

# Relazione Progetto Simulazione di Sistemi

Corso di Laurea in  
Informatica Magistrale

Anno Accademico  
2023-2024

Loretti Andrea [0001097539]

# Indice

|           |   |           |
|-----------|---|-----------|
| <b>1</b>  | <b>Introduzione</b>   | <b>3</b>  |
| <b>2</b>  | <b>Modello Analizzato</b>   | <b>3</b>  |
| 2.1       | Politiche di Servizio . . . . .   | 3         |
| 2.1.1     | Politica Exact-N . . . . .  | 3         |
| 2.1.2     | Politica N-Limited . . . . .  | 3         |
| <b>3</b>  | <b>Descrizione implementazione</b>  | <b>4</b>  |
| 3.1       | Codice Sorgente . . . . .   | 4         |
| 3.1.1     | Modifiche nel File <code>PassiveQueue.cc</code> . . . . .                       | 4         |
| 3.1.2     | Modifiche nel File <code>Server.h</code> . . . . .                              | 4         |
| 3.1.3     | Modifiche nel File <code>Server.cc</code> . . . . .                             | 5         |
| <b>4</b>  | <b>Parametri della Simulazione</b>  | <b>5</b>  |
| <b>5</b>  | <b>Analisi transiente iniziale</b>  | <b>6</b>  |
| <b>6</b>  | <b>Misure di Prestazione</b>  | <b>15</b> |
| <b>7</b>  | <b>Metodo di Simulazione</b>  | <b>16</b> |
| <b>8</b>  | <b>Risultati</b>  | <b>16</b> |
| <b>9</b>  | <b>Analisi dei Risultati</b>  | <b>16</b> |
| 9.1       | Analisi Generale . . . . .  | 16        |
| 9.2       | Analisi rispetto $\lambda$ . . . . .  | 17        |
| 9.3       | Analisi rispetto $m_1$ . . . . .  | 19        |
| 9.4       | Analisi rispetto $m_2$ . . . . .  | 21        |
| 9.5       | Analisi rispetto $N$ . . . . .  | 23        |
| <b>10</b> | <b>Convalida del Modello</b>  | <b>25</b> |
| 10.1      | Verifica del Teorema di Little per $\lambda = 1.25$ con tolleranza 3% . . . . . | 25        |
| 10.2      | Verifica del Teorema di Little per $\lambda = 2$ . . . . .                      | 26        |
| <b>11</b> | <b>Conclusioni</b>  | <b>26</b> |
| 11.1      | Performance del Sistema . . . . .   | 26        |
| 11.2      | Dinamiche di Comportamento Strategico . . . . .                                 | 27        |
| 11.3      | Validazione del Modello . . . . .   | 27        |
| 11.4      | Considerazioni Finali . . . . .   | 27        |

# 1 Introduzione

Questa relazione riguarda l'analisi di un modello di simulazione basato su una variante del comportamento strategico in una coda tandem con server alternato, come descritto nell'articolo di riferimento [DHY20]. Il modello di simulazione è stato realizzato considerando specifici parametri di interarrivo, tempi di servizio e politiche di gestione.

## 2 Modello Analizzato

Il modello può essere descritto come una coda tandem con server alternato.

### 2.1 Politiche di Servizio

Nell'articolo di riferimento vengono descritte due politiche di gestione utilizzate nelle simulazioni:

#### 2.1.1 Politica Exact-N

Sotto la politica Exact-N, il server completa il servizio di esattamente  $N$  clienti nella prima coda ( $Q_1$ ) prima di passare alla seconda coda ( $Q_2$ ) e servire quei  $N$  clienti, per poi ritornare a  $Q_1$  per iniziare un nuovo ciclo. Questa politica porta a un comportamento misto dei clienti, che possono decidere di seguire o evitare la folla.

#### 2.1.2 Politica N-Limited

Nella politica N-Limited, il server passa da  $Q_1$  a  $Q_2$  quando la prima coda è vuota o quando è stato raggiunto il numero massimo di  $N$  clienti serviti. Questo rende il sistema conservativo nel lavoro, poiché il server non rimane mai inattivo quando ci sono clienti in attesa, portando a un comportamento dei clienti che tende ad evitare la folla.

### 3 Descrizione implementazione

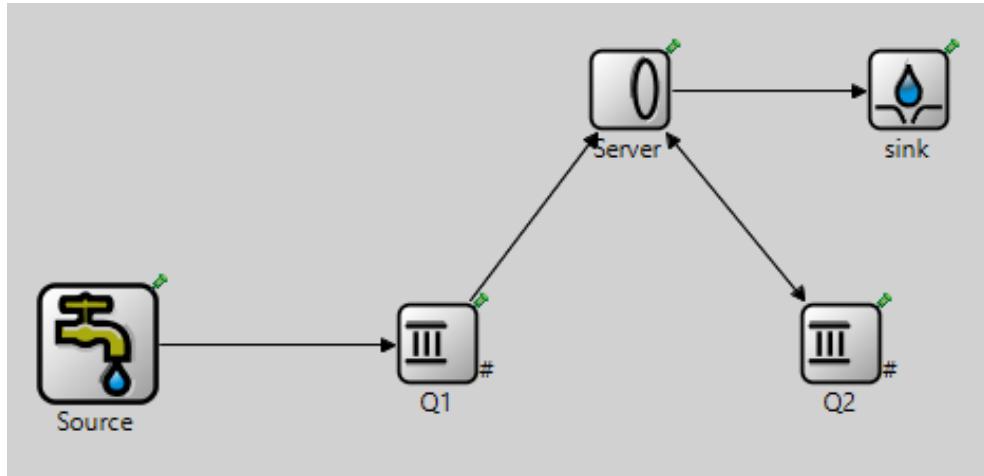


Figura 3.1: Rapresentazione in OMNeT++ del modello simulato

#### 3.1 Codice Sorgente

##### 3.1.1 Modifiche nel File `PassiveQueue.cc`

Per migliorare l'efficienza del sistema, sono state apportate diverse modifiche significative nel file `PassiveQueue.cc`. In particolare, è stata aggiunta la logica per identificare se il server è associato alla coda Q1 e per gestire lo stato di allocazione del server in base a questa informazione. Questa modifica permette al sistema di deallocare il server se è libero e di allocarlo se è occupato, migliorando così l'efficienza nella gestione dei job. Inoltre, la gestione dello stato della coda è stata ottimizzata per evitare sprechi di risorse e garantire un'allocazione più dinamica dei job ai server.

##### 3.1.2 Modifiche nel File `Server.h`

Nel file `Server.h`, sono state aggiunte variabili per tenere traccia del numero di clienti serviti nelle code Q1 e Q2, nonché flag per indicare se il server sta attualmente servendo Q1 o Q2 e se le code sono vuote. Queste variabili includono `customersServedQ1`, `customersServedQ2`, `fromQueue1`, `isQ1Empty` e `isQ2Empty`. Queste modifiche sono state necessarie per implementare le nuove strategie di servizio e per monitorare meglio lo stato del sistema. Inoltre, sono state aggiunte variabili per gestire la strategia di servizio (Exact-N o N-Limited) e la distribuzione delle probabilità, come `N`, `pDistribution` e `vDistribution`. Queste modifiche

permettono di modellare meglio scenari reali in cui i server possono avere regole specifiche per cambiare coda, migliorando la precisione delle simulazioni.

### 3.1.3 Modifiche nel File Server.cc

Il file `Server.cc` ha subito modifiche significative per implementare le politiche di servizio Exact-N e N-Limited. È stata aggiunta logica per cambiare la coda servita in base alla politica di servizio selezionata e al numero di clienti serviti. Ad esempio, sono state implementate le funzioni `Exact_N` e `N_Limited` per gestire queste politiche di servizio. La funzione `Exact_N` gestisce il passaggio tra le code dopo aver servito esattamente  $N$  clienti, mentre la funzione `N_Limited` gestisce il passaggio quando la coda è vuota o quando è stato raggiunto il numero massimo di  $N$  clienti serviti. Inoltre, sono stati inclusi controlli per verificare se le code  $Q_1$  e  $Q_2$  sono vuote e per allocare o deallocare il server di conseguenza.

Queste modifiche migliorano l'efficienza del server e garantiscono una gestione più dinamica e reattiva dei job. La verifica dello stato delle code prima di effettuare transizioni tra le code serve a evitare sprechi di risorse e a garantire che il server serva sempre la coda con job in attesa, aumentando l'efficienza complessiva del sistema.

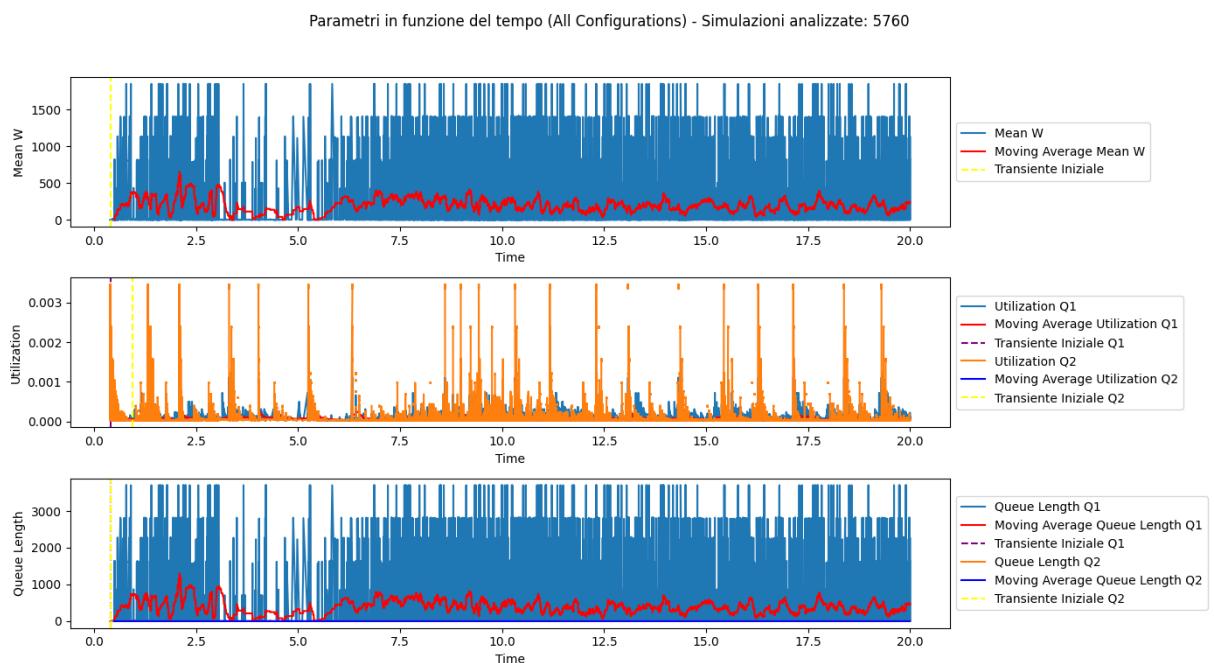
Queste modifiche rendono il modello di simulazione più robusto e realistico, permettendo una migliore analisi del comportamento strategico in sistemi di code tandem con server alternato.

