Analisi del flusso di utenti del sito web della Juventus FC

Mario Marcello 0333376

I. Introduzione

Il sistema in analisi è il sito web www.juventus.com della società sportiva Juventus FC. Tale sito permette agli utenti di accedere allo store per acquistare gadget e articoli sportivi del marchio Juventus e, soprattutto, di accedere ai botteghini "home", "women" ed "away" per poter acquistare i tagliandi per le partite della squadra maschile e di quella femminile, sia per i tifosi di casa che per i tifosi ospiti. L'abolizione della vendita fisica dei biglietti ha reso il sito web l'unico mezzo per accedere agli eventi sportivi della Juventus. In teoria, questo dovrebbe facilitare il processo d'acquisto, rendendolo più semplice e accessibile a tutti. Tuttavia, nella pratica, sebbene il sito semplifichi le operazioni, i tempi di attesa risultano spesso eccessivamente lunghi, soprattutto per chi non dispone di accessi prioritari. La campagna di vendita dei biglietti è articolata nel seguente modo: una decina di giorni prima dell'inizio del campionato vengono messi in vendita i tagliandi per la prima partita, mediamente in cinque giorni vengono esauriti per poi essere sostituiti dai biglietti della partita successiva, ciò si ripete fino alla fine della stagione sportiva. Poiché tra campionato, coppe europee e coppe nazionali, la Juventus è impegnata da fine agosto ai primi di maggio, il sito web della società registra un alto flusso di affluenza pressoché costante per quasi nove mesi, poiché, come detto in precedenza, in cinque giorni i biglietti di una partita vengono esauriti e, il giorno seguente, vengono rimpiazzati dai biglietti del match successivo. L'obbiettivo dell'analisi è quello di trovare una configurazione ottimale per il sito www.juventus.com in modo da rendere i tempi di attesa meno proibitivi. Per fare ciò è necessario andare a vedere come funziona il sito web nello specifico. Quando un utente tenta di accedere, esso viene fatto identificare nella fase di login tramite id e password, e, in questo momento, esso clicca sulla sezione del sito che vuole visitare. Qualora scegliesse di accedere ai botteghini women ed away andrebbe in una semplice coda FIFO; ma, se volesse accedere al botteghino home verrebbe inserito in una coda prioritaria. Questo avviene perché, durante l'estate, prima dell'inizio della stagione sportiva, la società Juventus FC mette a disposizione diversi tipi di abbonamenti. Il più classico è l'abbonamento a tutte le partite che, per ovvie ragioni, non verrà considerato in questa analisi, poichè non va ad interferire sul carico di lavoro del sistema durante la vendita dei tagliandi di ogni singola partita, ma va semplicemente a diminuire il numero di posti disponibili nello stadio. Gli abbonamenti di interesse sono altri tre: ovvero il J1987, lo Stadium Member e la Juventus Card. Questi abbonamenti permettono di accedere a molti servizi diversi che, però, non sono molto interessanti ai fini dell'analisi se non per uno: infatti loro garantisco accessi prioritari al botteghino "home". Il precedente elenco è ordinato in ordine di priorità decrescente ma la Juventus Card non è il livello di priorità più basso in assoluto. Questo perché nessun tifoso è obbligato a sottoscrivere un abbonamento per assistere alle partite e quindi, chi sceglie di non abbonarsi, si affida alla vendita libera ottenendo così il livello di priorità più basso. Infine, dopo che ogni utente ha visitato la sezione del sito di suo interesse è libero di decidere se acquistare qualcosa o meno, qualora decidesse di non comprare nulla abbandona il sistema, altrimenti viene indirizzato in una coda FIFO per procedere al pagamento, dopo il quale esce dal sistema.

II. Obiettivi

L'obiettivo finale è quello di modellare il sistema cercando di ottenere la distribuzione ottimale di serventi per gestire il carico totale di lavoro in modo efficiente. Per fare ciò è necessario andare ad impostare dei requisiti di QoS che devono essere rispettati in modo da rendere i tempi di attesa abbastanza brevi. Nello specifico sono i seguenti:

- Il tempo di attesa per accedere al servizio di login non dovrebbe essere maggiore di un millesimo di secondo
- Il tempo di attesa per accedere al botteghino women non dovrebbe essere maggiore di 2 minuti
- Il tempo di attesa per accedere al botteghino away non dovrebbe essere maggiore di 2 minuti
- Il tempo di attesa per accedere al botteghino home per un abbonato J1987 non dovrebbe essere maggiore di 2 minuti
- Il tempo di attesa per accedere al botteghino home per un abbonato Stadium Member non dovrebbe essere maggiore di 10 minuti
- Il tempo di attesa per accedere al botteghino home per un abbonato Juventus Card non dovrebbe essere maggiore di 20 minuti
- Il tempo di attesa per accedere al botteghino home per un tifoso che si affida alla vendita libera non dovrebbe essere maggiore di 30 minuti
- Il tempo di attesa per accedere ai servizi di pagamento non dovrebbe essere maggiore di 1 secondo

Tutto ciò è necessario per garantire un buon servizio in linea con i tempi di attesa classici evitando così eventuali abbandoni da parte di utenti stanchi di attendere troppo. Inoltre, è importante evitare di sovradimensionare il sistema, poiché, in questo scenario, si avrebbe un grande abbattimento dei tempi di attesa ma anche un aumento dei costi ed un inutile spreco di risorse.

III. MODELLO CONCETTUALE

Il sistema è stato modellato come una rete di code e in tale modello sono presenti i seguenti sette nodi:

- Nodo che si occupa del servizio di login (Login)
- Nodo che si occupa del servizio di vendita dei biglietti della squadra femminile (Women)
- Nodo che si occupa del servizio di vendita dei biglietti per i tifosi di casa (Home)
- Nodo che si occupa del servizio di vendita dei biglietti per i tifosi ospiti (Away)
- Nodo che si occupa del servizio di pagamento per i tifosi della squadra femminile (WomenPay)
- Nodo che si occupa del servizio di pagamento per i tifosi della squadra maschile (HomePay)
- Nodo che si occupa del servizio di pagamento per i tifosi ospiti (AwayPay)

Ogni coda è una coda FIFO con capacità infinita e con tempo di arrivo esponenziale. Per quanto riguarda la modellazione dei tempi di servizio, invece, sono state fatte scelte differenti. Il tempo del servizio login è stato modellato con una distribuzione esponenziale: tale scelta è dovuta dal fatto che tale servizio è solo computerizzato. Questo perché, quando l'utente decide di accedere al sito web, esso comunica con il browser e visualizza l'interfaccia del sito della Juventus, inserisce le sue credenziali e clicca invio; è in questo momento che l'utente entra in coda in attesa che il servizio di login si attivi e gli permetta l'accesso. Da ciò si deduce che tale servizio è solo di calcolo e di conseguenza può avere dei tempi molto vicini allo zero o molto grandi, indi per cui non necessita di alcun bound e può essere ben descritto da una distribuzione esponenziale. Per quanto riguarda gli altri servizi, invece, sono stati modellati tramite una distribuzione gaussiana troncata. Questo perché, nonostante tali servizi abbiano un tempo medio e possano avere eventuali variazioni, necessitano di bound poichè non possono né essere troppo grandi né essere troppo piccoli. Nello specifico è possibile accorpare, poiché simili, i sei nodi rimanenti in due gruppi: un gruppo è formato da Home, Women e Away; mentre l'altro è formato da HomePay, WomenPay e AwayPay. Quando un job entra in un nodo del primo gruppo l'utente ha a disposizione il servizio come se si trovasse in una sezione critica, questo per evitare che più utenti acquistino lo stesso biglietto. I servizi relativi a questi nodi permettono all'utente di visualizzare i vari posti disponibili, vederne il prezzo, selezionare il più adatto e inserirlo nel carrello; ovviamente, però, trovandosi in una sezione critica, l'utente ha a disposizione un tempo proprio ma limitato, superato il quale egli viene disconnesso. Nello specifico è stato impostato un upper bound di 15 minuti. Per quanto riguarda il lower bound, invece, è stato impostato un tempo minimo di 7 minuti. Per quanto riguarda il secondo gruppo, invece, i servizi che ne fanno parte permettono all'utente di pagare ciò che ha precedentemente inserito nel carrello; essendoci di mezzo una transazione è necessario garantire l'atomicità e quindi il servizio in questione garantisce un tempo proprio all'utente ma limitato, scaduto il quale l'acquisto viene annullato e l'utente disconnesso. Nello specifico tale limite è stato impostato a 5 minuti. Per quanto riguarda il lower bound è stato scelto un tempo minimo di 2 minuti. Terminata la disamina sulla generazione dei tempi di servizio è necessario sottolineare che tutti i server sono a coda singola tranne il server del nodo Home che ha una multi-coda a priorità astratta. Inoltre, va detto che ogni job è di tipo non preemptive e che ogni server è conservativo.

