α =0.05) l'intervallo è confidente al 95%
- calculate the sample mean $\bar{x}$ and standard deviation s (use Welford's algorithm)
- calculate the critical value t^* = *idfStudent*$(n - 1, 1 - \alpha/2)$
- calculate the interval endpoints $\bar{x} \pm \frac{t^* s}{\sqrt{n-1}}$

If n is sufficiently large, then you are $(1- \alpha)$x100% confident that the mean μ lies within the interval. The midpoint of the interval is $\bar{x}$

4

Abbiamo diverse criticità da analizzare: vediamo questa coda singola esponenziale (il caso a noi più familiare, risolvibile con KP).
Vogliamo vedere dimensione del campione adatto per includere la media teorica.
Il tempo di risposta medio sarà di 8 secondi (usando la KP).



STOP=200

seed (9 digits or less) >> 1234	seed (9 digits or less) >> 5678
for 103 jobs	for 89 jobs
average interarrival time = 1.93	average interarrival time = 2.22
average wait .... = 7.34	average wait .... = 4.19
average delay .... = 5.60	average delay .... = 2.61
average service time .... = 1.74	average service time .... = 1.58
average # in the node ... = 3.69	average # in the node ... = 1.88
average # in the queue .. = 2.81	average # in the queue .. = 1.17
utilization .... = 0.87	utilization .... = 0.71
jobs remaining in the node= 0	jobs remaining in the node= 0

tempi di risposta

abbiamo usato ssq3 con stop di 200.

Il sistema partiva da vuoto e terminava vuoto.

OR continua finchè sistema non è vuoto

```
....
while ((t.arrival < STOP) || (number > 0)) {
   t.next= Min(t.arrival, t.completion); /* next event time   */
   ....


printf(" … jobs", index);
printf(" average interarrival time ..", t.last / index);
printf(" average wait …", area.node / index);
printf(" average delay ...", area.queue / index);
printf(" average service time ...", area.service / index);
printf(" average # in the node ... ", area.node / t.current);
printf(" average # in the queue .. ", area.queue / t.current);
printf(" utilization ....", area.service / t.current);
printf(" jobs remaining in the node= ....", number);
```

rimuovendola non mi interesso più se ci siano ancora job, l'unica condizione è il tempo di STOP.

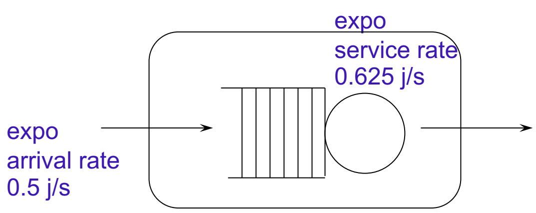
svuoto alla fine

non svuoto alla fine

STOP=200

seed (9 digits or less) >> 1234	seed (9 digits or less) >> 5678
for 103 jobs	for 89 jobs
average interarrival time = 1.93	average interarrival time = 2.22
average wait .... = 7.34	average wait .... = 4.19
average delay .... = 5.60	average delay .... = 2.61
average service time .... = 1.74	average service time .... = 1.58
average # in the node ... = 3.69	average # in the node ... = 1.88
average # in the queue .. = 2.81	average # in the queue .. = 1.17
utilization .... = 0.87	utilization .... = 0.71
jobs remaining in the node= 0	jobs remaining in the node= 0
for 99 jobs	for 88 jobs
average interarrival time = 2.01	average interarrival time = 2.25
average wait .... = 7.54	average wait .... = 4.23
average delay .... = 5.79	average delay .... = 2.64
average service time .... = 1.75	average service time .... = 1.59
average # in the node ... = 3.76	average # in the node ... = 1.88
average # in the queue .. = 2.89	average # in the queue .. = 1.17
utilization .... = 0.87	utilization .... = 0.71
jobs remaining in the node= 4	jobs remaining in the node= 1

7



seeds nuovo seme: 4321

	1234	5678	4321
average wait	7.34	4.19	2.65
	7.54	4.23	2.68

STOP=200
about 100 jobs

average wait
theoretical value
8 s

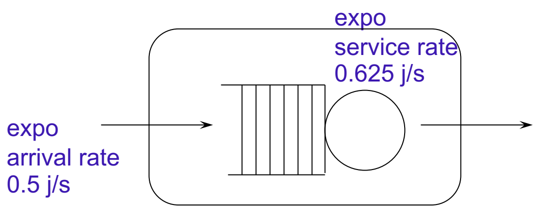
tempi di risposta per seme 4321:
2.65 se svuoto
2.68 se non svuoto
molto diverso da 8s che è il valore teorico

magari 100 job sono pochi per la stazionarietà

If the program is executed multiple times varying only the rngs initial seed from replication to replication, simulazioni che differisco SOLO per il seme sono dette REPLICHE