## 4 Parametri della Simulazione

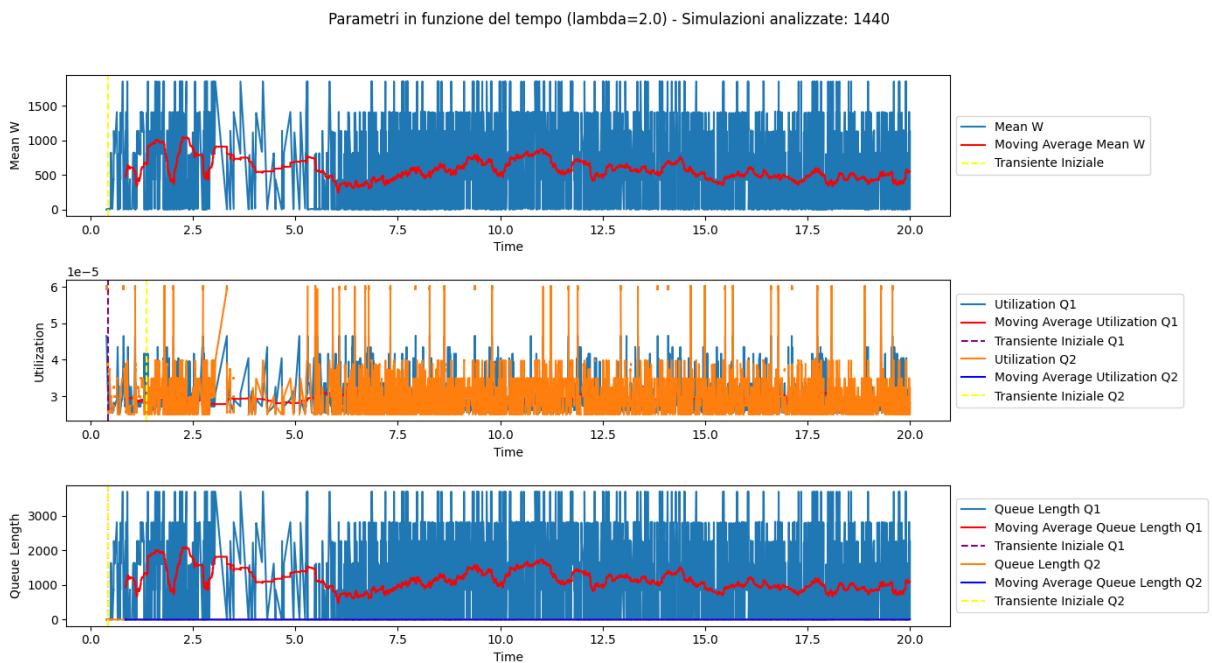
- **Tempo di interarrivo degli utenti:** Distribuzione esponenziale con media  $\frac{1}{\lambda}$
- **Tempi di servizio dei serventi  $Q_i$ :** Distribuzione esponenziale con media  $m_i$
- **Costo di switch:** Pari a zero
- **Politiche di servizio:** Exact-N, N-Limited
- **Parametri specifici:**
  - $\lambda$ : 2.0, 1.4, 1.2, 1.0
  - $m_1, m_2$ : 0.4, 0.3, 0.25, 0.20
  - $p$ : Uniformemente distribuito in  $[a, b]$

- $V$ : Uniformemente distribuito in  $[c, b]$
- $[a, b]$ :  $[1, 3], [4, 6], [8, 10]$
- $[c, b]$ :  $[14, 16], [18, 20], [20, 22]$
- CW: 1.0
- N: 1, 2, 3, 4, 5

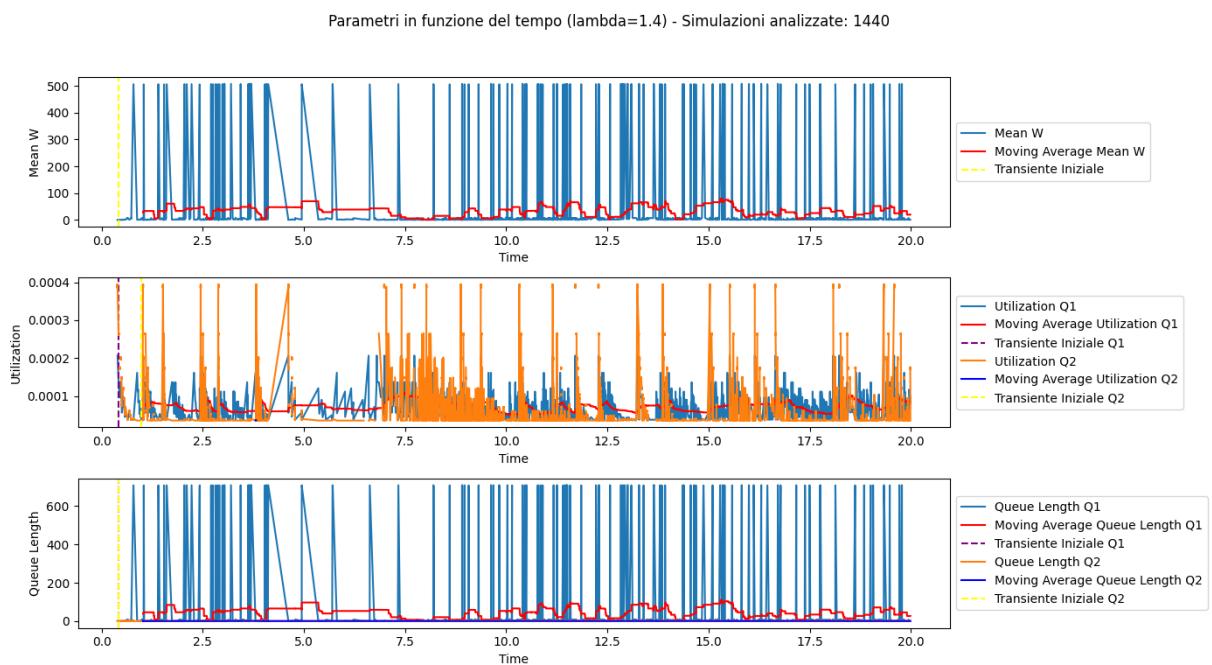
## 5 Analisi transiente iniziale



**Figura 5.1:** Panoramica di tutte le simulazioni.

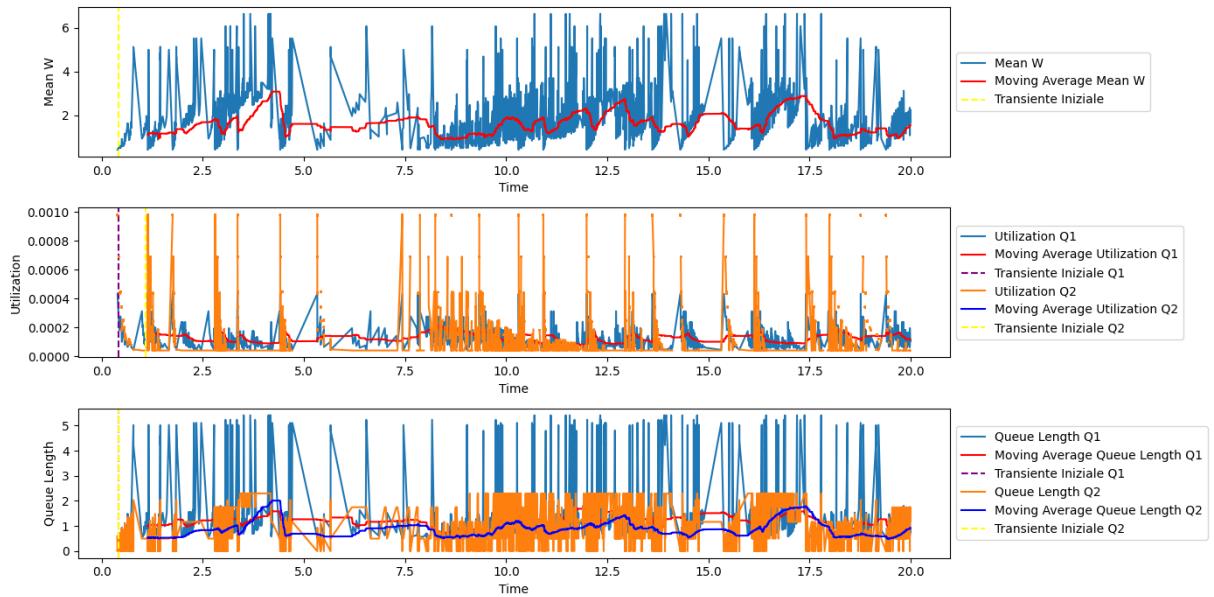


**Figura 5.2:** Panoramica delle simulazioni con  $\lambda = 2.0$ .



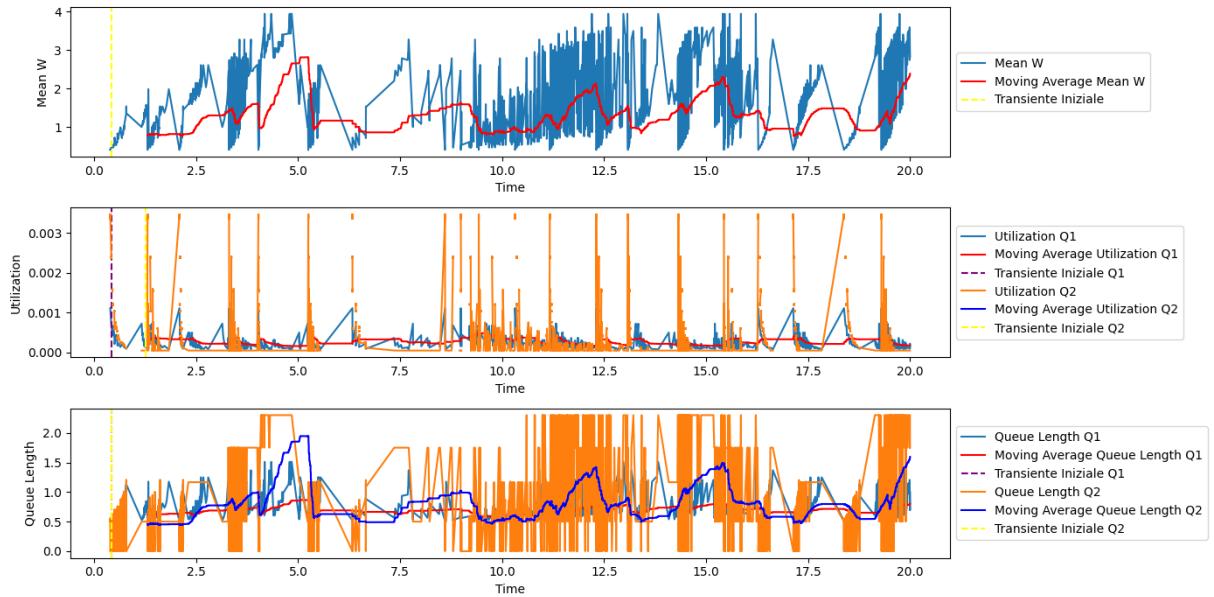
**Figura 5.3:** Panoramica delle simulazioni con  $\lambda = 1.4$ .