Tutti i job che arrivano passano nel nodo Login dove vengono divisi ed indirizzati nei corretti percorsi. Per effettuare tale divisione il servente del primo centro assegna un codice ad ogni job; i codici possibili sono i seguenti:

- codeWomen: assegnato ad un job che deve essere indirizzato al nodo Women
- codeAway: assegnato ad un job che deve essere indirizzato al nodo Away
- codeJ1987: assegnato ad un job che deve essere indirizzato al nodo Home con priorità massima
- codeStadium: assegnato ad un job che deve essere indirizzato al nodo Home con priorità media
- codeCard: assegnato ad un job che deve essere indirizzato al nodo Home con priorità bassa
- codeFree: assegnato ad un job che deve essere indirizzato al nodo Home con priorità minima

A questo punto, dopo che il job è stato eseguito, esso viene indirizzato verso il rispettivo "Nodo-Pay"; ciò vuol dire che se il job è stato processato dal nodo Women passerà in WomenPay, se era in Home andrà in HomePay ed analogamente da Away entrerà in AwayPay. Però qui è necessaria una considerazione. Quando un utente visita un qualsiasi sito di e-commerce egli non è obbligato ad acquistare qualcosa e ciò vale anche per gli utenti del sito web in analisi. Un utente che accede al botteghino "home" potrebbe semplicemente dare un'occhiata ma non comperare nulla, ciò vuol dire che usufruirebbe del servizio offerto dal nodo Home ma non necessiterà del servizio offerto dal nodo HomePay; ovviamente discorso analogo per gli altri due casi. Quindi è necessario tener conto che alcuni job potrebbero abbandonare il sistema dopo aver visitato uno dei nodi subito successivi a quello di login senza mai arrivare ai nodi terminali. Tale eventualità è stata modellata come una probabilità di abbandono.

IV. MODELLO DELLE SPECIFICHE

I parametri di arrivo necessari al simulatore sono gli arrivi medi, le probabilità di assegnazione per ogni coda, il tempo di servizio medio di ogni nodo e la probabilità di abbandono nelle code di WomenPay, HomePay e AwayPay. Purtroppo, trovare dati puntuali per tali parametri non è stato possibile, di conseguenza tutti i dati di input utilizzati durante la simulazione sono frutto di ragionamenti effettuati su dati attendibili ottenuti da siti come quello della Juventus o quello di Sky Sport e da esperienze personali; e sono mirati a rendere il tutto il più verosimile possibile.

Per calcolare il tasso di arrivo medio λ è necessario calcolare prima il tasso di arrivo totale nei cinque giorni di vendita e poi dividerlo per i 7200 minuti che compongono queste giornate. Per trovare questo valore è necessario capire quanti sono gli utenti interessati alla partita della squadra maschile e quanti sono quelli interessati al match della squadra femminile

Ricavare il numero di utenti interessati all'acquisto di biglietti per la squadra maschile è relativamente semplice. Lo Juventus Stadium ha una capienza di 42507 spettatori ed in media ha una copertura annua del 98%. Ciò vuol dire che mediamente i tifosi presenti ad ogni partita sono circa 41676

di cui 17083 (il 41%) sono abbonati a tutte le partite della stagione. I restanti 23598 sono i tifosi che accedono al sito per acquistare i biglietti per la partita, di questi 3099 sono ospiti, i restanti sono bianconeri. Per quanto riguarda il numero di utenti interessati alla partita della squadra femminile si può effettuare un ragionamento analogo al precedente. La Juventus Women disputa le partite casalinghe nello stadio La Marmora-Pozzo di Biella che ha una capienza di 5827 spettatori ed in media ha una copertura annua del 23%. Ciò vuol dire che mediamente i tifosi presenti ad ogni partita sono circa 1340. Dato che per le partite della squadra femminile non sono previsti abbonamenti, questo dato indica il numero di utenti che accedono al sito per acquistare i tagliandi della partita.
Quindi, in questo momento, il tasso di arrivo totale è pari a 26054 tifosi paganti suddivisi in 21499 utenti home, 3099 utenti away e 1340 utenti women. A questi valori, però, vanno aggiunti anche tutti quegli utenti che accedono ai vari botteghini, ma poi, a causa dei prezzi troppo elevati, rinunciano ad acquistare i biglietti. Indi per cui, per ottenere la stima corretta, bisogna tener conto anche delle probabilità di abbandono:

- Per un tifoso home la probabilità di abbandono è pari al 10%
- Per un tifoso away la probabilità di abbandono è pari al 30%
- Per un tifoso women la probabilità di abbandono è pari al 5%

Tali valori vanno ad aumentare i dati ottenuti precedentemente portando gli utenti home ad essere 23649, gli utenti away a 4028 e gli utenti women a 1407; per un totale di 29084. Questi sono gli utenti che accedono al sito nell'arco dei cinque giorni necessari all'esaurimento dei biglietti; quindi, il tasso di arrivo medio λ è di 4,039 job/min.
Da questi dati è possibile ottenere le varie probabilità con le quali un job possa essere assegnato ad una coda dopo la fase di login:

- Probabilità di assegnazione alla coda del nodo Women pari al 4,84%
- Probabilità di assegnazione alla coda del nodo Home pari al 81,31%
- Probabilità di assegnazione alla coda del nodo Away pari al 13,85%

Un altro aspetto da considerare è l'assegnazione della priorità per i job di tipo home. Questo dato non è facilmente reperibile ma si può definire una stima andando ad analizzare i prezzi dei singoli abbonamenti. I costi degli abbonamenti ammontano annualmente a 190€ per il J1987, 45€ per lo Stadium Member e 15€ per la Card; ovviamente la vendita libera non necessita di alcun abbonamento. Da questi dati è possibile definire le seguenti probabilità:

- Probabilità di assegnare ad un job il codice di priorità massima codeJ1987 pari al 5%
- Probabilità di assegnare ad un job il codice di priorità media codeStadium pari al 15%
- Probabilità di assegnare ad un job il codice di priorità bassa codeCard pari al 30%
- Probabilità di assegnare ad un job il codice di priorità minima codeFree pari al 50%

A questo punto gli ultimi valori da dover determinare sono i tempi di servizio medi dei singoli serventi che possono essere stimati come segue:

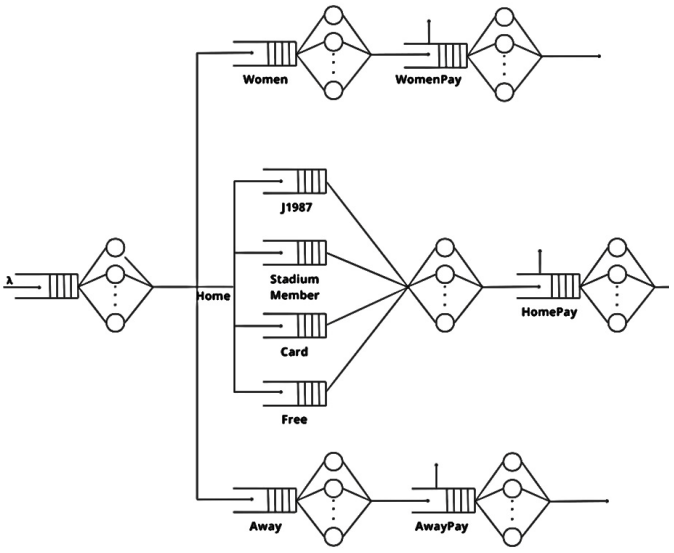
- Tempo di servizio del servente del nodo Login pari a 3 secondi
- Tempo di servizio del servente del nodo Women pari a 12 minuti
- Tempo di servizio del servente del nodo Home pari a 12 minuti
- Tempo di servizio del servente del nodo Away pari a 12 minuti
- Tempo di servizio del servente del nodo WomenPay pari a 4 minuti
- Tempo di servizio del servente del nodo HomePay pari a 4 minuti
- Tempo di servizio del servente del nodo AwayPay pari a 4 minuti

La simulazione, invece, è stata pensata come una Next Event Simulation per cui necessita di diversi eventi per poter garantire l'avanzamento del clock. Gli eventi stabiliti sono i seguenti:

- Arrivo
- Completamento del servizio di login
- Completamento del servizio women
- Completamento del servizio home
- Completamento del servizio away
- Completamento del servizio pagamento women
- Completamento del servizio pagamento home
- Completamento del servizio pagamento away