- the average wait in the node will vary significantly
- for most replications, the average wait will not be close to the steady-state average wait
- the initial conditions affect the results

8 Cioè stiamo dicendo che con repliche diverse, in questa situazione, comportano risultati abbastanza diversi.



seeds

average wait theoretical value 8 s	1234	5678	4321	
	7.34	4.19	2.65	STOP=200 about 100 jobs
	7.54	4.23	2.68	
	8.39	7.30	8.59	STOP=20000 about 10000 jobs
	8.34	7.30	8.59	

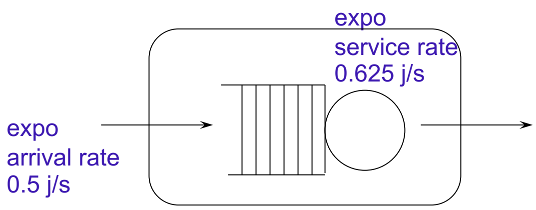
con 100 job non riuscivamo a caratterizzare bene il campione, con 10000 va meglio.

- If a relatively large number of jobs (say 10000) are used, variability of the average wait will be much less significant and be close to the steady-state average number
- When the number of jobs becomes infinite, the initial condition bias disappears

i risultati si somigliano di più, cioè sono meno variabili, mi avvicino allo stazionario. All'infinito, dovremmo perdere l'influenza delle condizioni iniziali.

9

Adesso facciamo decrescere il tasso di servizio in modo che l'utilizzazione sia esattamente 1.
La coda cresce sempre di più, i tempi di risposta crescono. Tutto perchè utilizzazione è 1.
Il sistema non converge, ma non si stabilizza!



If we further decrease the service rate → 0.5 j/s, utilization → 1

$\overline{w}, \overline{l}$ will tend to increase, without limit

$\Downarrow$

The server cannot keep up with demand, so the queue will grow as more jobs arrive
In this case, the steady-state average wait and the average number in the node are both infinite

10

Distinguiamo le statistiche dello stato STAZIONARIO, se esistono (non è detto!)

Def. *Steady-state statistics*

Steady-state system statistics are those statistics, if they exist, that are produced by simulating the operation of a *stationary* discrete-event system for an effectively infinite length of time

Simulano l'operatività del sistema stazionario. Si ottengono quando esistono simulando il sistema per un tempo potenzialmente infinito. Ovviamente noi fermeremo la simulazione, ma ad un tempo abbastanza lungo considerabile come infinito.

A *finite-horizon* discrete-event simulation is one for which the simulated operational time is finite volontariamente la finestra è limitata

An *infinite-horizon* discrete-event simulation is one for which the simulated operational time is effectively infinite

- Transient system statistics are those statistics that are produced by a finite-horizon discrete-event simulation oppure all'inizio di un orizzonte infinito)
- Steady-state statistics are produced by an infinite-horizon simulation
- The initial conditions affect finite-horizon statistics (l'infinito no, tempo così lungo che l'effetto si perde)
- The initial conditions do not affect infinite-horizon statistics: after enough time, the system loses memory of its initial state

Il sistema perde memoria, perchè "passa tanto tempo".

Another Important Distinction

- In an infinite-horizon simulation, the system “environment” is assumed to remain static (non ci sono cambiamenti, cose come il tasso di arrivo non cambiano, sennò sarebbe transitorio)
  If the system is a single-server service node, both the arrival rate and the service rate are assumed to remain constant in time
- In a finite-horizon simulation, no need to assume a static environment infatti il mio obiettivo è proprio vedere questi cambiamenti, transienti!

Se ragioniamo in fasce orarie, nella singola fascia oraria, se il sistema non varia in nulla, posso assumere orizzonte infinito. Bisogna analizzarli bene.

Relative Importance of Two Statistics

Per alcuni sono più rilevanti le statistiche stazionarie, per altri le transienti.

- The “traditional” view: steady-state statistics are most important
  - Steady-state statistics are better understood because they are much more easy to analyze mathematically
  - It is frequently difficult to accurately model initial conditions and non-stationary system parameters
- The “pragmatic” view: transient statistics are most important because steady-state is just a convenient fiction
- Depending on the application, both transient and steady-state statistics may be important

L'importante è usarle in maniera appropriate!

(nb: lo stazionario è più facile, anche perchè le condizioni iniziali non hanno influenza)

Relative Importance of Two Statistics

- Important to decide which statistics best characterize the system's performance

come stavamo dicendo prima, dobbiamo capire caso per caso se è opportuno caratterizzare lo studio in modo transiente o stazionario.

one of the most important skills:

the ability to decide, on a system-by-system basis, which kind of statistics best characterizes the system's performance.