Parametri in funzione del tempo ( $\lambda=1.2$ ) - Simulazioni analizzate: 1440

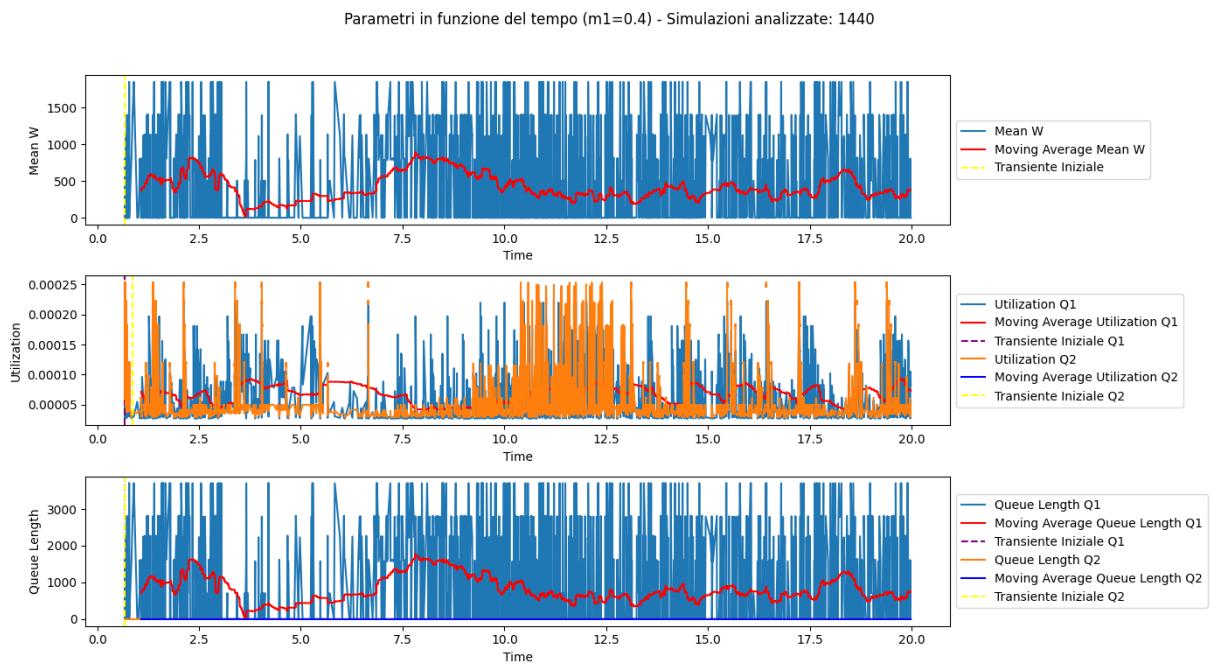


**Figura 5.4:** Panoramica delle simulazioni con  $\lambda = 1.2$ .

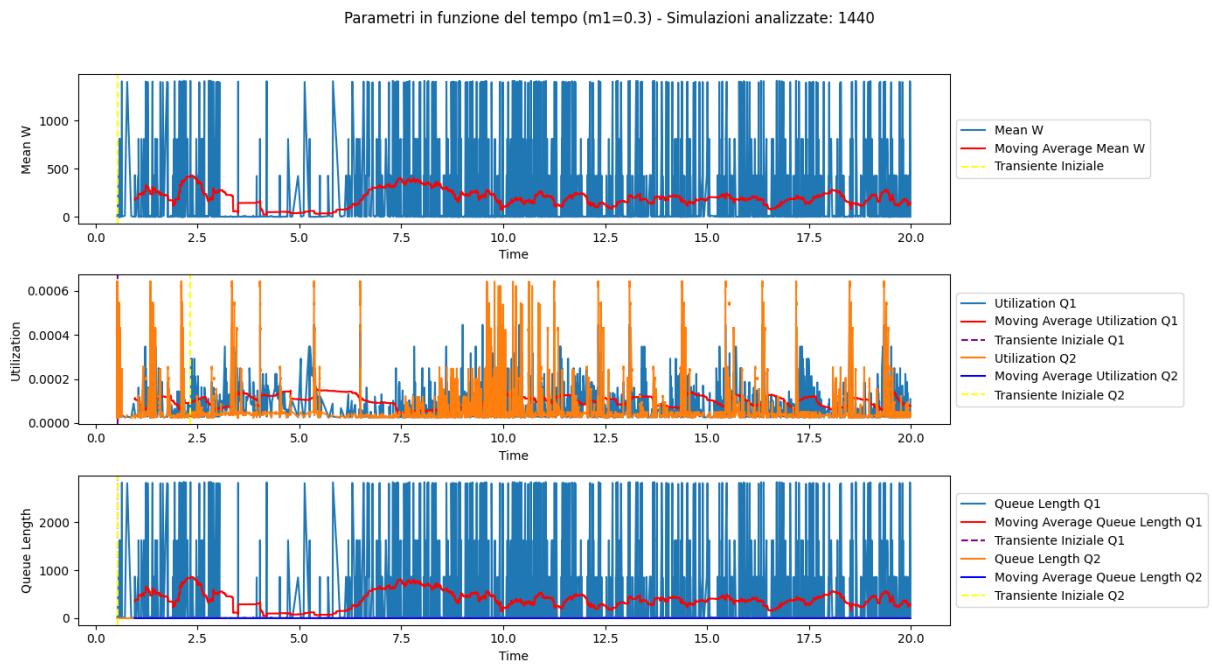
Parametri in funzione del tempo ( $\lambda=1.0$ ) - Simulazioni analizzate: 1440



**Figura 5.5:** Panoramica delle simulazioni con  $\lambda = 1.0$ .

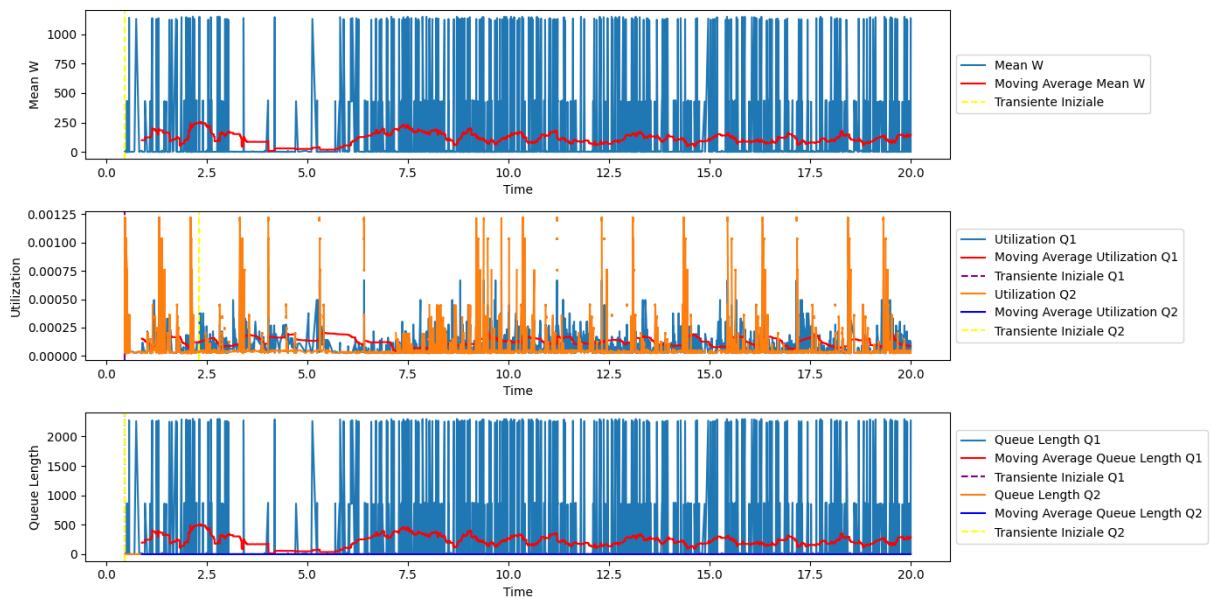


**Figura 5.6:** Panoramica delle simulazioni con  $m1 = 0.4$ .



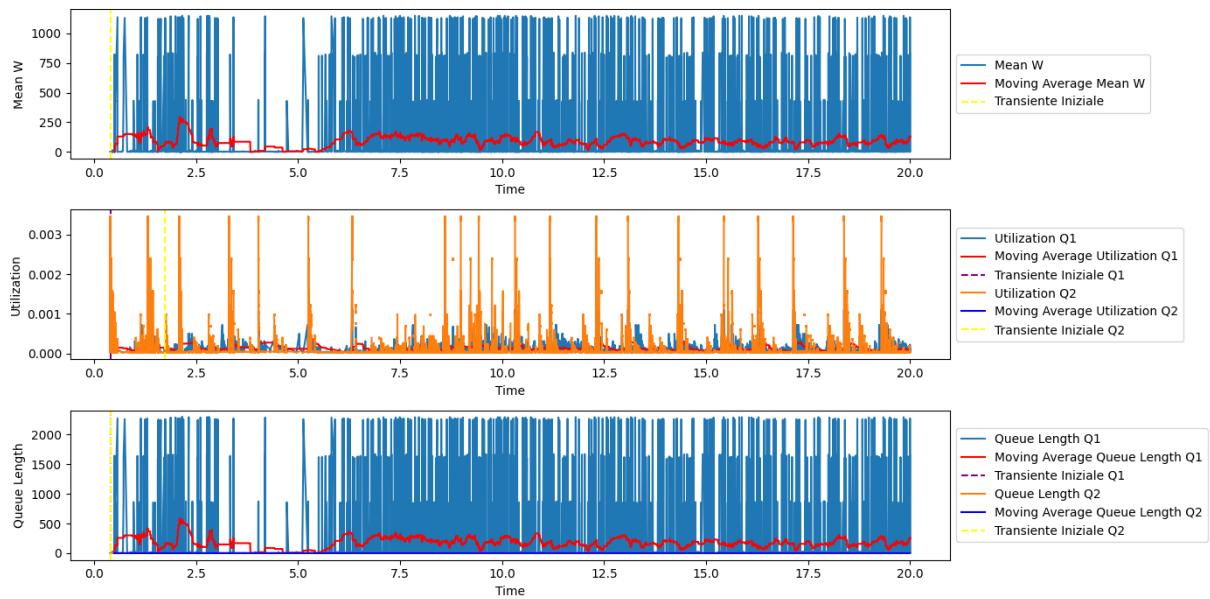
**Figura 5.7:** Panoramica delle simulazioni con  $m1 = 0.3$ .

Parametri in funzione del tempo ( $m_1=0.25$ ) - Simulazioni analizzate: 1440

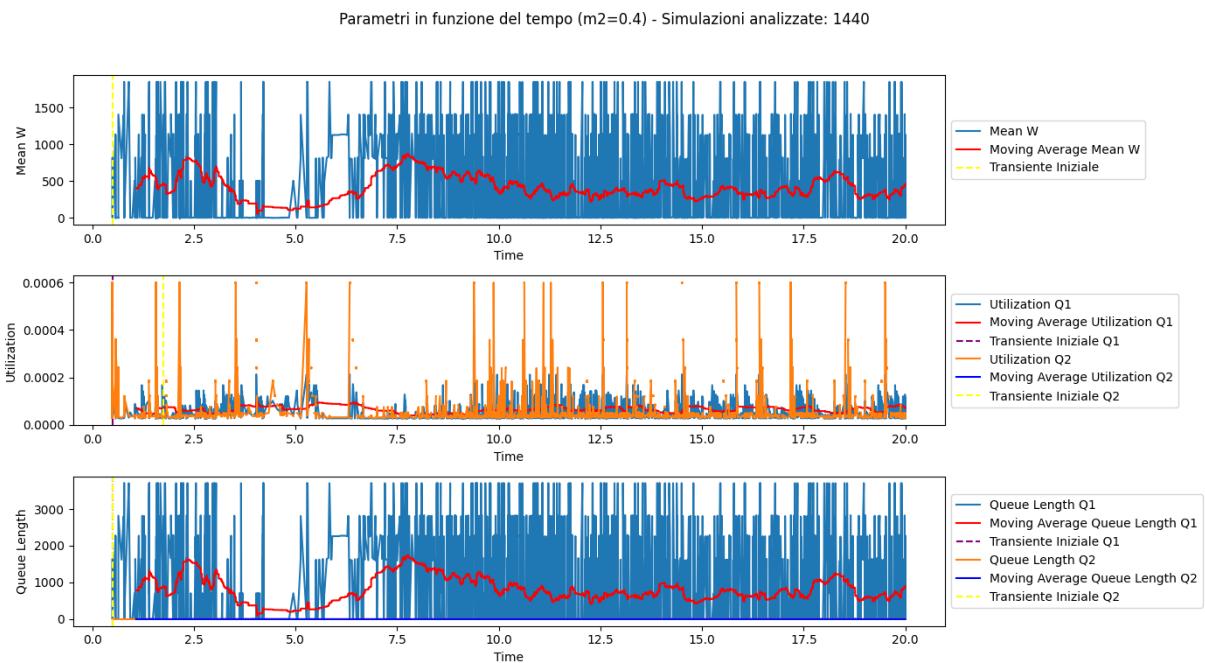


**Figura 5.8:** Panoramica delle simulazioni con  $m_1 = 0.25$ .

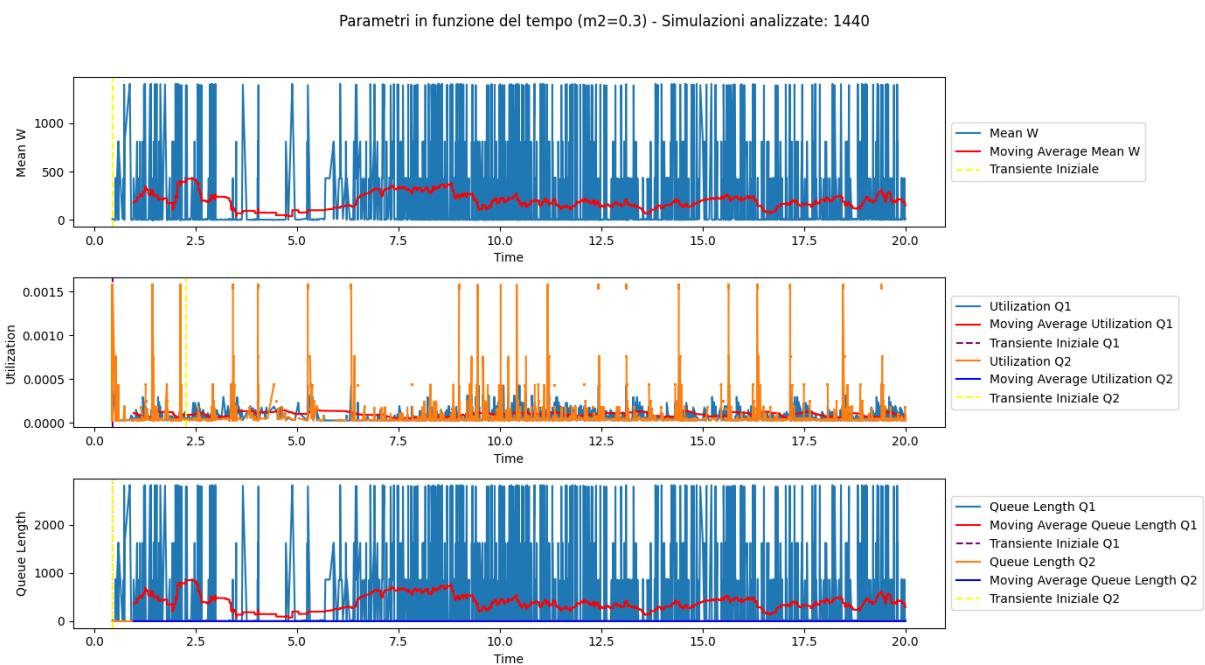
Parametri in funzione del tempo ( $m_1=0.2$ ) - Simulazioni analizzate: 2880