L'evento di arrivo è necessariamente il primo che occorre e, quindi, colui che da inizio alla simulazione. A livello algoritmico esso inserisce un nuovo job in login e genera un nuovo arrivo. Qualora fosse presente almeno un server libero cerca il primo server disponibile; imposta lo stato del server ad occupato e genera un tempo di completamento per il job.
L'evento di completamento della fase di login è responsabile dell'indirizzamento dei vari job nei corretti nodi della rete. Precisamente si occupa di svuotare il server e diminuire il numero di job nel nodo Login, fatto ciò, controlla se sono presenti job in coda e in caso di risposta affermativa inserisce un job nel server appena liberato e genera un nuovo tempo di completamento. Dopo di che assegna un codice al job appena terminato e lo indirizza al relativo nodo; i codici possibili sono sei e sono i già citati codeWomen, codeAway, codeJ1987, codeStadium, codeCard e codeFree; il primo viene indirizzato in Women, il secondo in Away ed i restanti in Home. Dopo tale distribuzione la procedura è la stessa in ogni nodo; infatti, verrà controllata la presenza di un server libero ed in caso di risposta positiva verrà inserito il job in servizio e verrà generato un tempo di completamento. Gli eventi di completamento in Women, Away ed Home hanno un funzionamento analogo; infatti, tutti e tre, svuotano il proprio server ed eleggono il prossimo job da mettere in servizio; i primi due lo scelgono con una semplice politica FIFO mentre l'ultimo effettua diversi controlli poiché deve gestire correttamente le varie priorità. Fatto ciò, generano un tempo di completamento ed infine estraggono una probabilità per il job appena terminato, in caso venga superata un determinata soglia differente da centro a centro, il job appena terminato viene eliminato dal sistema; altrimenti viene indirizzato alla coda del relativo nodo di pagamento. Tale

probabilità è la già citata probabilità di abbandono necessaria per modellare il caso in cui un utente usufruisca del servizio senza però acquistare nulla.
Gli ultimi tre eventi da analizzare sono quelli di completamento dei pagamenti, questi eventi sono identici e si occupano di svuotare il relativo server che ha terminato, controllare la presenza di un job in coda pronto all'esecuzione e, se presente, inserirlo nel server appena liberato.
Con la gestione di questi eventi è possibile simulare l'intero sistema. Il clock funzionerà nel seguente modo: verrà inizializzato ad un valore di START e saranno inizializzati gli eventi, usando infinito per quelli impossibili e tenendo conto che all'inizio l'unico evento possibile è quello di arrivo dato che il sistema è vuoto. Ogni volta verrà calcolato il next event come l'evento più vicino nel tempo e si farà avanzare il clock fino ad esso. Processato questo evento verrà calcolato il prossimo next event e così via, fino al raggiungimento di una determinata condizione di stop.



V. MODELLO COMPUTAZIONALE

Per quanto riguarda il modello computazionale la scelta del linguaggio di programmazione per l'implementazione del simulatore è ricaduta su C. Al termine delle simulazioni verranno calcolate le varie statistiche di interesse e verranno salvate in un file csv, che verrà elaborato tramite Python e la libreria mathplotlib.py in modo da ricavare grafici per visualizzare i risultati ottenuti.
Poiché il fine ultimo del modello è quello di andare ad estrarre alcune statistiche che descrivono il sistema in analisi è stato necessario implementare un meccanismo di calcolo e di memorizzazione di tali statistiche. Per fare ciò è stata creata la seguente struct che memorizza come variano le statistiche ad ogni avanzamento del clock:

```
struct nodeData{
   double node;
   double queue;
   double service;
   int index;
   double current;
   int serverNumber;
}
```

I primi tre campi della struct contengono, rispettivamente, il numero di job presenti nel nodo, nella coda o in servizio, integrandoli sul tempo. Per poter ottenere questi valori, ad ogni avanzamento del clock si va a sommare al valore precedente di node, queue o service il numero degli utenti presenti nel centro, nella coda o in servizio moltiplicati per la differenza tra il valore del tempo corrente e il precedente valore del tempo. Il campo index tiene conto del numero di job entrati nel centro mentre current contiene il valore del tempo corrente; questi due campi sono importanti per poter ottenere statistiche mediate sul numero di job o sul tempo. L'ultimo campo, serverNumber, memorizza il numero di server del nodo.
Arrivati al termine della simulazione, tramite l'utilizzo dei valori contenuti nei campi citati in precedenza, vengono calcolate le statistiche finali che vengono memorizzate nella seguente struct:

```
struct output{
   double wait;
   double delay;
   double service;
   double numberNode;
   double numberQueue;
   double utilization;
   double job;
}
```

Il campo wait indica il tempo medio passato da un job nel nodo ed è calcolato come nodeData.node / nodeData.index.
Il campo delay indica il tempo medio passato da un job nella coda ed è calcolato come nodeData.queue/ nodeData.index.
Il campo service indica il tempo medio di servizio ed è calcolato come nodeData.service/ nodeData.index.
Il campo numberNode indica il numero medio di job nel sistema ed è calcolato come nodeData.node/ nodeData.current.
Il campo numberQueue indica il numero medio di job in coda ed è calcolato come nodeData.queue/ nodeData.current.
Il campo utilization indica l'utilizzazione del sistema ed è calcolata come nodeData.service/ nodeData.current/ nodeData.serverNumber ed è riferita all'intero nodo, non al singolo servente.
L'ultimo campo, job, salva solo il numero totale di job entrati nel nodo ed è pari a nodeData.index.
Queste statistiche vengono calcolate per ogni singolo nodo, e poi, nel nodo Home, poiché sistema a multi-coda di priorità, le stesse statistiche vengono calcolate per ogni singola coda. Questa scelta è stata fatta poiché è più utile estrapolare le statistiche di ognuno dei quattro livelli di priorità piuttosto che quelle medie dell'intero sistema.
Per quanto riguarda il simulatore vero e proprio, la generazione dei numeri casuali è stata fatta tramite la libreria rngs.h; nello specifico sono state utilizzate le funzioni PlantSeed(SEED) per l'inizializzazione degli stream; Random() per la generazione di un numero casuale e SelectStream(stream) per cambiare stream per la generazione di variabili differenti. Nello specifico sono stati utilizzati dodici stream per le seguenti variabili:

- Stream 0 per la generazione dei tempi di completamento del servizio di login
- Stream 1 per la generazione dei tempi di completamento del servizio women

- Stream 2 per la generazione dei tempi di completamento del servizio home
- Stream 3 per la generazione dei tempi di completamento del servizio away
- Stream 4 per la generazione dei tempi di completamento del servizio di pagamento dei biglietti women
- Stream 5 per la generazione dei tempi di completamento del servizio di pagamento dei biglietti home
- Stream 6 per la generazione dei tempi di completamento del servizio di pagamento dei biglietti away
- Stream 7 per la generazione del prossimo arrivo
- Stream 8 per assegnare un codice ad un job dopo il termine del servizio di login
- Stream 9 per stabilire se ci sarà un abbandono nella coda di WomenPay
- Stream10 per stabilire se ci sarà un abbandono nella coda di HomePay
- Stream 11 per stabilire se ci sarà un abbandono nella coda di AwayPay

Le variabili di arrivo e i tempi di servizio del nodo login vengono generate come delle esponenziali, quindi come – m * log(1-Random) dove m è la media; mentre i tempi di servizio degli altri nodi sono generati tramite una distribuzione gaussiana troncata. Per implementarla è stato necessario scrivere la seguente funzione:

```
double idfTruncatedNormal(double media, double devS,
   double lowBound, double upBound, double random){

      double a = cdfNormal(media, devS, lowBound-1)
      double b = 1.0-cdfNormal(media, devS, upBound)
      double u = idfUniform(a, 1.0-b, random)
      return idfNormal(media, devS, u)
}
```

Le altre variabili, invece, sono generate come delle uniformi, quindi con a+(b-a) * Random(); dove a e b sono gli estremi entro i quali viene generata la variabile e sono stati posti a 0 e a 100 in modo da poter gestire facilmente le percentuali.

VI. VERIFICA

Per la fase di verifica si è fatto uso di tempi di servizio esponenziali, in quanto, altrimenti, non si avrebbe un confronto valutabile dal punto di vista analitico utilizzando distribuzioni gaussiane troncate. Per garantire significatività alla verifica anche le simulazioni sono state realizzate con distribuzione dei servizi esponenziale. Dato che i calcoli teorici fanno riferimento al ad un sistema stazionario è stata utilizzata una simulazione ad orizzonte infinito.

Login

λ_{login} = 4,039 job/min E[S_i]= 0,05 min

N=3 $\mu = \frac{1}{E[Si]} = \frac{1}{0,05} = 20$ job/min

$E[S] = \frac{1}{m\mu} = \frac{1}{20 * 3} = 0,016667$ min

$\rho = \frac{\lambda}{m\mu} = \frac{4,039}{20*3} = 0,067317$

Essendo un sistema M/M/m a coda singola:

$E[T_q] = \frac{Pq\ E[S]}{1-\rho}$ con

$Pq = \frac{(mp)^m}{m!(1-\rho)} p(0)$ e $p(0) = [\sum_{i=0}^{m-1} \frac{(m\rho)^i}{i!} + \frac{(m\rho)^m}{m!(1-\rho)}]^{-1}$

$p(0) = [\sum_{i=0}^{2} \frac{(3 * 0,067317)^i}{i!} + \frac{(3 * 0,067317)^3}{3!(1-0,067317)}]^{-1} = 0,815878$

$Pq = \frac{(3 * 0,067317)^3}{3!(1-0,067317)} * 0,817569 = 0,001201$

$E[T_q] = \frac{0,001201 * 0,067317}{1 - 0,067317} = 0,000025$ min

$E[T_s] = E[T_q] + E[S_i] = 0,000025 + 0,05 = 0,050025$ min

$E[N_q] = \lambda E[T_q] = 4,039 * 0,000025 = 0,000101$

$E[N_s] = \lambda E[T_s] = 4,039 * 0,050025 = 0,202051$

I risultati del simulatore sono:

- $\rho = 0,068019 \pm 0,000683$
- $E[T_q] = 0,000019 \pm 0,000006$
- $E[T_s] = 0,050235 \pm 0,000387$
- $E[N_q] = 0,000079 \pm 0,000026$
- $E[N_s] = 0,204134 \pm 0,002052$

Poiché molto simili tra loro possono essere considerati verificati.