Facciamo un esempio: uno pensa che se il tempo da simulare è piccolo allora uso transiente, se lungo stazionario.

NON E' COSÌ, VEDIAMO UN ESEMPIO

Steady-state or Transient Statistics

caso1:

Consider a bank that opens at 9 AM and closes at 5 PM.
A finite-horizon simulation over the 8-hour period produces transient statistics valuable in determining the optimal staffing of tellers throughout the day.

caso2:

Consider a fast food restaurant with a drive-up window that experiences a lunch rush period between 11:45 AM and 1:15 PM with an arrival rate that remains constant over the rush period.
This 90-minute period could be simulated for a much longer time period, producing steady-state statistics which might be valuable for estimating the average wait time at the drive-up window.

tempo piccolo, ma operatività costante, userò l'orizzonte infinito, perchè il tasso di arrivo è costante!

Se una persona arriva alle 13:10 sicuramente terminerà di mangiare dopo le 13:15, anche per questo usiamo orizzonte infinito, per farlo terminare nell'osservazione.

Orizzonte finito o infinito non vengono scelti in base alla durata dell'osservazione, ma solo se le condizioni durante l'osservazione cambiano o rimangono costanti.

Anche se nel mio studio io dovessi far riferimento ad un transiente, con l'analisi stazionaria posso validare, e dopo aver validato concentrarmi sul transiente.

Nel caso dell'inventori system, stato iniziale e finale del sistema era uguale (inventario al massimo)
e la condizione terminale era definita
dal nostro periodo di osservazione (es: osservo 12 settimane)

Initial and Terminal Conditions

- Finite-horizon discrete-event simulations are also known as *terminating simulations* (simulazioni a termine)
  - In program ssq4, the *system state* is idle at the beginning and at the end of the simulation a fine simulazione devo tornare nella condizione iniziale, cioè idle.
  - The *terminal condition* is specified by the "close the door" time
  - The *system state* of sis4 is the current and on-order inventory levels; these states are the same at the beginning and at the end of the simulation
  - The *terminal condition* is specified by the number of time intervals

Nell'orizzonte finito simuliamo un tempo che modella l'infinito, ma non è che la simulazione non finisca mai!
Posso fermarmi con sistema pieno, vuoto, posso fare come voglio, perchè sto simulando un tempo infinito, non mi cambia nulla, e rispetto a una simulazione così lunga, i pochi punti all'inizio (o alla fine) hanno impatto quasi nullo!

- Infinite-horizon discrete-event simulations (*non-terminating simulations*) must be terminated; typically done using whatever stopping conditions are most convenient
  - The steady-state statistics are based on such a huge amount of data that a few "non-steady-state" data points accumulated at the beginning and the end of the simulation should have no significant impact (bias) on output statistics

Sto stimando processo stocastico dipendente dal tempo.

Formal Representation

- The state variable $X(\cdot)$ is known formally as a *stochastic process*
  - The typical objective of a finite-horizon simulation of this system would be to estimate the time-averaged *transient* statistic

statistica transiente mediata nel tempo

random variable

$$\bar{X}(\tau) = \frac{1}{\tau}\int_0^\tau X(t)dt$$

qui "tau" è il tempo finale dell'osservazione, il tempo di risposta è in funzione di questo "tau"

where $\tau > 0$ is the terminal time

  - The typical objective of an infinite-horizon simulation of this system would be to estimate the time-averaged *steady-state* statistics

is not a random variable

$$\bar{x} = \lim_{\tau\to\infty} \bar{X}(\tau) = \lim_{\tau\to\infty} \frac{1}{\tau}\int_0^\tau X(t)dt$$

all'infinito porto questo X(tau) ad infinito.
X(tau) è realizzazione del processo stocastico (ma è comunque variabile random).

Il metodo delle repliche può essere usato anche nell'orizzonte finito.
Si varia il seme iniziale tra le varie "run", ma essi non devono sovrapporsi (altrimenti le realizzazioni sono correlate, mentre i teoremi che usiamo richiedono l'INDIPENDENZA). A noi basta definire un ciclo di 300 repliche, in cui in ogni run azzero le statistiche, riporto il sistema a vuoto, ma non "sposto lo stato del generatore", lo faccio avanzare, non è che lo sostituisco con un altro, sennò non so "dove va"!