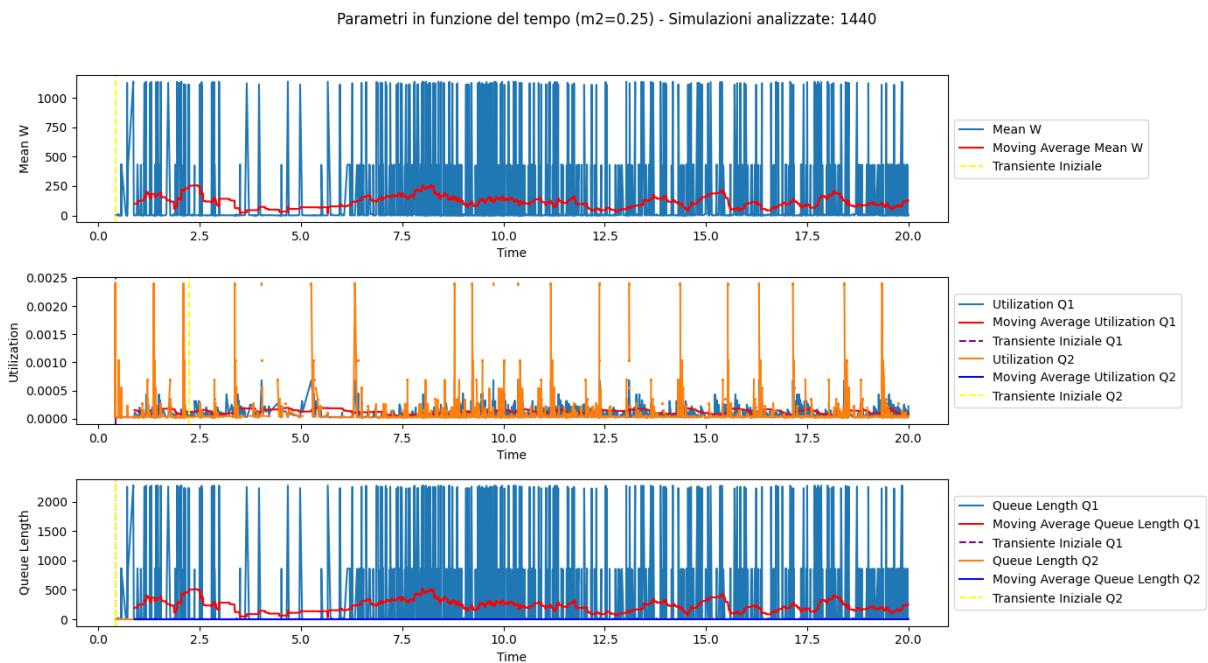
**Figura 5.9:** Panoramica delle simulazioni con  $m_1 = 0.2$ .



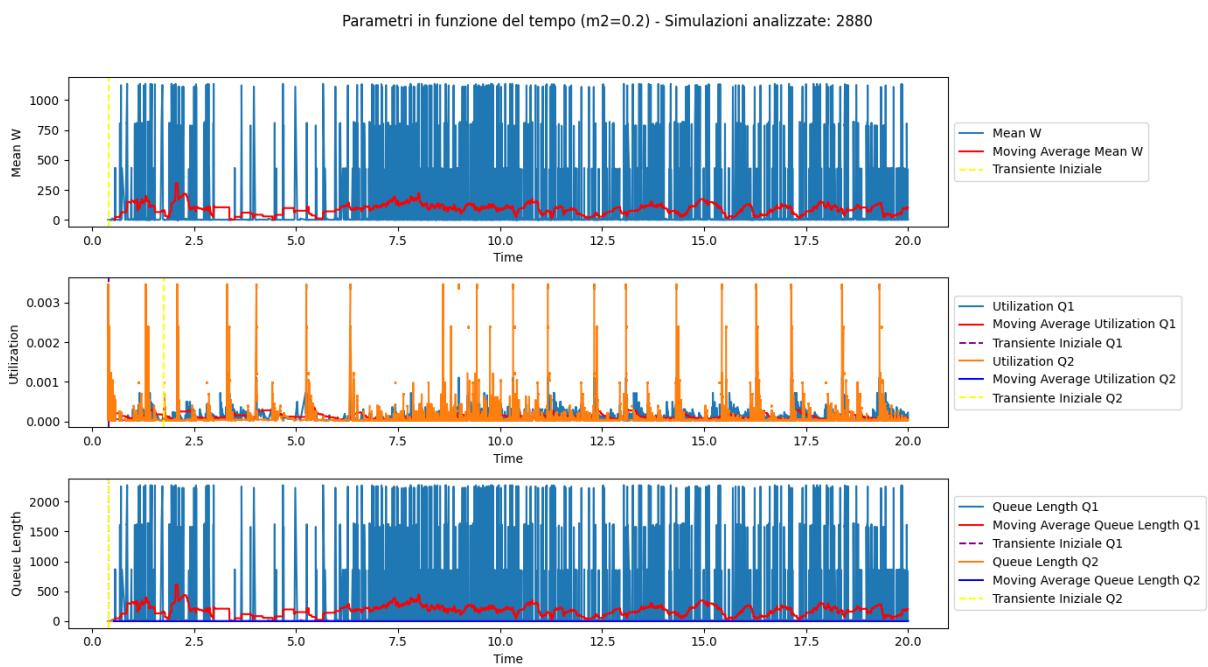
**Figura 5.10:** Panoramica delle simulazioni con  $m_2 = 0.4$ .



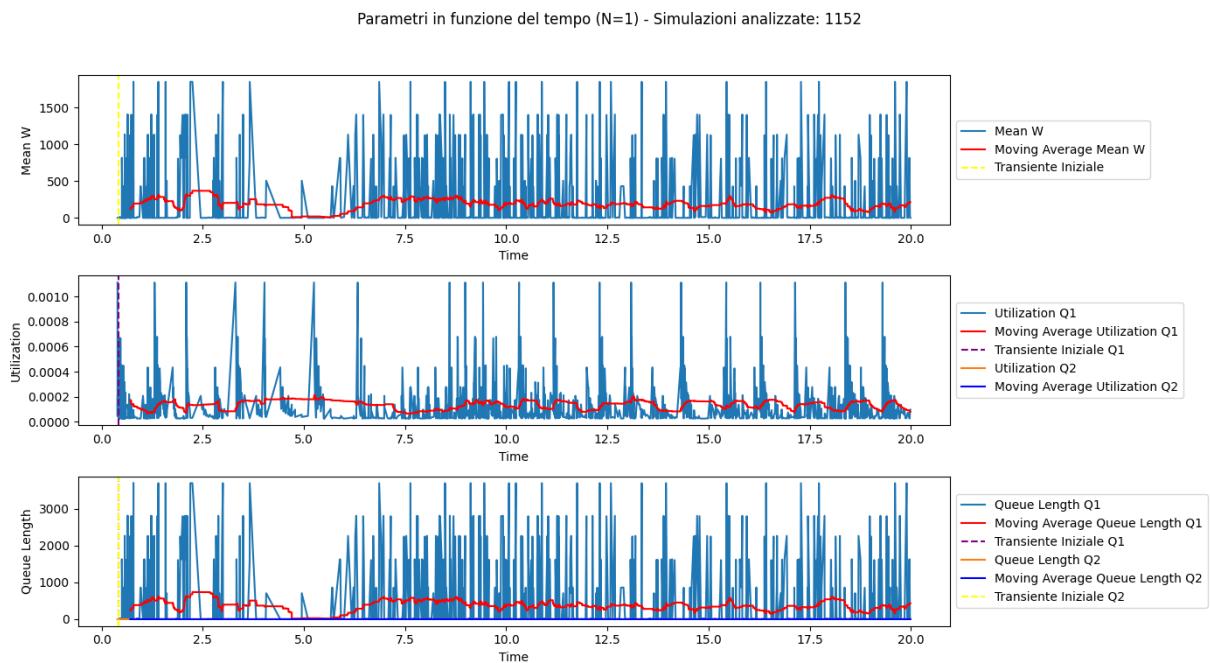
**Figura 5.11:** Panoramica delle simulazioni con  $m_2 = 0.3$ .



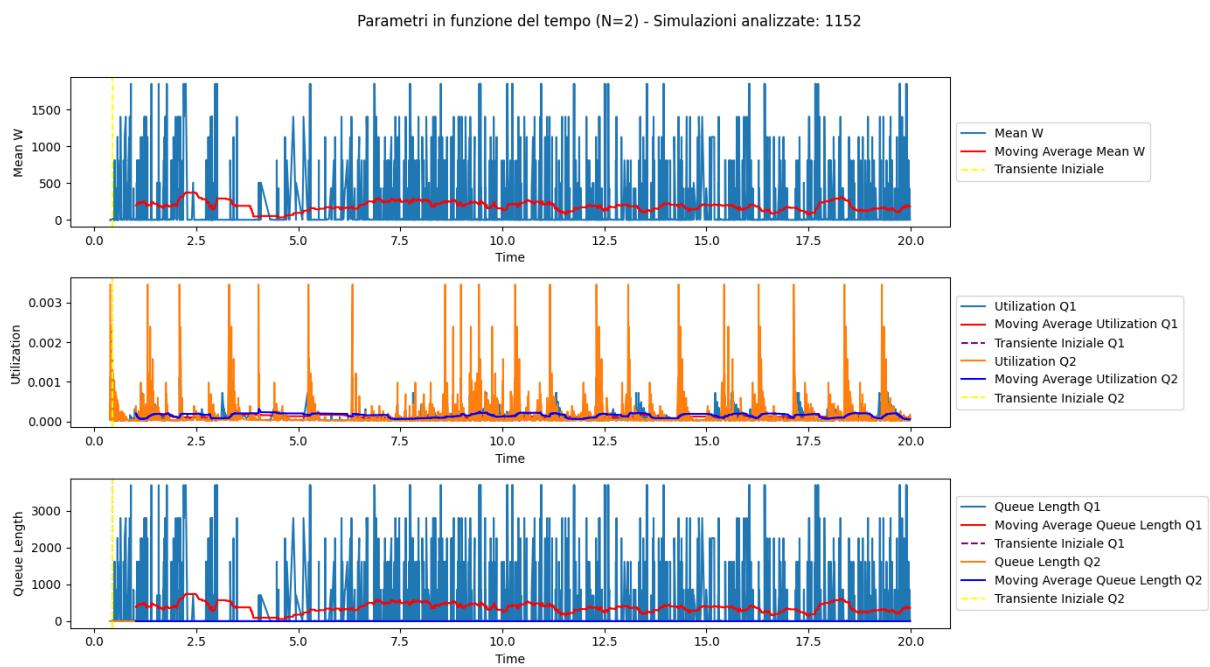
**Figura 5.12:** Panoramica delle simulazioni con  $m_2 = 0.25$ .