Nodo Women

p_{women} = 4,84% = 0,0484 E[S_i]= 12 min N=10

$\lambda_{women} = \lambda_{login} * p_{women} = 4,039 * 0,0484 = 0,195488$ job/min

$\mu = \frac{1}{E[Si]} = \frac{1}{12} = 0,083333$ job/min

$E[S] = \frac{1}{m\mu} = \frac{1}{10 * 0,083333} = 1,200005$ min

$\rho = \frac{\lambda}{m\mu} = \frac{0,195488}{10*0,083333} = 0,234587$

Essendo un sistema M/M/m a coda singola:

$p(0) = [\sum_{i=0}^{9} \frac{(10 * 0,234587)^i}{i!} + \frac{(10 * 0,234587)^{10}}{10!(1-0,234587)}]^{-1} = 0,095763$

$Pq = \frac{(mp)^m}{m!(1-\rho)} p(0) = \frac{(10 * 0,234587)^{10}}{10!(1-0,234587)} * 0,095763 = 0,000174$

$E[T_q] = \frac{Pq\ E[S]}{1-\rho} = \frac{0,000174 * 1,200005}{1 - 0,234587} = 0,000273$ min

$E[T_s] = E[T_q] + E[S_i] = 0,000273 + 12 = 12,000273$ min

$E[N_q] = \lambda E[T_q] = 0,234587 * 0,000273 = 0,000064$

$E[N_s] = \lambda E[T_s] = 0,234587 * 12,000273 = 2,815108$

I risultati del simulatore sono:

- $\rho = 0,234626 \pm 0,002444$
- $E[T_q] = 0,000308 \pm 0,000249$
- $E[T_s] = 11,995645 \pm 0,095498$
- $E[N_q] = 0,000061 \pm 0,000050$
- $E[N_s] = 2,346325 \pm 0,024443$

Poiché molto simili tra loro possono essere considerati verificati.

Nodo Away

p_{away} = 13,85% = 0,1385 E[S_i]= 12 min N=10

$\lambda_{away} = \lambda_{login} * p_{away} = 4,039 * 0,1385 = 0,559402$ job/min

$\mu = \frac{1}{E[Si]} = \frac{1}{12} = 0,083333$ job/min

$E[S] = \frac{1}{m\mu} = \frac{1}{10 * 0,083333} = 1,200005$ min

$\rho = \frac{\lambda}{m\mu} = \frac{0,559402}{10*0,083333} = 0,671285$

Essendo un sistema M/M/m a coda singola:

$p(0) = [\sum_{i=0}^{9} \frac{(10 * 0,671285)^i}{i!} + \frac{(10 * 0,671285)^{10}}{10!(1-0,671285)}]^{-1} = 0,001159$

$Pq = \frac{(mp)^m}{m!(1-\rho)} p(0) = \frac{(10 * 0,671285)^{10}}{10!(1-0,671285)} * 0,001159 = 0,180539$

$E[T_q] = \frac{Pq\ E[S]}{1-\rho} = \frac{0,180539 * 1,200005}{1 - 0,671285} = 0,659072$ min

$E[T_s] = E[T_q] + E[S_i] = 0,659072 + 12 = 12,659072$ min

$E[N_q] = \lambda E[T_q] = 0,559402 * 0,659072 = 0,368686$

$E[N_s] = \lambda E[T_s] = 0,559402 * 12,659072 = 7,081510$

I risultati del simulatore sono:

- $\rho = 0{,}684463 \pm 0{,}008199$
- $E[T_q] = 0{,}658324 \pm 0{,}063598$
- $E[T_s] = 12{,}649824 \pm 0{,}094233$
- $E[N_q] = 0{,}367546 \pm 0{,}024587$
- $E[N_s] = 7{,}085765 \pm 0{,}086543$

Poiché molto simili tra loro possono essere considerati verificati.

Nodo Home

$p_{home} = 81{,}31\% = 0{,}8131$ $E[S_i] = 12$ min $N=40$

$\lambda_{home} = \lambda_{login} * p_{home} = 4{,}039 * 0{,}8131 = 3{,}284111$ job/min

In questo nodo si hanno quattro code con quattro livelli di priorità:

$p_{j1987} = 5\% = 0{,}05$

$\lambda_{j1987} = \lambda_{home} * p_{j1987} = 3{,}284111 * 0{,}05 = 0{,}164206$ job/min

$p_{stadium} = 15\% = 0{,}15$

$\lambda_{stadium} = \lambda_{home} * p_{stadium} = 3{,}284111*0{,}15 = 0{,}492617$ job/min

$p_{card} = 30\% = 0{,}30$

$\lambda_{card} = \lambda_{home} * p_{card} = 3{,}284111*0{,}30 = 0{,}985233$ job/min

$p_{free} = 50\% = 0{,}50$

$\lambda_{free} = \lambda_{home} * p_{free} = 3{,}284111*0{,}50 = 1{,}642056$ job/min

$\mu = \frac{1}{E[Si]} = \frac{1}{12} = 0{,}083333$ job/min

$E[S] = \frac{1}{m\mu} = \frac{1}{40 * 0{,}083333} = 0{,}300001$ min

$\rho = \frac{\lambda}{m\mu} = \frac{3{,}284111}{40*0{,}083333} = 0{,}985237$

$\rho_{j1987} = \frac{\lambda_{j1987}}{m\mu} = \frac{0{,}164206}{40*0{,}083333} = 0{,}049262$

$\rho_{stadium} = \frac{\lambda_{stadium}}{m\mu} = \frac{0{,}492617}{40*0{,}083333} = 0{,}147786$

$\rho_{card} = \frac{\lambda_{card}}{m\mu} = \frac{0.985233}{40*0{,}083333} = 0{,}295571$

$\rho_{free} = \frac{\lambda_{free}}{m\mu} = \frac{1{,}642056}{40*0{,}083333} = 0{,}492619$

$p(0) = [\sum_{i=0}^{39} \frac{(40 * 0{,}985237)^i}{i!} + \frac{(40 * 0{,}985237)^{40}}{40!(1-0{,}985237)}]^{-1} = 8 \times 10^{-18}$

$Pq = \frac{(mp)^m}{m!(1-\rho)}\, p(0) = \frac{(40 * 0{,}985237)^{40}}{40!(1-0{,}985237)} * 8x10^{-18} = 0{,}442893$

$E[T_{Qj1987}] = \frac{Pq\ E[S]}{1-\rho_{j1987}} = \frac{0{,}442893 * 0{,}300001}{1-0{,}049262} = 0{,}139753$ min

$E[T_{Qstadium}] = \frac{Pq\ E[S]}{(1-\rho_{j1987})(1-\rho_{j1987}-\rho_{stadium})} = \frac{0{,}442893 * 0{,}300001}{(1-0{,}049262)(1-0{,}049262-0{,}147786)} = 0{,}174049$ min

$E[T_{Qcard}] = \frac{Pq\ E[S]}{(1-\rho_{j1987}-\rho_{stadium})(1-\rho_{j1987}-\rho_{stadium}-\rho_{card})} = \frac{0{,}442893 * 0{,}300001}{(1-0{,}049262-0{,}147786)(1-0{,}049262-0{,}147786-0{,}295571)} = 0{,}326135$

$E[T_{Qfree}] = \frac{Pq\ E[S]}{(1-\rho_{j1987}-\rho_{stadium}-\rho_{card})(1-\rho)} = \frac{0{,}442893 * 0{,}300001}{(1-0{,}049262-0{,}147786-0{,}295571)(1-0{,}985237)} = 17{,}738329$ min

$E[T_q] = p_{j1987} * E[T_{Qj1987}] + p_{free} * E[T_{Qfree}] + p_{card} * E[T_{Qcard}] + p_{stadium} * E[T_{Qstadium}] = (0{,}05*0{,}139753) + (0{,}15*0{,}174049) + (0{,}3*0{,}326135) + (0{,}5*17{,}738329) = 8{,}999698$ min

$E[T_{Sj1987}] = E[T_{Qj1987}] + E[S_i] = 0{,}139753 + 12 = 12{,}139753$

$E[T_{Sstadium}] = E[T_{Qstadium}] + E[S_i] = 0{,}174049 + 12 = 12{,}174049$

$E[T_{Scard}] = E[T_{Qcard}] + E[S_i] = 0{,}326135 + 12 = 12{,}326135$

$E[T_{Sfree}] = E[T_{Qfree}] + E[S_i] = 17{,}738329 + 12 = 29{,}738329$

$E[T_s] = E[T_q] + E[S_i] = 8{,}999698 + 12 = 20{,}899969$ min

$E[N_{Qj1987}] = \lambda_{j1987} * E[T_{Qj1987}] = 0{,}164206*0{,}263361 = 0{,}043245$

