Airport Simulation

Performance Modeling Of Computer Systems And Networks 2021-2022

Lisa Trombetti – Alessandra Fanfano

1 INTRODUZIONE

Un aeroporto determina la prima e l'ultima impressione di una città da parte del viaggiatore. Un'esperienza positiva è vantaggiosa per le vendite e può influenzare le scelte di viaggio future dei passeggeri. Per questo motivo gli aeroporti di tutto il mondo continuano a prendere provvedimenti per aumentare l'attenzione al cliente, al fine di migliorare l'esperienza di quest'ultimo.

In particolare, l'area di check-in è il punto nevralgico di un aeroporto in cui vengono svolte diverse attività come i controlli di sicurezza e l'imbarco. È probabile, tuttavia, che si verifichino problemi di congestione e ritardi che vanno ad influire in maniera negativa sulla soddisfazione del cliente.

I passeggeri trascorrono effettivamente una buona parte del loro tempo negli aeroporti impegnati in tali attività o in attesa che queste vengano svolte. Diventa, quindi, sempre più evidente la necessità di andare ad individuare la migliore configurazione delle postazioni adibite al check-in, così da andare ad elaborare in modo opportuno il carico di lavoro.

In questo studio simuleremo il funzionamento di un aeroporto e cercheremo di trovare, una volta per tutte, il miglior modo per organizzare il check-in.

1.1 DESCRIZIONE DEL SISTEMA

Il sistema sotto esame è un qualsiasi aeroporto in cui è possibile cambiare la configurazione delle postazioni adibite al check-in. Le code di passeggeri indipendentemente dal tipo di configurazione vengono servite seguendo uno scheduling First-Come-First-Served (FCFS). Tale scelta è dettata dal fatto che è necessario mantenere tra i viaggiatori una situazione di "quieto vivere", la quale verrebbe sicuramente a mancare permettendo ad alcuni passeggeri di superare il resto della fila.

È inoltre possibile che un passeggero dopo aver atteso un periodo "troppo" lungo nelle code decida di andarsene ed annullare il suo volo.

1.2 OBIETTIVO

L'obiettivo di questo studio è quello di determinare la migliore organizzazione dell'area di check-in di un aeroporto in modo da **minimizzare il tempo di attesa** dei passeggeri e di conseguenza minimizzare anche il numero di clienti che andranno in futuro a favorire una diversa compagnia a causa dello scontento dovuto alle lunghe code.

2 MODELLO CONCETTUALE

Il check-in è costituito da un insieme di operazioni che devono essere effettuate prima di imbarcarsi su un volo. Nel sistema preso in esame vengono considerati diversi tipi di check-in che i passeggeri possono eseguire al loro arrivo in aeroporto, differenziati in base al numero di operazioni che devono essere effettuate e quindi al tempo necessario:

- **Check-in online**: si identifica un tipo di servizio che permette al viaggiatore di stampare la propria carta di imbarco e confermare il volo direttamente dalla propria abitazione, richiedendo solo un check-in parziale e più veloce in aeroporto
- **Check-in per voli nazionali**: si identifica un tipo di servizio che permette al viaggiatore di effettuare tutte le operazioni necessarie per il proprio imbarco direttamente al proprio arrivo in aeroporto
- **Check-in per voli internazionali**: si identifica un tipo di servizio che permette al viaggiatore di effettuare tutte le operazioni necessarie per il proprio imbarco direttamente al proprio arrivo in aeroporto, ma a differenza del precedente, viene considerato un tempo necessario maggiore dovuto ai controlli della dogana

Nel sistema oggetto di studio è possibile modificare la configurazione delle postazioni adibite al check-in. Con tale espressione ci si riferisce a una variazione del numero dei serventi e delle code di attesa nella quale i passeggeri aspettano il proprio turno per essere serviti.

Si considerano le seguenti configurazioni delle postazioni:

- **Coda singola**: tutti i passeggeri si accodano in un'unica coda, senza considerare distinzioni in base al tipo di operazioni da effettuare. Il primo servente a liberarsi andrà a servire il primo elemento nella coda

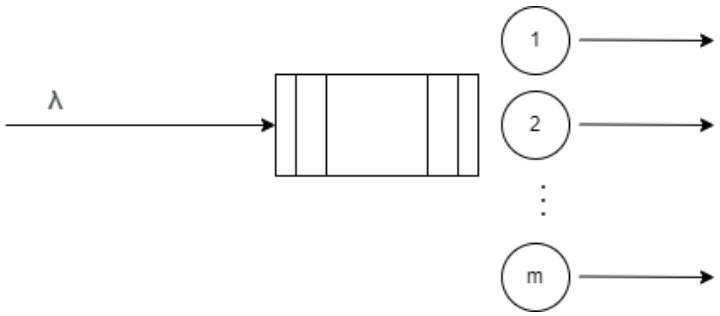


Figura 1 - Configurazione a coda singola

- **Coda multipla**: ad ogni servente presente nel sistema viene associata una coda nella quale i passeggeri si possono accodare, scegliendo quella con il minor numero di utenti in attesa