**Figura 5.13:** Panoramica delle simulazioni con  $m_2 = 0.2$ .

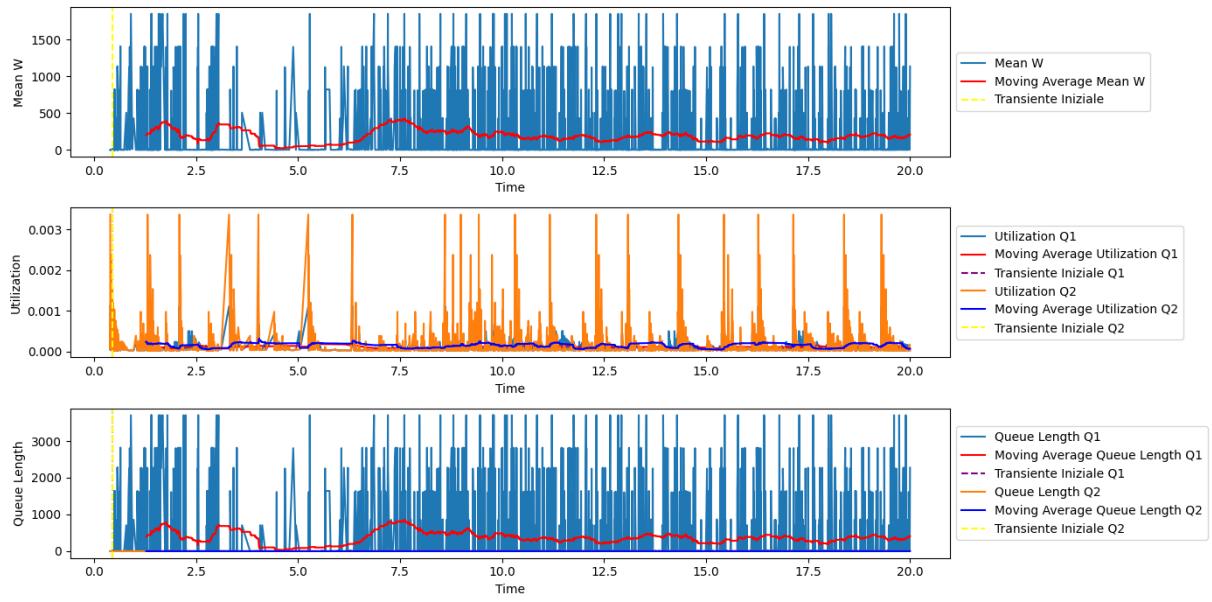


**Figura 5.14:** Panoramica delle simulazioni con  $N = 1$ .



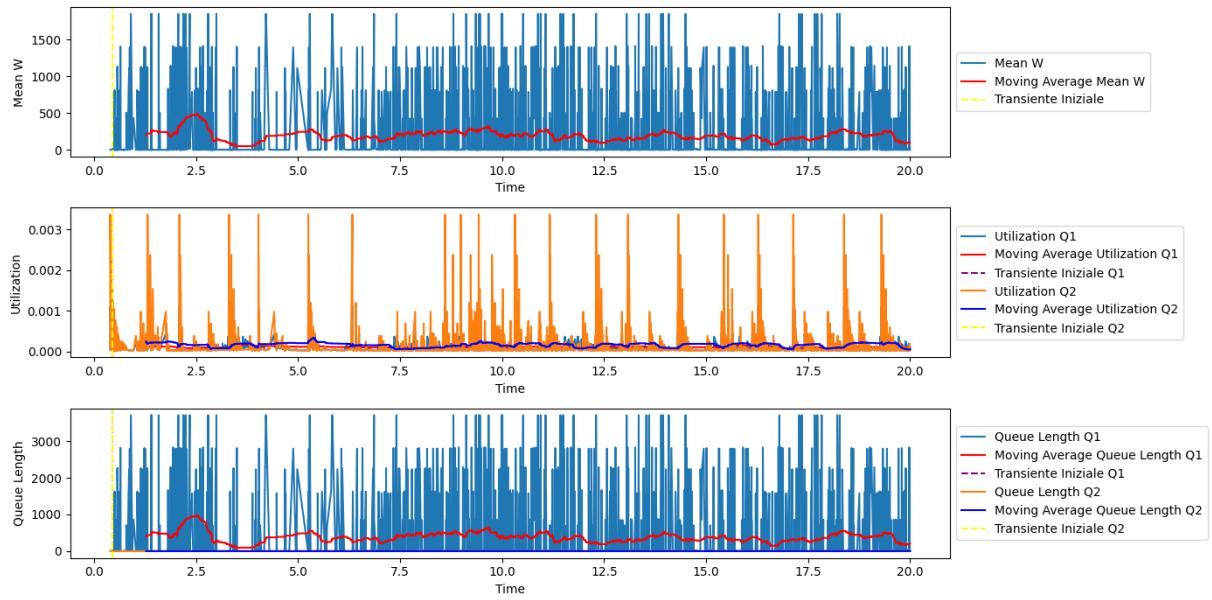
**Figura 5.15:** Panoramica delle simulazioni con  $N = 2$ .

Parametri in funzione del tempo (N=3) - Simulazioni analizzate: 1152



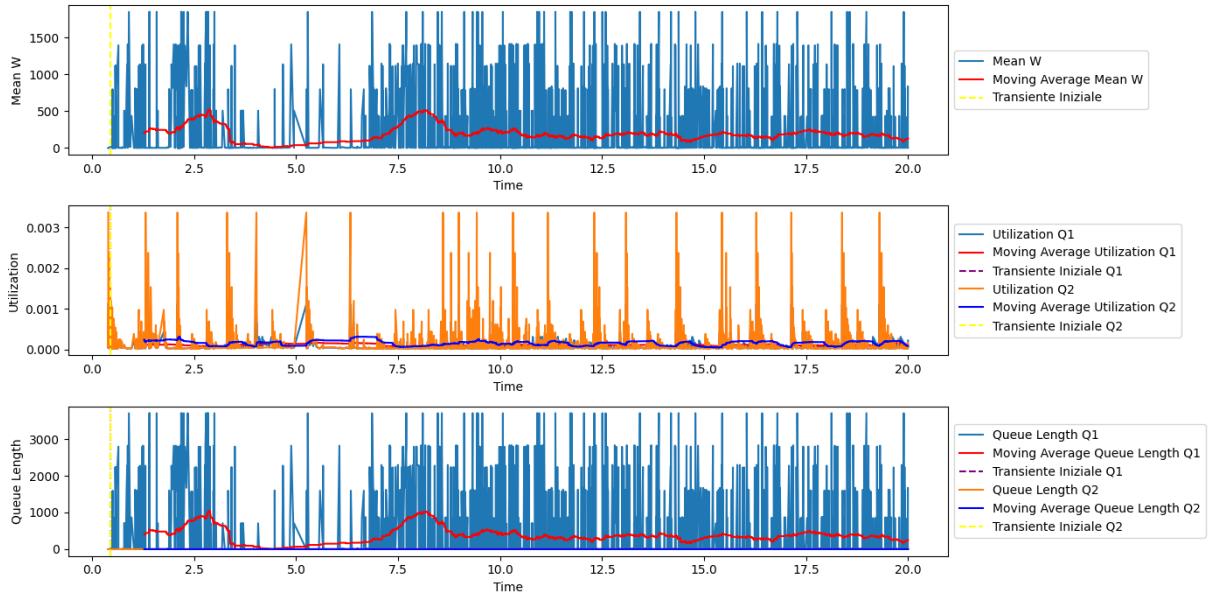
**Figura 5.16:** Panoramica delle simulazioni con  $N = 3$ .

Parametri in funzione del tempo (N=4) - Simulazioni analizzate: 1152



**Figura 5.17:** Panoramica delle simulazioni con  $N = 4$ .

Parametri in funzione del tempo (N=5) - Simulazioni analizzate: 1152



**Figura 5.18:** Panoramica delle simulazioni con  $N = 5$ .

Il transiente è stato calcolato utilizzando una finestra mobile di dimensione 1000s. Come si può vedere dai grafici i valori della simulazione rimangono stabili fin da subito quindi il transiente iniziale spesso si trova al secondo 1 rendendo triviale l'esclusione dei dati antecedenti al transiente per l'analisi dei risultati.

## 6 Misure di Prestazione

- **Tempo medio di permanenza nel sistema  $W(N)$ :** Stima puntuale e intervalli di confidenza
- **Guadagno medio  $U = V - p - (CW \times W)$ :** Stima puntuale e intervalli di confidenza
- **Tempo massimo e minimo di permanenza nel sistema:** Stima puntuale e intervalli di confidenza
- **Fattore di utilizzo dei singoli server:** Stima puntuale e intervalli di confidenza

## 7 Metodo di Simulazione

Il modello di simulazione è stato implementato utilizzando la piattaforma **Omnet++**. Le componenti sono state modificate dalla libreria **queueinglib**. Sono stati effettuati almeno 20 esperimenti per ogni configurazione del modello per un tempo simulazione di 10000s.

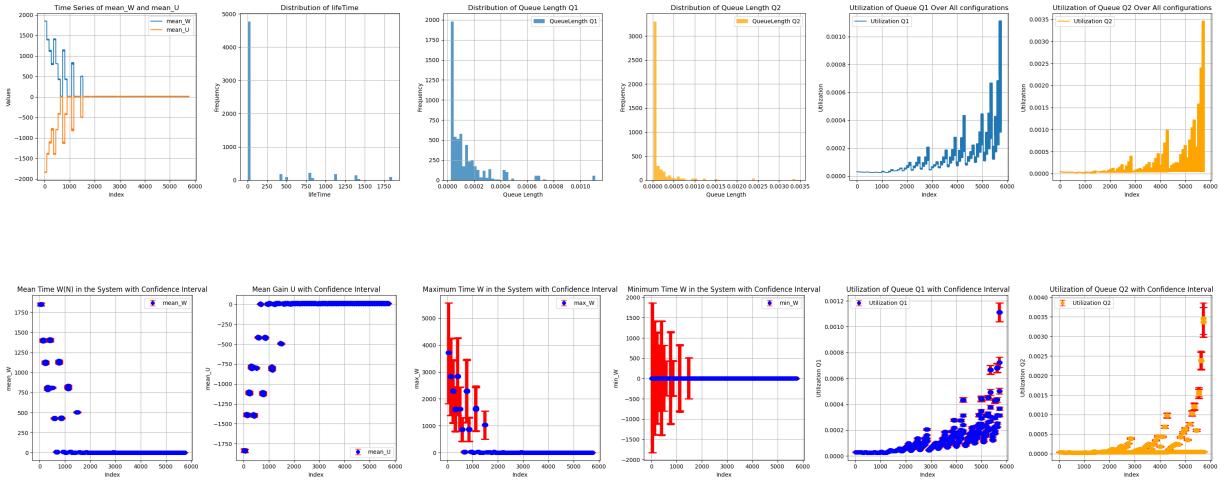
## 8 Risultati

I risultati delle simulazioni sono stati raccolti nei seguenti file:

- **LittleLaw\_general\_lambda\_results.csv**
- **LittleLaw\_results.csv**
- **results\_summary.csv**
- **results\_summary\_convalidation.csv**

## 9 Analisi dei Risultati

### 9.1 Analisi Generale



#### 1. Carico del Sistema e Performance

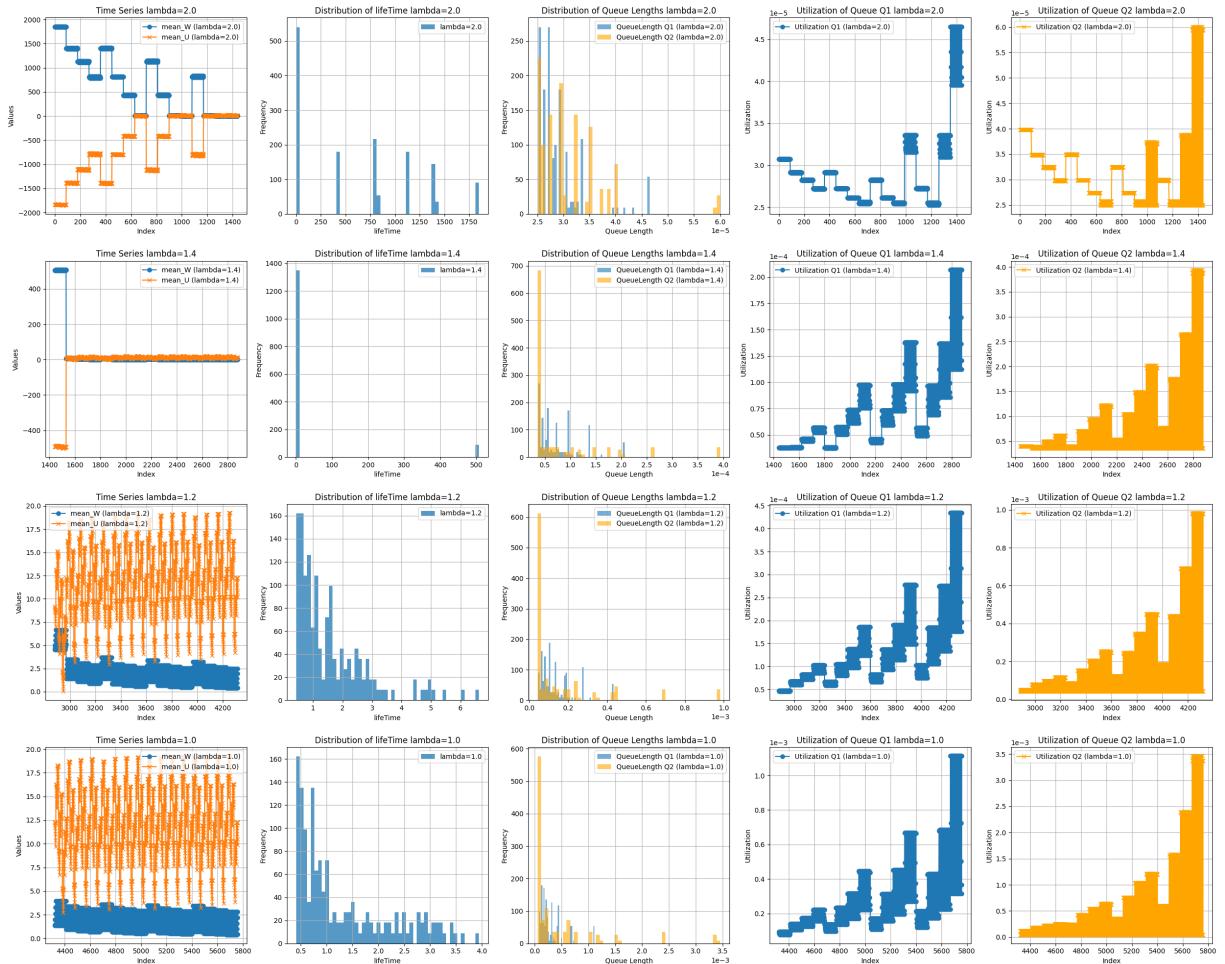
- Il tempo medio e massimo nel sistema varia notevolmente, con occasionali picchi molto alti, indicando che il sistema può essere soggetto a congestioni.