$E[N_{Qstad}] = \lambda_{stad} * E[T_{Qstad}] = 0{,}492617 * 0{,}174049 = 0{,}085739$

$E[N_{Qcard}] = \lambda_{card} * E[T_{Qcard}] = 0{,}985233 * 0{,}326135 = 0{,}321319$

$E[N_{Qfree}] = \lambda_{free} * E[T_{Qfree}] = 1{,}642056 * 17{,}738329 = 29{,}127329$

$E[N_q] = \lambda E[T_q] = 3{,}284111 * 8{,}999698 = 29{,}556007$

$E[N_{Sj1987}] = \lambda_{j1987} * E[T_{Sj1987}] = 0{,}164206*12{,}139753 = 1{,}993420$

$E[N_{Sstad}] = \lambda_{stad} * E[T_{Sstad}] = 0{,}492617 * 12{,}174049 = 5{,}997143$

$E[N_{Scard}] = \lambda_{card} * E[T_{Scard}] = 0{,}985233 * 12{,}326135 = 12{,}14411$

$E[N_{Sfree}] = \lambda_{free} * E[T_{Sfree}] = 1{,}642056 * 29{,}738329 = 48{,}832002$

$E[N_s] = \lambda E[T_s] = 3{,}284111 * 20{,}899969 = 68{,}637818$

I risultati del simulatore sono i seguenti:

1. Utilizzazione
   - $\rho = 0{,}946732 \pm 0{,}004172$
   - $\rho_{j1987} = 0{,}04818 \pm 0{,}001041$
   - $\rho_{stadium} = 0{,}144187 \pm 0{,}002029$
   - $\rho_{card} = 0{,}287856 \pm 0{,}002720$
   - $\rho_{free} = 0{,}490573 \pm 0{,}007168$

2. Tempo in coda
   - $E[T_q] = 7{,}920805 \pm 1{,}391255$
   - $E[T_{Qj1987}] = 0{,}247015 \pm 0{,}009801$
   - $E[T_{Qstadium}] = 0{,}304667 \pm 0{,}012306$
   - $E[T_{Qcard}] = 0{,}473817 \pm 0{,}027149$
   - $E[T_{Qfree}] = 15{,}389024 \pm 2{,}771272$

3. Tempo nel sistema
   - $E[T_s] = 19{,}871621 \pm 1{,}409032$
   - $E[T_{Sj1987}] = 12{,}199702 \pm 0{,}191275$
   - $E[T_{Sstadium}] = 12{,}265990 \pm 0{,}113604$
   - $E[T_{Scard}] = 12{,}498571 \pm 0{,}089414$
   - $E[T_{Sfree}] = 27{,}352488 \pm 2{,}777620$

4. Numero di job in coda
   - $E[N_q] = 25{,}177015 \pm 4{,}457229$
   - $E[N_{Qj1987}] = 0{,}039882 \pm 0{,}001808$
   - $E[N_{Qstadium}] = 0{,}147241 \pm 0{,}006706$
   - $E[N_{Qcard}] = 0{,}455178 \pm 0{,}027872$
   - $E[N_{Qfree}] = 24{,}479934 \pm 4{,}434818$

5. Numero di job nel sistema
   - $E[N_s] = 63{,}046288 \pm 4{,}557944$
   - $E[N_{Sj1987}] = 1{,}965804 \pm 0{,}042677$
   - $E[N_{Sstadium}] = 5{,}914723 \pm 0{,}085776$
   - $E[N_{Scard}] = 12{,}069404 \pm 0{,}126654$
   - $E[N_{Sfree}] = 43{,}466350 \pm 4{,}463466$

Poiché molto simili tra loro possono essere considerati verificati.

Nodo WomenPay

$p_{women\_loss} = 5\% = 0{,}05$ $E[S_i] = 4$ min $N=5$

$\lambda_{womenPay} = \lambda_{women} * (1 - p_{women\_loss}) = 0{,}195488*0{,}95 = 0{,}185714$

$\mu = \frac{1}{E[Si]} = \frac{1}{4} = 0{,}25$ job/min

$E[S] = \frac{1}{m\mu} = \frac{1}{5 * 0{,}25} = 0{,}8$ min

$\rho = \frac{\lambda}{m\mu} = \frac{0{,}185714}{5*0{,}25} = 0{,}148571$

Essendo un sistema M/M/m a coda singola:

$p(0) = [\sum_{i=0}^{4} \frac{(5 * 0{,}148571)^i}{i!} + \frac{(5 * 0{,}148571)^5}{5!(1-0{,}148571)}]^{-1} = 0{,}475739$

$Pq = \frac{(mp)^m}{m!(1-\rho)}\, p(0) = \frac{(5 * 0{,}148571)^5}{5!(1-0{,}148571)} * 0{,}475739 = 0{,}000105$

$E[T_q] = \frac{Pq\ E[S]}{1-\rho} = \frac{0{,}000105 * 0{,}8}{1-0{,}148571} = 0{,}000098$ min

$E[T_s] = E[T_q] + E[S_i] = 0{,}000098 + 4 = 4{,}000098$ min

$E[N_q] = \lambda E[T_q] = 0{,}185714 * 0{,}000098 = 0{,}000018$
$E[N_s] = \lambda E[T_s] = 0{,}185714 * 4{,}000098 = 0{,}742874$
I risultati del simulatore sono:

- $\rho = 0{,}148628 \pm 0{,}001772$
- $E[T_q] = 0{,}000101 \pm 0{,}000048$
- $E[T_s] = 4{,}000278 \pm 0{,}033927$
- $E[N_q] = 0{,}000105 \pm 0{,}000084$
- $E[N_s] = 0{,}743343 \pm 0{,}008871$

Poiché molto simili tra loro possono essere considerati verificati.

Nodo AwayPay

$P_{away\_loss} = 30\% = 0{,}3$ $E[S_i] = 4$ min $N=5$
$\lambda_{awayPay} = \lambda_{away} * (1 - p_{away\_loss}) = 0{,}559402 * 0{,}7 = 0{,}391581$
$\mu = \frac{1}{E[Si]} = \frac{1}{4} = 0{,}25$ job/min
$E[S] = \frac{1}{m\mu} = \frac{1}{5 * 0{,}25} = 0{,}8$ min
$\rho = \frac{\lambda}{m\mu} = \frac{0{,}391581}{5 * 0{,}25} = 0{,}313265$
Essendo un sistema M/M/m a coda singola:
$p(0) = [\sum_{i=0}^{5} \frac{(5 * 0{,}313265)^i}{i!} + \frac{(5 * 0{,}313265)^5}{5!(1-0{,}313265)}]^{-1} = 0{,}238750$
$Pq = \frac{(mp)^m}{m!(1-\rho)}\, p(0) = \frac{(5 * 0{,}313265)^5}{5!(1-0{,}313265)} * 0{,}23875 = 0{,}027314$
$E[T_q] = \frac{Pq\ E[S]}{1-\rho} = \frac{0{,}027314 * 0{,}8}{1-0{,}313265} = 0{,}031819$ min
$E[T_s] = E[T_q] + E[S_i] = 0{,}031819 + 4 = 4{,}031819$ min
$E[N_q] = \lambda E[T_q] = 0{,}391581 * 0{,}031819 = 0{,}012459$
$E[N_s] = \lambda E[T_s] = 0{,}391581 * 4{,}031819 = 1{,}578784$
I risultati del simulatore sono:

- $\rho = 0{,}293775 \pm 0{,}006664$
- $E[T_q] = 0{,}033769 \pm 0{,}012269$
- $E[T_s] = 4{,}013615 \pm 0{,}034450$
- $E[N_q] = 0{,}016671 \pm 0{,}001973$
- $E[N_s] = 1{,}668512 \pm 0{,}077020$

Poiché molto simili tra loro possono essere considerati verificati.

Nodo HomePay

$P_{home\_loss} = 10\% = 0{,}1$ $E[S_i] = 4$ min $N=15$
$\lambda_{homePay} = \lambda_{home} * (1 - p_{home\_loss}) = 3{,}284111 * 0{,}9 = 2{,}95570$
$\mu = \frac{1}{E[Si]} = \frac{1}{4} = 0{,}25$ job/min
$E[S] = \frac{1}{m\mu} = \frac{1}{15 * 0{,}25} = 0{,}266667$ min
$\rho = \frac{\lambda}{m\mu} = \frac{2{,}9557}{15 * 0{,}25} = 0{,}788187$
Essendo un sistema M/M/m a coda singola:
$p(0) = [\sum_{i=0}^{14} \frac{(15 * 0{,}788187)^i}{i!} + \frac{(15 * 0{,}788187)^{15}}{15!(1-0{,}788187)}]^{-1} = 0{,}0000007$
$Pq = \frac{(mp)^m}{m!(1-\rho)}\, p(0) = \frac{(15 * 0{,}788187)^{15}}{15!(1-0{,}788187)} * 0{,}000007 = 0{,}03114$
$E[T_q] = \frac{Pq\ E[S]}{1-\rho} = \frac{0{,}03114 * 0{,}8}{1-0{,}788187} = 0{,}117613$ min
$E[T_s] = E[T_q] + E[S_i] = 0{,}117613 + 4 = 4{,}117613$ min
$E[N_q] = \lambda E[T_q] = 2{,}95570 * 0{,}117613 = 0{,}347629$
$E[N_s] = \lambda E[T_s] = 2{,}95570 * 4{,}117613 = 12{,}170429$
I risultati del simulatore sono:

- $\rho = 0{,}786558 \pm 0{,}009026$
- $E[T_q] = 0{,}116336 \pm 0{,}048388$
- $E[T_s] = 4{,}190788 \pm 0{,}0066359$
- $E[N_q] = 0{,}320858 \pm 0{,}050446$
- $E[N_s] = 12{,}749228 \pm 0{,}009026$

Poiché molto simili tra loro possono essere considerati verificati.