Replication

- If a discrete-event simulation is repeated, varying only the rngs initial states from run to run, each run of the simulation program is a *replication* and the totality of replications is said to be *ensemble*
- Replications are used to generate *estimates* of the same transient statistic
- The initial seed for each replication should be chosen to be no replication-to-replication overlap
- The standard way is to use the final state of each rngs stream from *one* replication as the initial state for the *next* replication accomplished by calling PlantSeeds once outside the main replication loop

19

qui, quando parliamo di "stato", è lo "stato del generatore". Lo stato finale del generatore diventa lo stato iniziale per il run che parte, non corro il rischio di prendere gli stessi numeri random. Come? facendo un ciclo ed inizializzando il generatore fuori.
Se primo run inizia con 20 job (magari studio il transiente), allora questa condizione non deve essere azzerata, tutte le repliche avranno statistiche azzerate ogni volta, ma se la condizionale iniziale è con 20, deve essere per tutti. I generatori NON LI TOCCO.

Independent Replications and Interval Estimation

Suppose the finite-horizon simulation is replicated n times, each time generating a state time history $x_i(t)$

$$\bar{x}_i(\tau) = \frac{1}{\tau}\int_0^{\tau} x_i(t)dt$$

where $i = 1, 2, \ldots, n$ is the replication index

lo faccio "n" volte
per costruire il campione

Each data point $\bar{x}_i(\tau)$ is an independent observation of the random variable $\bar{X}(\tau)$ Le ipotetiche 300 repliche con i loro valori sono osservazioni INDIPENDENTI del processo stocastico X(tau)

If n is large enough, the pdf of $\bar{X}(\tau)$ can be estimated from a histogram of the $\bar{x}_i(\tau)$

Per "n" grande, posso stimare anche la densità mediante istogramma.

20

Posso anche fare un intervallo di stima. A noi interessa spesso E[X(tau)].
Anche se "n" non è grande, ma c'è indipendenza, posso calcolarla.

Independent Replications and Interval Estimation

In practise, it is usually only the expected value $E[\bar{X}(\tau)]$ that is desired.
A *point* estimate of this transient statistic is available as an *ensemble average*, even if n is not large

$$\frac{1}{n}\sum_{i=1}^{n}\bar{x}_i(\tau)$$

An interval estimate for $E[\bar{X}(\tau)]$ can be calculated

Use the interval estimation technique from Section 8.1

Example 8.3.6

Noi facciamo studio transiente, ma qui sembra che riusciamo a prendere anche lo stazionario!

A modified version of ssq2 was used to produce 20 replications

- 100 jobs processed through M/M/1 service node
  - Node is initially idle
  - Arrival rate is = 1.0
  - Service rate is = 1.25

1/(mu - lamda) = 4 = valore teorico

- The resulting 20 observations of the average wait in the node:

from program estimate 95%-confidence-interval: we are 95% confident that if we were to do millions of replications the ensamble average would be somewhere between 2.29 and 4.37 (il centro è 3.33, calcolata con le 20 repliche)

4 is the exact steady state value

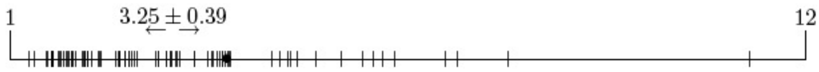
It is possible that 100 jobs are enough to produce steady-state statistics

Sembra che 20 repliche con 100 bastino per catturare il valore teorico. Possibile? Proviamo!

L'intervallo di confidenza scende con radice di "n", siamo passati da 20 repliche a 80 repliche, il nuovo "n" è 4*20 =80, e la nuova "radice" è il doppio di quella di prima, ma poichè la radice è al denominatore, l'intervallo di confidenza diventa la metà.

Example 8.3.7

- The modified version of program ssq2 was used to produce 60 more replications
- Consistent with $\sqrt{n}$ rule, expect two-fold decrease in the width of the interval estimate
- Based on 80 replications, the resulting 95% confidence interval estimate was 3.25 ± 0.39 (2.86, 3.64)



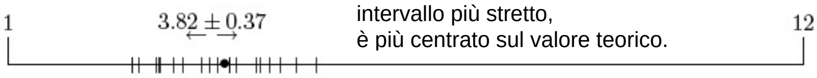
In this case 100 jobs <u>are not enough</u> to produce steady-state statistics

the bias of the initially idle state is still evident in the transient statistic

Example 8.3.8

Passiamo a 1000 job, 20 repliche.

- As a continuation of Example 8.3.6, the number of jobs per replication was increased from 100 to 1000
- 20 replications were used to produce 20 observations of the average wait in the node (3.45, 4.19)



intervallo più stretto,
è più centrato sul valore teorico.

Relative to Example 8.3.6, much more symmetric sample mean in Example 8.3.8 (2.29, 4.37)

Example 8.3.8

- The 1000-jobs per replication results are more consistent with the underlying theory of interval estimation
  - Requires a sample mean distribution that is approximately *Normal*(μ, $\sigma / \sqrt{n}$)
  - Sample mean distribution is centered on (unknown) population
- 1000 jobs may achieve steady-state; 100 jobs cannot