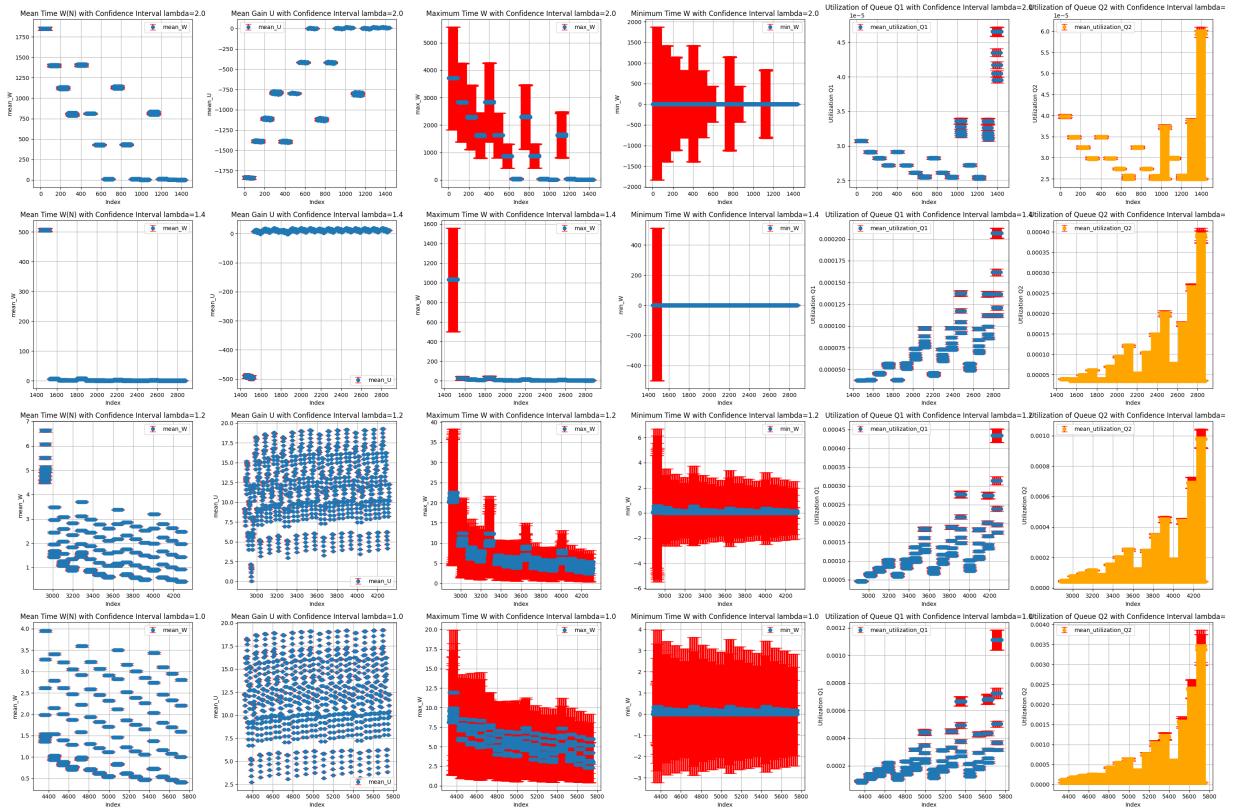
- L'utilizzo delle code è generalmente basso, ma tende ad aumentare nel tempo, specialmente per  $Q1$ , come si può vedere in Figura 5.1 suggerendo che il sistema potrebbe diventare più congestionato man mano che la simulazione procede.

## 2. Dinamiche di Comportamento Strategico

- Il comportamento strategico degli elementi nel sistema sembra influenzare significativamente le prestazioni del sistema. I picchi nel tempo massimo nel sistema e l'aumento dell'utilizzo di  $Q1$  potrebbero essere dovuti ai tempi di servizio  $m1$  ed  $m2$  generalmente troppo alti per i valori di  $\lambda$  forniti.

### 9.2 Analisi rispetto $\lambda$





## 1. Carico del Sistema e Performance

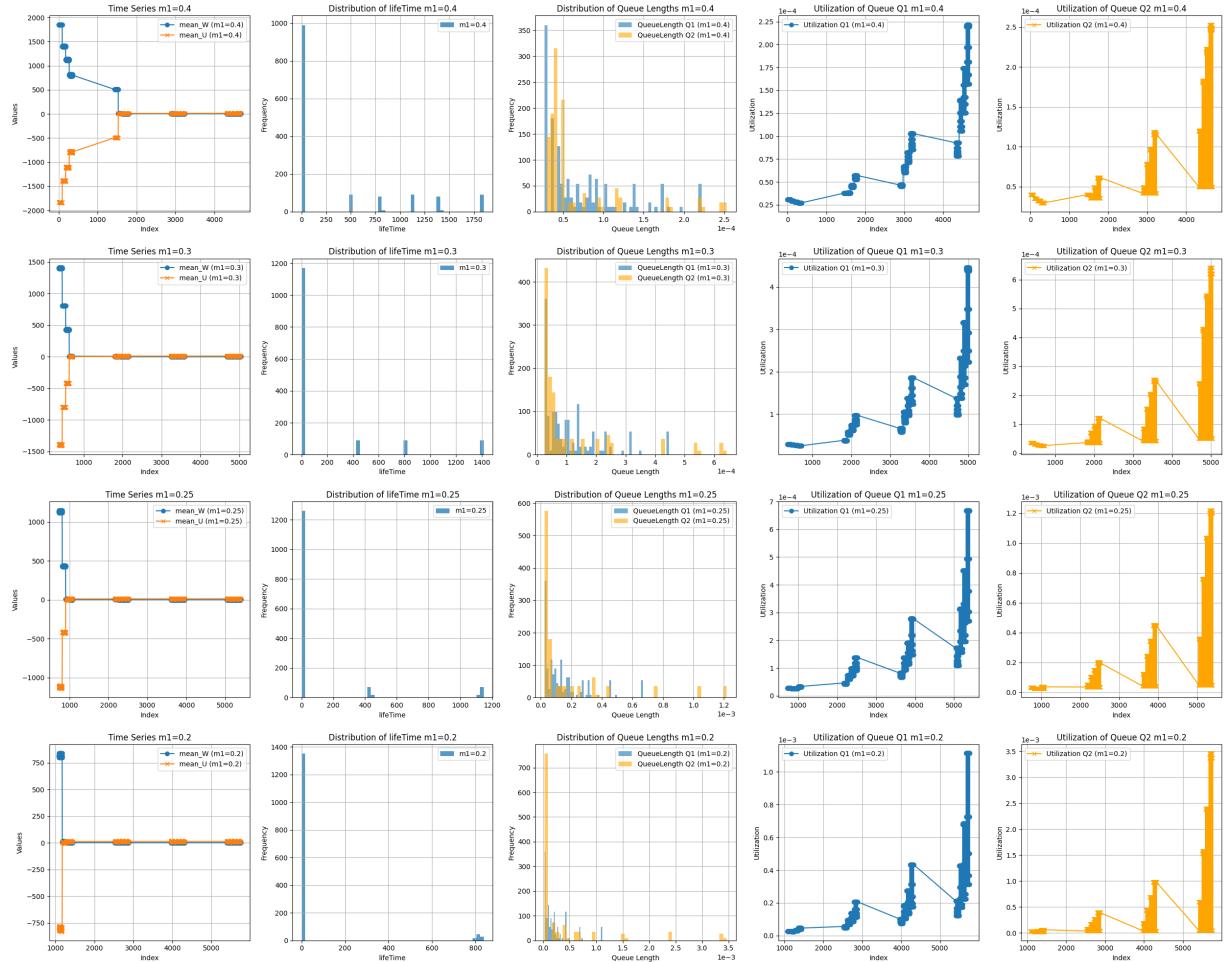
- Con diversi valori di  $\lambda$  (2.0, 1.4, 1.2, 1.0), il tempo medio e massimo nel sistema varia notevolmente. I grafici mostrano che per valori più alti di  $\lambda$ , il tempo massimo nel sistema può essere molto elevato, indicando possibili congestioni nel sistema.
- La lunghezza delle code  $Q_1$  e  $Q_2$  rimane stabile per  $\lambda$  inferiori o uguali a 1.2 come è possibile vedere in Figura 5.3 e Figura 5.4.

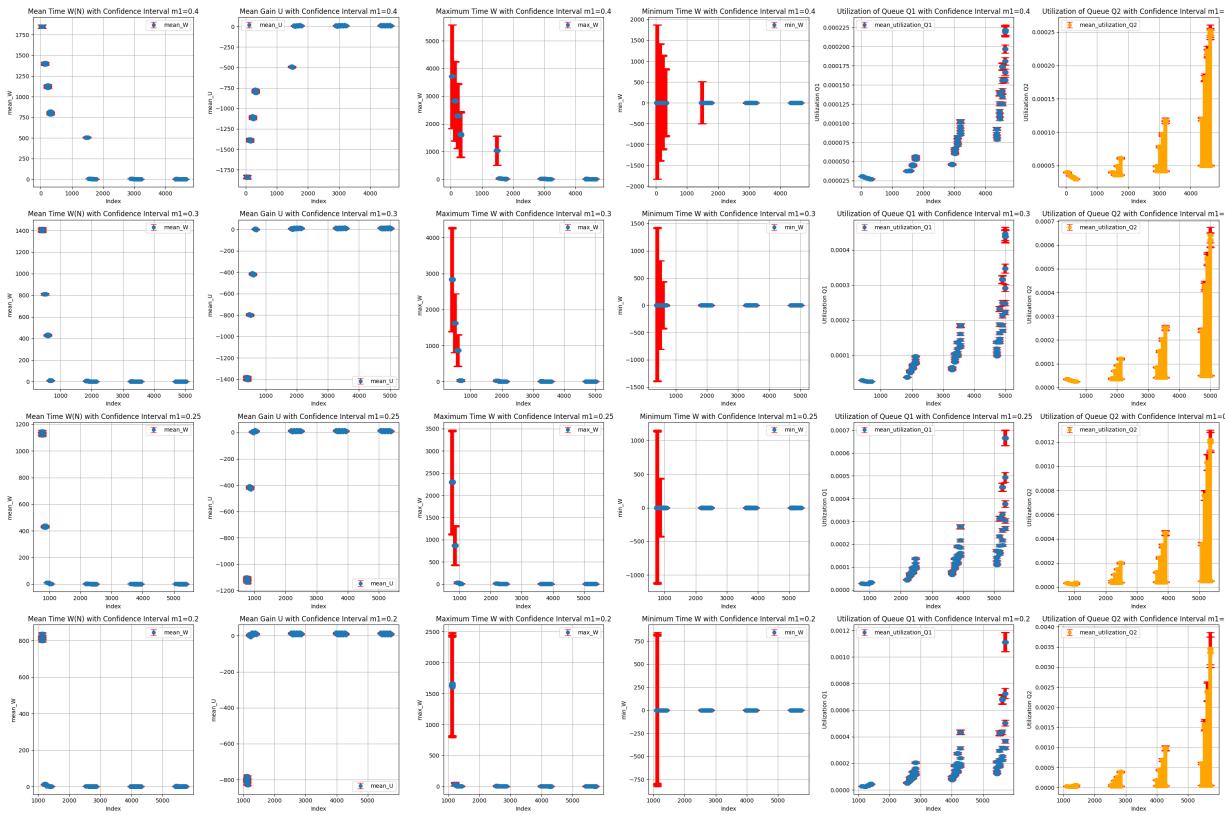
## 2. Dinamiche di Comportamento Strategico

- L'analisi delle serie temporali per  $mean\_W$  e  $mean\_U$  mostra che i valori di  $W$  Crescono al Crescere del  $\lambda$  portando a dei costi molto alti espressi con  $U$ , inoltre si può notare che con un  $\lambda$  inferiore o uguale a 1.2 i costi diventano profitti portando ad un valore di  $U$  positivo.
- I picchi nel tempo massimo nel sistema e l'aumento dell'utilizzo delle code, soprattutto per  $Q_1$ , suggeriscono che il comportamento strategico

degli elementi porta a situazioni di congestione nel sistema. Questo è particolarmente evidente con valori più alti di  $\lambda$ .