VII. VALIDAZIONE

Per fare una validazione accurata si dovrebbe disporre di dati ufficiali con i quali confrontare le simulazioni ottenute dal modello. Purtroppo, questi dati non sono reperibili, e, di conseguenza, è stato scelto un approccio alla validazione differente tramite l'utilizzo di controlli di consistenza. Per effettuare tali controlli sono stati elaborati alcuni casi di test con configurazioni differenti del sistema, in termini di numero di server per nodo o di tassi di arrivo, con lo scopo di verificare che il comportamento in output sia conforme a questi cambiamenti. In particolare, ci si aspetta che, a parità di server per nodo ed all'aumentare del λ, ci sia un aumento dei tempi medi di attesa su tutte le code, con la possibilità di rendere alcuni nodi non stazionari. E, inoltre, mantenendo il tasso di ingresso costante ed aumentando il numero di server su ogni nodo, ci si aspetta che i tempi di attesa diminuiscano su ogni nodo. Di conseguenza, sono stati scelti due test di validazione che sono stati effettuati sui nodi Women ed Home, rappresentativi dell'intero sistema.
Il primo test ha come obiettivo quello di verificare che all'aumentare del λ vi sia un incremento dell'utilizzazione e dei tempi in coda e nel sistema; per fare ciò sono stati messi a paragone i risultati di due differenti simulazioni:

- La prima simulazione ha la stessa configurazione con cui è stata fatta la verifica: 10 server per il nodo Women e 40 server per il nodo Home, con un λ di 4,039 job/min.
- La seconda simulazione ha la stessa configurazione di server ma ha λ raddoppiato, quindi pari a 8,078 job/min.

Dalle due simulazioni è possibile evincere i seguenti cambiamenti. In Women l'utilizzazione aumenta da 0,234626 a 0,469236, il tempo in coda aumenta da 0,000308 a 0,063872 ed il tempo nel sistema aumenta da 11,995645 a 12,058411. Per quanto riguarda Home l'utilizzazione passa 0,946732 ad essere maggiore di 1 portando così il nodo a perdere la sua stazionarietà.
Per il secondo test, invece, l'obiettivo è quello di verificare che, dato un λ costante, all'aumentare del numero di server nel nodo ci sia un abbassamento dell'utilizzazione e dei tempi in coda e nel sistema; anche in questo caso, per fare ciò, sono stati messi a paragone i risultati di due differenti simulazioni:

- La prima simulazione ha la stessa configurazione con cui è stata fatta la verifica: 10 server per il nodo Women e 40 server per il nodo Home, con un λ di 4,039 job/min.
- La seconda simulazione ha lo stesso λ ma ha un numero di server maggiorato del 50% per ogni nodo, nello specifico 15 per Women e 60 per Home.

Dalle due simulazioni è possibile evincere i seguenti cambiamenti. In Women l'utilizzazione scende da 0,234626 a 0,156417, il tempo in coda scende da 0,000308 fino a toccare lo 0, e, di conseguenza, il tempo nel sistema è molto vicino al tempo di servizio. Per quanto riguarda Home, invece, l'utilizzazione scende da 0,946732 a 0,635167, il tempo in coda scende da 7,920805 a 0,000442 ed il tempo nel sistema passa da 19,871621 a 12,03127.
A questo punto anche la fase di validazione può essere considerata superata, dato che, per entrambi i casi di test, il

comportamento del modello si è rivelato essere congruo alle aspettative.

VIII. Simulazione ad orizzonte finito

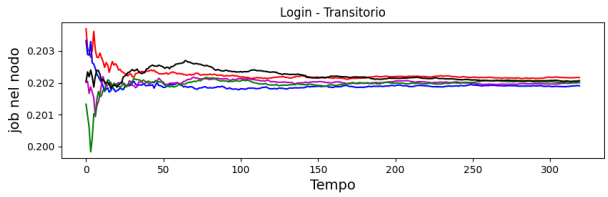
La simulazione ad orizzonte finito viene effettuata tramite la tecnica della replicazione, in modo da ottenere stime indipendenti della stessa statistica transitoria. Nel caso in analisi vengono effettuate 64 run consecutive con un tempo di stop di 1440 minuti, necessario per poter analizzare l'andamento del sistema nell'arco di un giorno intero. Per garantire che non ci sia sovrapposizione tra le replicazioni viene utilizzata la funzione PlantSeed in modo tale che ogni run abbia un seed differente. Per quanto riguarda il calcolo dell'intervallo di confidenza è stata utilizzata la formula $\frac{t^*\sigma}{\sqrt{n-1}}$, dove σ è la deviazione standard, n-1 è il numero di ripetizioni considerate e t^* è il valore critico ottenuto dall'inverso della distribuzione di Student con n-1 gradi di libertà. In particolare, t^* è calcolato come idfStudent (n-1, 1-$\frac{\alpha}{2}$); dove α è il parametro di confidenza (scelto come 0,05) che determina di quanto ci si possa scostare dalla media in positivo o in negativo.

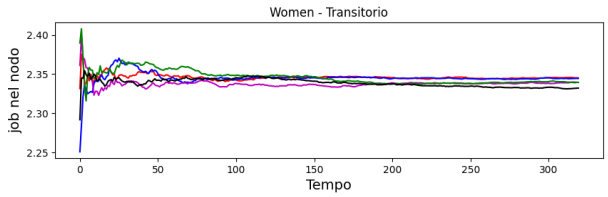
IX. Simulazione ad orizzonte infinito

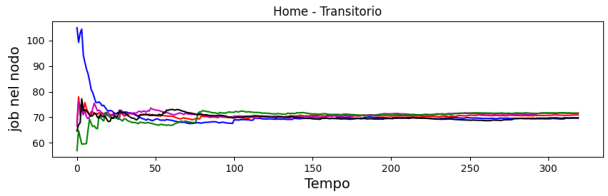
Mentre la simulazione ad orizzonte finito va ad analizzare il comportamento del sistema in uno stato transiente per un periodo di tempo limitato, la simulazione ad orizzonte infinito va ad analizzare il comportamento del simulatore in un regime stazionario per un periodo di tempo illimitato. Per poter implementare questo tipo di simulazione viene utilizzato il metodo delle batch means. Questa tecnica consiste nell'eseguire una run molto lunga, per ridurre l'influenza dello stato iniziale, e nel dividerla in diverse batch, organizzate in modo che lo stato iniziale della batch successiva coincida con lo stato finale di quella precedente. Per poter utilizzare questa tecnica sono necessari due parametri: K, che indica il numero delle batch, ed B, che indica la lunghezza delle batch. Nel caso in analisi K è stato posto a 64 e B a 1024. A differenza della simulazione ad orizzonte finito, che ha un vero e proprio istante di stop, tale simulazione termina quando tutte le batch di tutti i centri sono riempite.

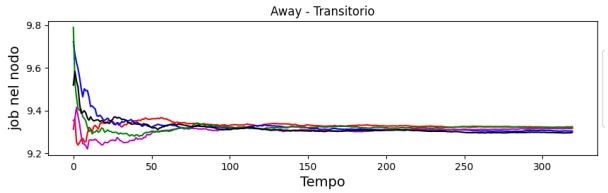
X. Analisi del transitorio

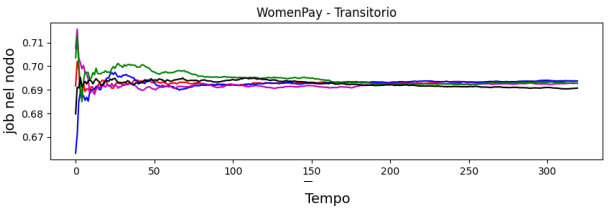
L'analisi del transitorio è un passaggio necessario per verificare che il sistema raggiunga la stazionarietà. Per condurre tale analisi vengono effettuate alcune simulazioni ad orizzonte finito con seed indipendenti in modo da verificare una convergenza non casuale. In questo simulatore la durata della simulazione ad orizzonte finito è di default impostata a 1440 minuti; questo perché l'idea è quella di andare a vedere l'andamento della simulazione nel corso di una intera giornata. Il problema è che tale durata potrebbe essere troppo breve per garantire la convergenza in tutti i nodi. Allora, per riuscire ad ottenere la convergenza, il tempo di simulazione è stato aumentato fino a 7200 minuti, ovvero cinque giorni. Questa scelta è stata dettata dal fatto che, come precedentemente citato, i biglietti di una singola partita restano in commercio per cinque giorni prima di essere sostituiti dai biglietti della partita successiva. Tale aumento del tempo di simulazione ha permesso di dimostrare il raggiungimento della stazionarietà. La statistica presa in considerazione è il numero di job all'interno del nodo.
Di seguito i risultati in forma grafica.

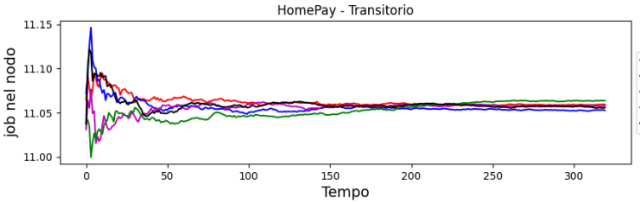


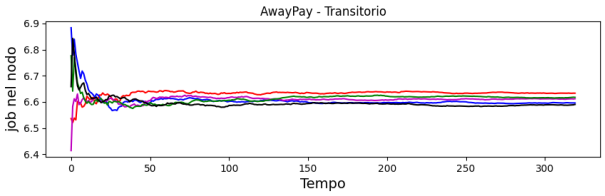










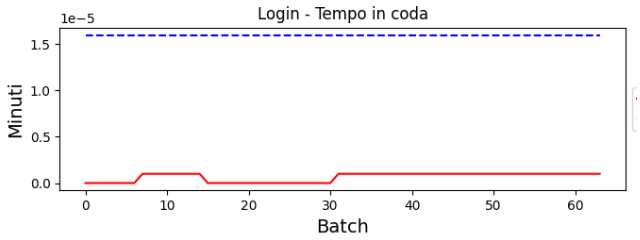


XI. ESPERIMENTI

Il simulatore in analisi è stato implementato con l'obiettivo di andare a diminuire gli enormi tempi di attesa per l'acquisto di biglietti sul sito della Juventus FC. Per raggiungere questo obiettivo è necessario elaborare alcuni esperimenti per determinare la configurazione ottimale di serventi per ogni singolo nodo in modo da rispettare dei requisiti di QoS sui tempi di attesa, evitando però, allo stesso tempo, di sovradimensionare il sistema utilizzando così più risorse del dovuto. Poiché la vendita di biglietti segue di pari passo la stagione calcistica, tali requisiti di QoS devono essere rispettati per circa nove mesi. Proprio per questo motivo gli esperimenti verranno eseguiti e studiati sul regime stazionario; quindi facendo riferimento alle simulazioni ad orizzonte infinito. Per riuscire ad ottenere la configurazione ottima di serventi si partirà dalla configurazione utilizzata in fase di verifica per poi apportare gli opportuni cambiamenti.

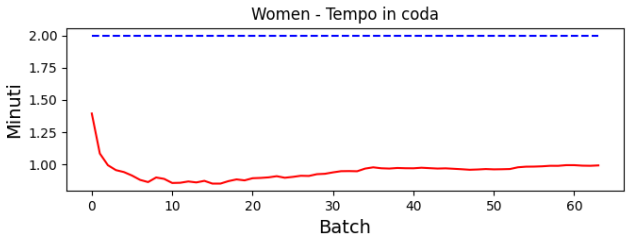
Legenda: linea tratteggiata = massimo delay ammesso
linea continua = delay del nodo

Nodo Login:
In fase di verifica tale nodo aveva 3 serventi ed un tempo di attesa in coda di 0,000024 minuti. Poiché l'obiettivo è ottenere un tempo di delay massimo di un millesimo di secondo bisogna far si che questo valore scenda al di sotto dei 0,000017 minuti. Dato che la configurazione di partenza è già molto vicina all'obiettivo, poiché il delay è di poco più grande del consentito, il primo esperimento consiste nel vedere se con un solo servente in più il requisito di QoS venga rispettato. Infatti, andando ad aumentare il numero di serventi a 4 si ottiene un tempo di attesa in coda pari a 0,000001.



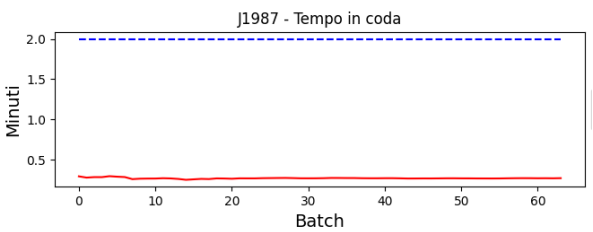
Da ciò si deduce che la configurazione ottimale per il nodo Login è di 4 serventi.

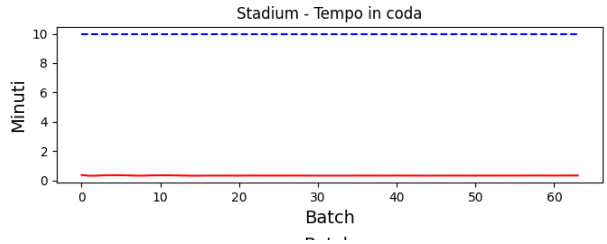
Nodo Women:
In fase di verifica tale nodo aveva 10 serventi ed un tempo di attesa in coda pari a 0,000051 minuti. Poiché l'obiettivo è ottenere un tempo di delay massimo di 2 minuti si può notare fin da subito che il nodo è sovradimensionato poiché il tempo di delay attuale è di molto inferiore al massimo consentito. Per ovviare a tale problema il primo esperimento ha portato la configurazione da 10 serventi a 4, ottenendo così un tempo di attesa in coda pari a 0,992745 minuti: sicuramente meglio della configurazione precedente ma c'è ancora un minuto di scarto da poter recuperare. Per tale motivo è stato effettuato un secondo esperimento che ha portato il numero di serventi a 3, ottenendo così un tempo di delay di 5,361796: decisamente troppo. Visti i risultati del secondo esperimento si è tornati a considerare la configurazione del primo esperimento.

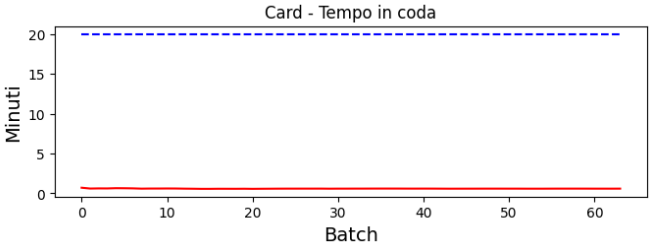


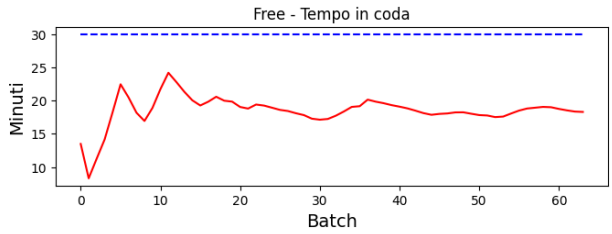
Da ciò si deduce che la configurazione ottimale per il nodo Women è di 4 serventi.

NodoHome:
Il nodo Home ha quattro code di priorità ognuna delle quali ha un proprio requisito da rispettare. La coda J1987, a priorità massima, deve avere un delay di 2 minuti, la coda Stadium di 10 minuti, la coda Card di 20 minuti e la coda Free di 30 minuti. Per la modellazione degli esperimenti si è partiti dalla coda a priorità minima Free e dal suo relativo requisito di QoS. In fase di verifica il nodo Home aveva una configurazione di 40 serventi ed il delay della coda Free era pari a 6,069331 min. Tale valore rispetta il QoS ma è anche indice di sovradimensionamento poiché inferiore al delay massimo consentito di circa 24 minuti. Di conseguenza si è pensato a diminuire il numero di serventi da 40 a 39 ottenendo così un tempo di attesa per la coda Free di 18,216170 minuti. Poiché tale valore è vicino al limite massimo consentito ma si ha ancora uno scarto di 12 minuti si è provato ad abbassare il numero di serventi a 38. Purtroppo, con questa configurazione il tempo di attesa sale fino a 300 minuti, per questo motivo si è tornati alla configurazione con 39 serventi. Per quanto riguarda le altre code si hanno dei tempi di delay molto piccoli, ovvero 0,271989 minuti per la coda J1987; 0,342173 minuti per la coda Stadium e 0,596577 minuti per la coda Card. I tempi di delay in queste tre code sono molto più piccoli rispetto ai massimi consentiti, ma in questo caso non si può parlare di sovradimensionamento poiché questa è l'unica configurazione possibile per rispettare i requisiti di QoS della coda Free.