### 9.3 Analisi rispetto $m_1$





## 1. Carico del Sistema e Performance

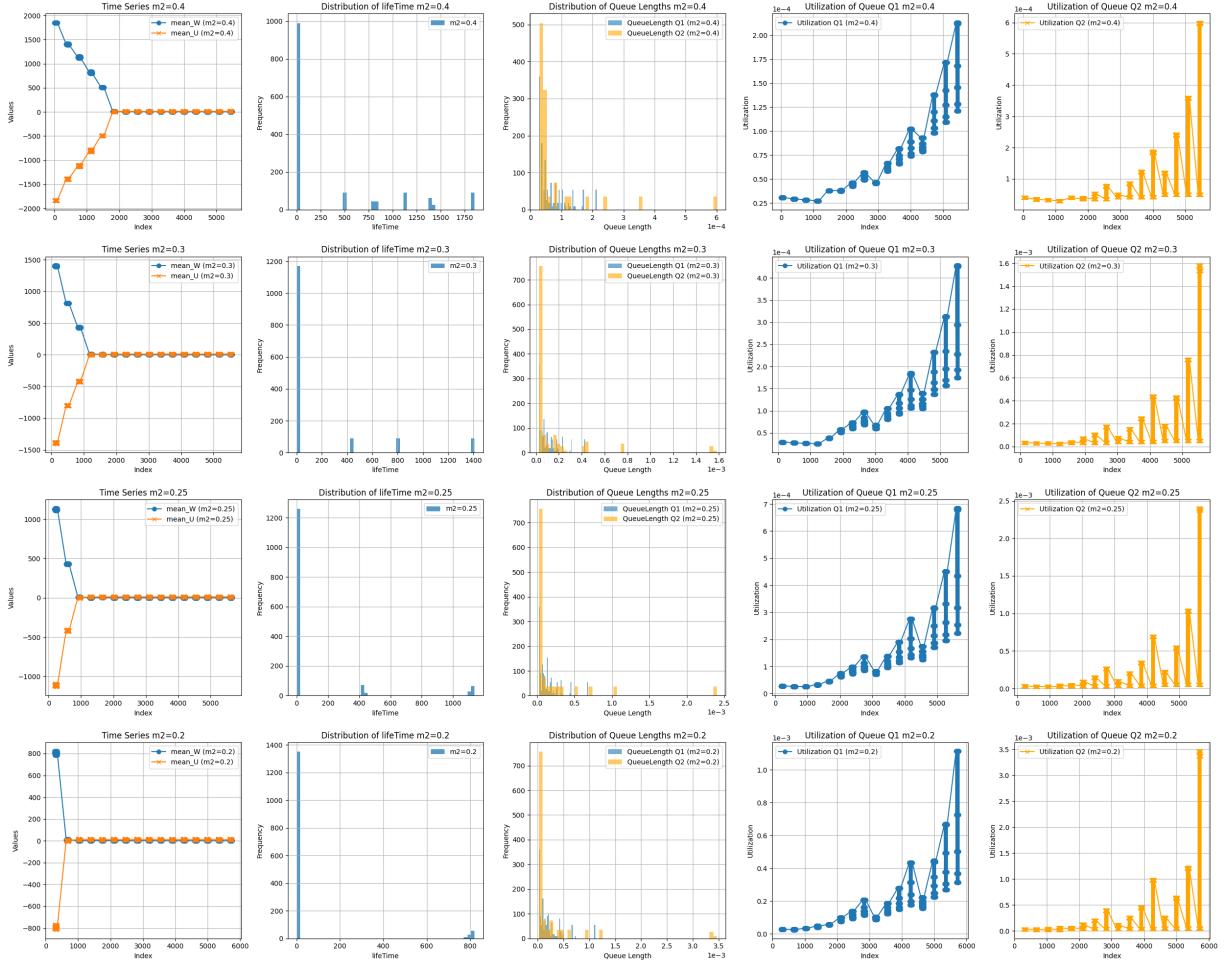
- Con diversi valori di  $m_1$  (0.4, 0.3, 0.25, 0.2), il l'utilizzo delle code nel sistema varia notevolmente. I grafici mostrano che per valori più alti di  $m_1$ , il sistema tenderà a cambiare coda meno spesso si può notare dal grafico sull'utilizzo delle code come mostrato in Figura 5.9 e Figura 5.6.
- La lunghezza delle code  $Q_1$  e  $Q_2$  diminuisce al diminuire del valore di  $m_1$  anche se non in maniera significativa.

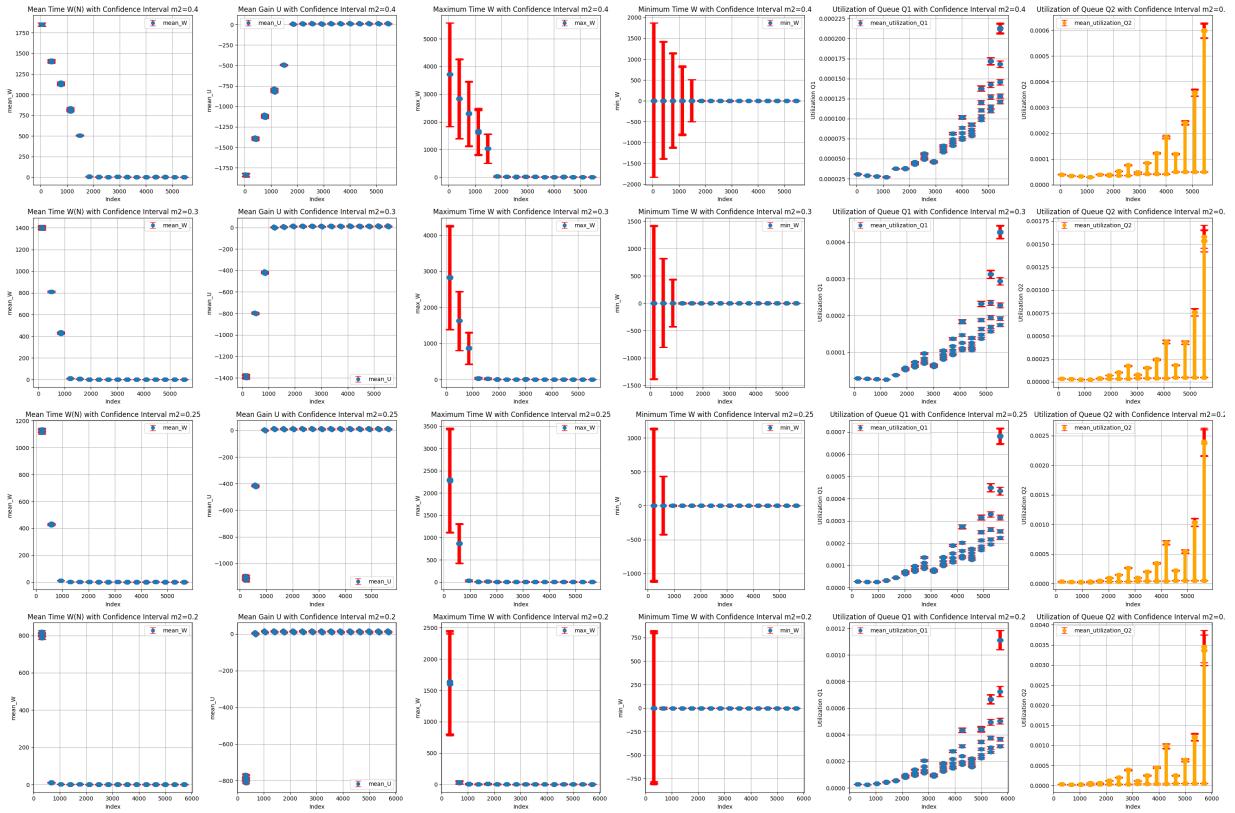
## 2. Dinamiche di Comportamento Strategico

- L'analisi delle serie temporali per  $\text{mean\_}_W$  e  $\text{mean\_}_U$  mostra che i valori di  $W$  decrescono drasticamente all'inizio e poi si stabilizzano, mentre i valori di  $U$  mostrano una diminuzione iniziale e poi rimangono costanti. Questo comportamento è osservato per tutti i valori di  $m_1$ , indicando che le strategie adottate dagli elementi influenzano in modo simile le prestazioni del sistema indipendentemente dal parametro  $m_1$ .

- I picchi nel tempo massimo nel sistema e l'aumento dell'utilizzo delle code, soprattutto per  $Q_1$ , suggeriscono che il comportamento strategico degli elementi porta a situazioni di congestione nel sistema. Questo è particolarmente evidente con valori più alti di  $m_1$ .

## 9.4 Analisi rispetto $m_2$





## 1. Carico del Sistema e Performance

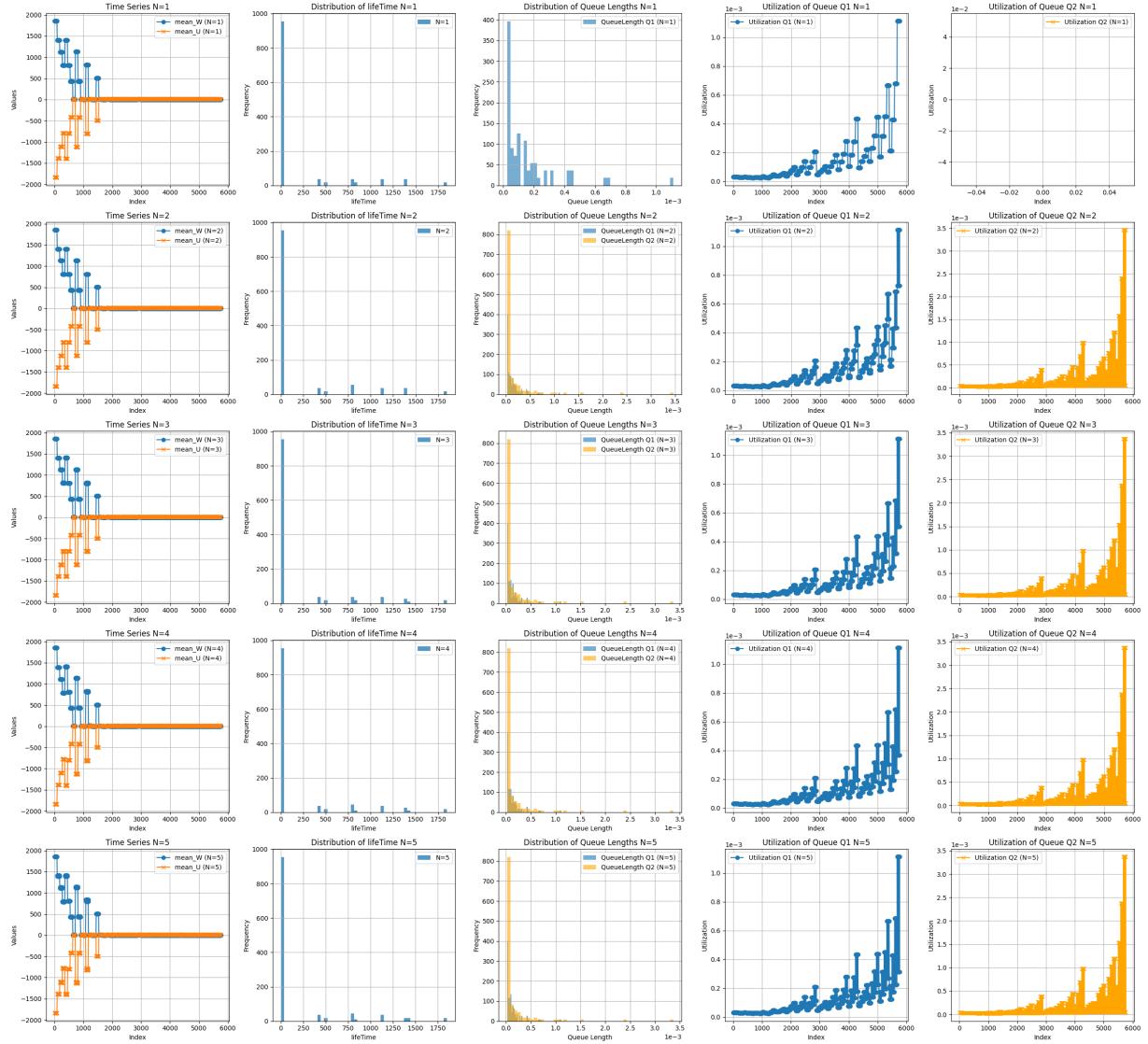
- Con diversi valori di  $m_2$  (0.4, 0.3, 0.25, 0.2), si può notare che al crescere di  $m_2$  cresce anche il tempo di vita medio  $W$  come ossrvato in Figura 5.13 e Figura 5.10.
- L'utilizzo delle code  $Q_1$  e  $Q_2$  rimane stabile per tutti i valori di  $m_2$ .