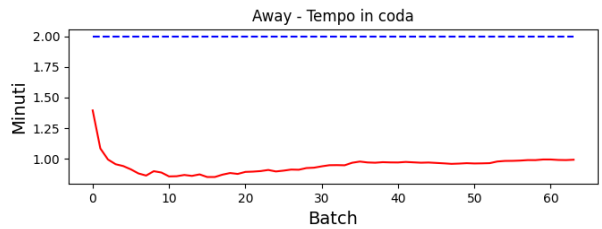






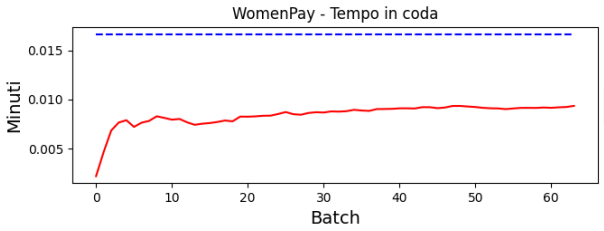
Da ciò si deduce che la configurazione ottimale per il nodo Home è di 39 serventi.

Nodo Away:
In fase di verifica tale nodo aveva 10 serventi ed un tempo di attesa in coda pari a 0,367546 minuti. Poiché l'obiettivo è ottenere un tempo di delay massimo di 2 minuti si può notare fin da subito che il nodo è sovradimensionato poiché il tempo di delay attuale è inferiore rispetto al massimo consentito. Per ovviare a tale problema il primo esperimento ha portato la configurazione da 10 serventi ad 8, ottenendo così un tempo di attesa in coda pari a 2,235570 minuti. Poiché tale valore è leggermente più alto del limite di 2 minuti è stato necessario cambiare ulteriormente la configurazione portando il numero di serventi a 9. In questo caso il tempo di delay diventa di 0,822115 minuti, in piena linea con i requisiti di QoS.



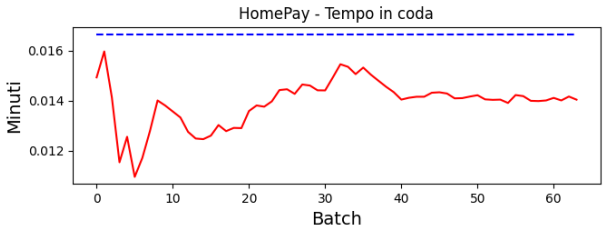
Da ciò si deduce che la configurazione ottimale per il nodo Away è di 9 serventi.

Nodo WomenPay:
In fase di verifica tale nodo aveva 5 serventi ed un tempo di attesa in coda pari a 0,000098 minuti. Poiché l'obiettivo è ottenere un tempo di delay massimo di 1 secondo, ovvero 0,016667 minuti, si può notare che l'attuale delay è al di sotto del limite massimo consentito. Il primo esperimento ha quindi portato la configurazione del nodo da 5 serventi a 2, ottenendo così un tempo di attesa in coda di 0,144325. Poiché tale valore è troppo elevato un secondo esperimento ha portato il numero di serventi a 3, ottenendo il delay ottimale di 0,009348



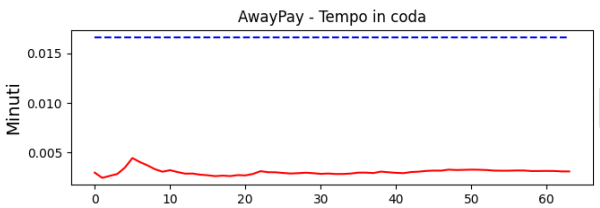
.
Da ciò si deduce che la configurazione ottimale per il nodo WomenPay è di 3 serventi.

Nodo HomePay:
In fase di verifica tale nodo aveva 15 serventi ed un tempo di attesa in coda pari a 0,032976 minuti. Poiché l'obiettivo è ottenere un tempo di delay massimo di 1 secondo è necessario far scendere quel valore al di sotto della soglia dei 0,016667 minuti. Tramite un esperimento si è portato il numero di serventi da 15 a 16 e si è ottenuto un tempo di attesa in coda che rispetta i requisiti di QoS: 0,014037 minuti.



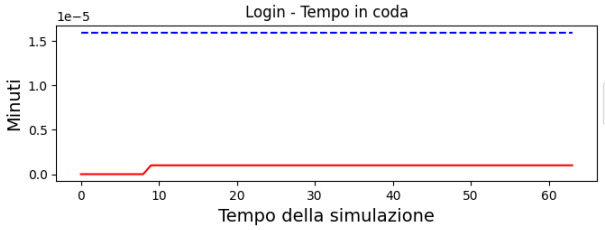
Da ciò si deduce che la configurazione ottimale per il nodo HomePay è di 16 serventi.

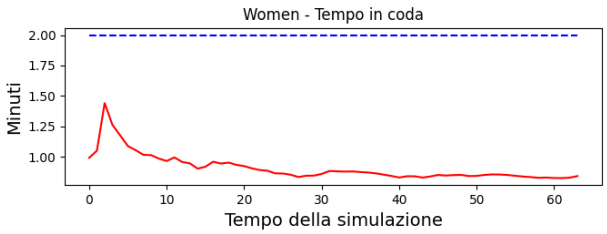
Nodo AwayPay:
In fase di verifica tale nodo aveva 5 serventi ed un tempo di attesa in coda di 0,010973 minuti. Poiché l'obiettivo è ottenere un tempo di delay massimo di un secondo tale valore sembra già essere ottimale poiché molto vicino agli 0,0166667 minuti. Per vedere se esiste una configurazione migliore di serventi, tramite un esperimento si è diminuito il numero di serventi da 5 a 4, ottenendo però un valore troppo grande pari a 0,070613 minuti. In seguito a tale risultato si è ripristinato il tutto alla configurazione di partenza.

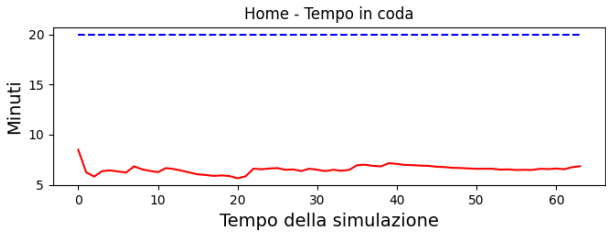


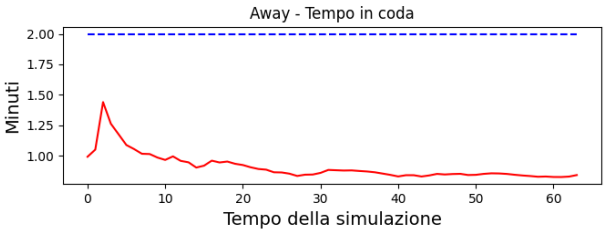
Da ciò si deduce che la configurazione ottimale per il nodo AwayPay è di 5 serventi.

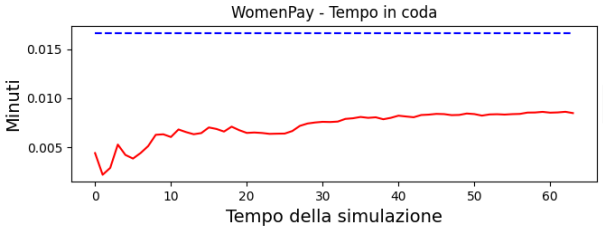
Per motivi di completezza, nonostante la loro non necessarietà ai fini dell'analisi appena conclusa, vengono proposti i grafici relativi ai tempi di attesa in coda di tutti i nodi durante la simulazione ad orizzonte finito con termine fissato a 1440 minuti e configurata con il numero ottimale di serventi per nodo

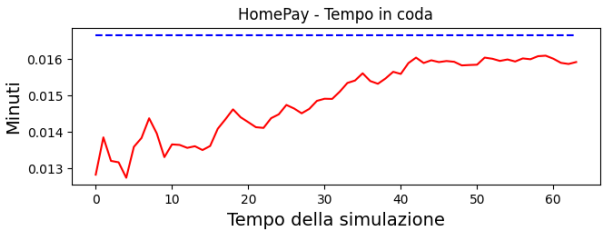


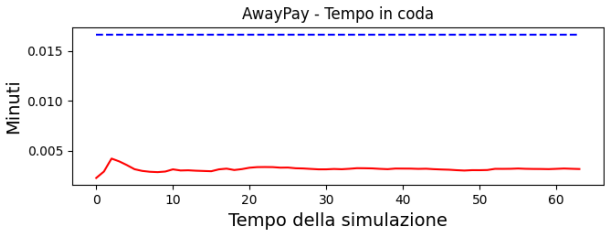












XII. CONCLUSIONI

Al termine delle varie simulazioni e dell'analisi che ne è conseguita è possibile determinare la configurazione ottimale del sistema. Per il nodo di login è necessario utilizzare 4 serventi per garantire un delay massimo di un millesimo di secondo. Per i nodi Women e Away sono necessari, rispettivamente, 4 e 9 serventi, in modo da garantire un tempo massimo di attesa in coda di 2 minuti. Per il nodo Home, contenente quattro code con diversa priorità, sono necessari 39 serventi per garantire un delay massimo di 30 minuti per la coda a priorità più bassa e, di conseguenza, per garantire i relativi requisiti di QoS per le altre tre code. Infine, per i nodi WomenPay, AwayPay e HomePay, sono necessari, rispettivamente, 3, 5 e 16 serventi; in modo da garantire un delay massimo di 1 secondo.