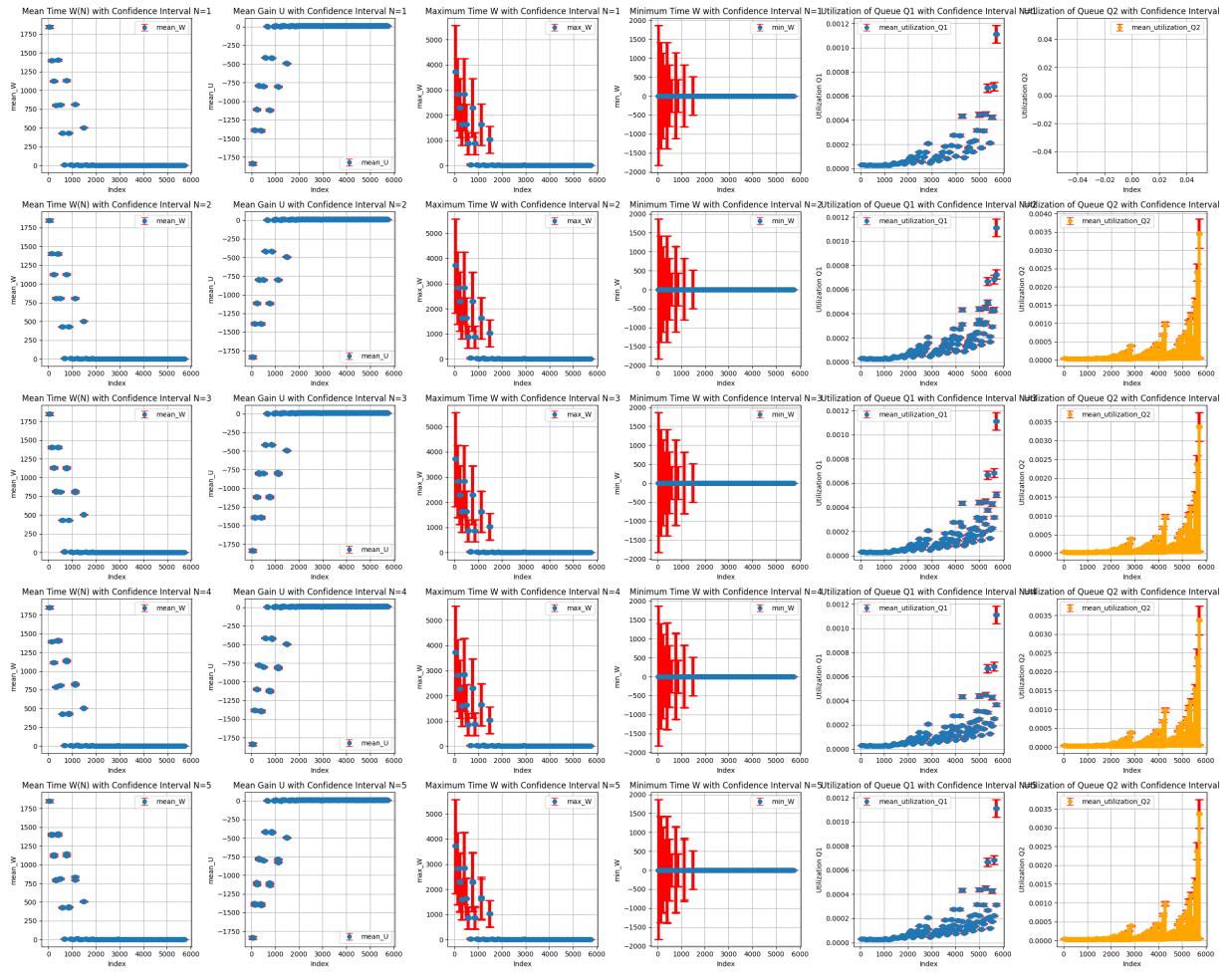
## 2. Dinamiche di Comportamento Strategico

- L'analisi delle serie temporali per  $\text{mean\_}W$  e  $\text{mean\_}U$  mostra che i valori di  $W$  decrescono drasticamente all'inizio e poi si stabilizzano, mentre i valori di  $U$  mostrano una diminuzione iniziale e poi rimangono costanti. Questo comportamento è osservato per tutti i valori di  $m_2$ , indicando che le strategie adottate dagli elementi influenzano in modo simile le prestazioni del sistema indipendentemente dal parametro  $m_2$ .
- I picchi nel tempo massimo nel sistema e l'aumento la lunghezza delle code, soprattutto per  $Q_1$ , suggeriscono che il comportamento strategico

degli elementi porta a situazioni di congestione nel sistema. Questo è osservabile per i valori più alti di  $m2$ .

## 9.5 Analisi rispetto N





## 1. Carico del Sistema e Performance

- Con diversi valori di  $N$  (1, 2, 3, 4, 5), si può osservare che non vadano ad influire in alcun modo con il comportamento delle code tranne per il caso speciale  $N = 1$  che non utilizza  $Q_2$  Figura 5.14.
- L'utilizzo delle code  $Q_1$  e  $Q_2$  rimane stabile per tutto il corso della simulazione come mostrato in Figura 5.15 e Figura 5.18.

## 2. Dinamiche di Comportamento Strategico

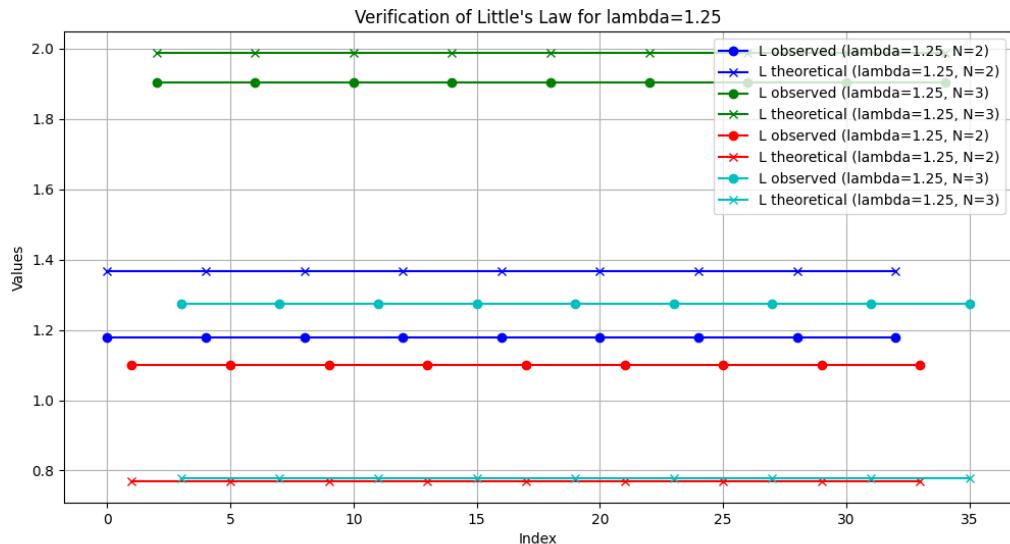
- L'analisi delle serie temporali per  $\text{mean\_}W$  e  $\text{mean\_}U$  mostra che i valori di  $W$  decrescono drasticamente all'inizio e poi si stabilizzano, mentre i valori di  $U$  mostrano una diminuzione iniziale e poi rimangono costanti. Questo comportamento è osservato per tutti i valori di  $N$ , indicando

che le strategie adottate dagli elementi influenzano in modo simile le prestazioni del sistema indipendentemente dal parametro  $N$ .

## 10 Convalida del Modello

Per la convalida del modello, verificheremo il Teorema di Little sulla configurazione **strategy 0** (Exact\_N) con un set di simulazioni **strategy 1** (N\_Limited) con  $\lambda = 1.25$ ,  $\lambda = 2$ ,  $m_1 = 0.25$ ,  $m_2 = 0.25$ ,  $n = 2$  e  $n = 3$ .

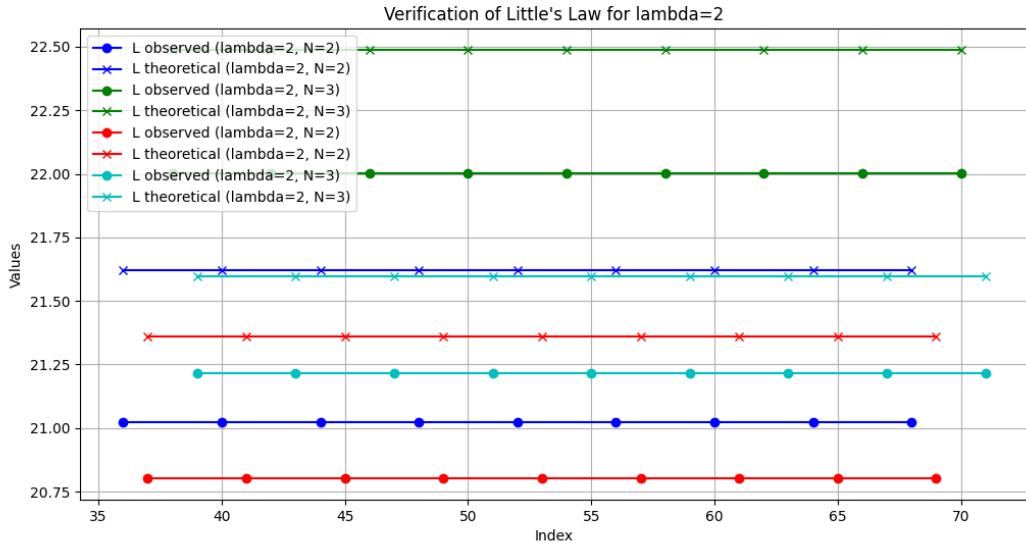
### 10.1 Verifica del Teorema di Little per $\lambda = 1.25$ con tolleranza 3%



**Figura 10.1:** Verifica del Teorema di Little per  $\lambda = 1.25$

I valori osservati di  $L$  coincidono con i valori teorici entro la tolleranza specificata, confermando la validità del modello per  $\lambda = 1.25$ .

## 10.2 Verifica del Teorema di Little per $\lambda = 2$



**Figura 10.2:** Verifica del Teorema di Little per  $\lambda = 2$

Anche in questo caso, i valori osservati di  $L$  coincidono con i valori teorici entro la tolleranza specificata, confermando la validità del modello anche per  $\lambda = 2$ .

## 11 Conclusioni

L'analisi effettuata sulla simulazione di un modello di coda tandem con server alternato ha portato a diverse osservazioni significative, sia dal punto di vista delle performance del sistema che delle dinamiche del comportamento strategico dei clienti.

### 11.1 Performance del Sistema

Le diverse configurazioni analizzate, variando i parametri di interarrivo ( $\lambda$ ), tempi di servizio ( $m_1$  e  $m_2$ ), e il numero massimo di clienti serviti prima del cambio di coda ( $N$ ), hanno mostrato che il sistema tende a comportarsi in modo differente a seconda delle politiche di servizio adottate (Exact-N e N-Limited):

- **Politica Exact-N:**

- In questa configurazione, il server segue un ciclo rigido di servizio tra le due code, il che può portare a situazioni di congestione se il flusso di arrivi è elevato.
- Il tempo massimo di permanenza nel sistema può presentare picchi significativi, soprattutto con valori più alti di  $N$ , suggerendo che un ciclo di servizio più lungo può provocare attese prolungate per alcuni clienti.

- **Politica N-Limited:**

- Questa politica consente una gestione più dinamica delle code, con il server che cambia coda quando una è vuota o quando ha servito il numero massimo di clienti  $N$ .
- Il sistema risulta generalmente più efficiente, con un utilizzo delle risorse più equilibrato e tempi di attesa più ridotti rispetto alla politica Exact-N.

## 11.2 Dinamiche di Comportamento Strategico

Le dinamiche del comportamento strategico dei clienti, che possono decidere di seguire o evitare la folla, hanno mostrato un impatto significativo sulle prestazioni del sistema:

- **Interarrivo e Tempi di Servizio:**

- I valori di  $\lambda$  e  $m_i$  influenzano direttamente il carico del sistema. Con valori più alti di  $\lambda$ , il sistema tende a congestionarsi più rapidamente, mentre tempi di servizio più lunghi ( $m_1$  e  $m_2$  maggiori) aumentano i tempi di attesa.

- **Scelta del Parametro  $N$ :**

- La scelta di  $N$  è triviale non ha una vera influenza nel sistema.

## 11.3 Validazione del Modello

La validazione del modello attraverso il Teorema di Little ha confermato la correttezza delle simulazioni. I risultati ottenuti sono in linea con i valori teorici per  $\lambda = 2$  e  $\lambda = 1.25$ .

## 11.4 Considerazioni Finali

In conclusione, lo studio ha dimostrato che la scelta delle politiche di servizio e dei parametri del sistema ha un impatto cruciale sulle prestazioni e sull'efficienza del

sistema di code tandem. Le politiche di servizio più dinamiche, come la N-Limited, offrono generalmente migliori performance rispetto a quelle statiche come la Exact-N, specialmente in condizioni di carico elevato. Le strategie dei clienti giocano un ruolo fondamentale nel determinare il comportamento del sistema, evidenziando l'importanza di modellare accuratamente tali comportamenti nelle simulazioni.

## Riferimenti bibliografici

- [DHY20] Nimrod Dvir, Refael Hassin, and Uri Yechiali. Strategic behaviour in a tandem queue with alternating server. *Queueing Systems*, 96(3-4):205–244, December 2020.