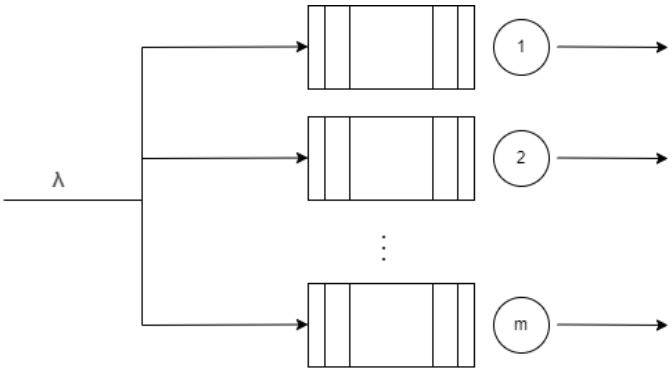


Figura 2 - Configurazione a coda multipla

- **Coda sita**: i passeggeri vengono suddivisi nelle diverse code in base alle operazioni che devono eseguire: check-in online, check-in nazionale, check-in internazionale

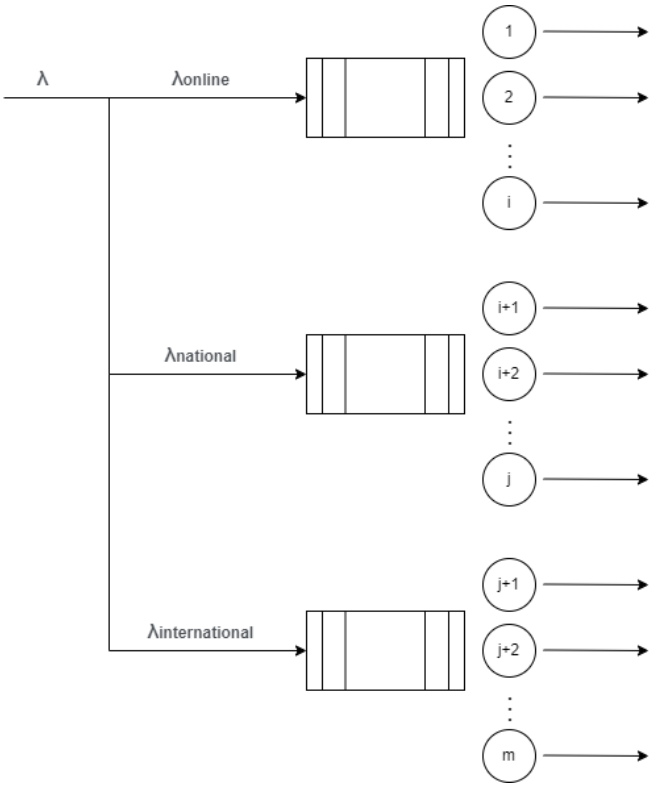


Figura 3 - Configurazione a coda sita

2.1 EVOLUZIONE DEL MODELLO

Considerando la situazione che il nostro Paese e che il mondo in generale sta vivendo da ormai due anni, è opportuno andare ad analizzare come il Covid-19 possa influire sulle code in un aeroporto. Infatti, sono ormai moltissimi gli aeroporti che mettono a disposizione dei passeggeri la possibilità di effettuare test rapidi di controllo prima dell'imbarco.

Si è quindi specializzato il sistema in esame andando ad effettuare una distinzione tra i passeggeri già in possesso di un green pass valido e coloro che devono effettuare il test. Per quest'ultimo caso, si evidenzia anche la possibilità di avere un certo tasso di "perdita" dovuto a coloro che risultano essere positivi e che quindi non avranno possibilità di partire.

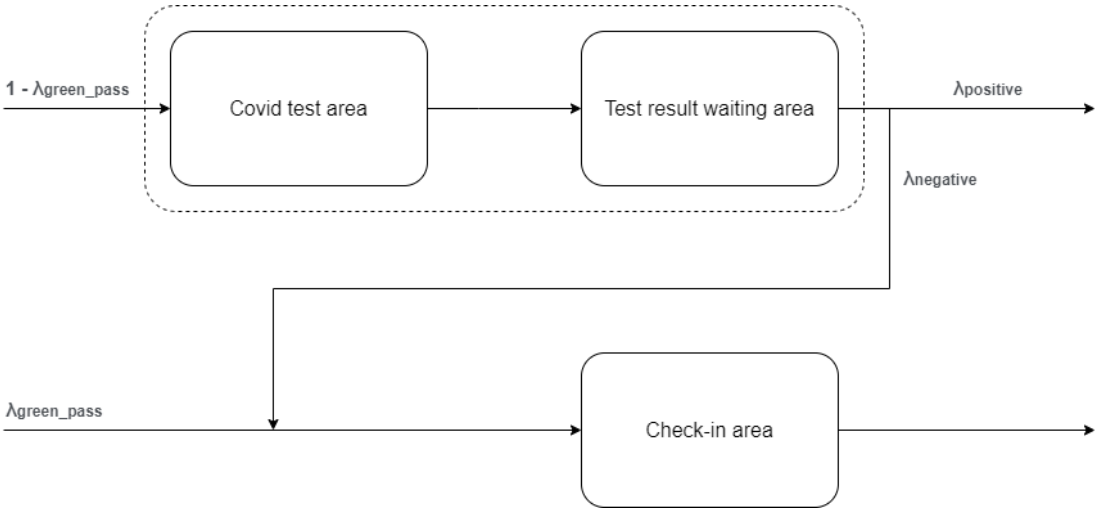


Figura 4 - Aggiunta area covid al modello

Come illustrato in Figura 4, per gestire coloro che devono effettuare il tampone antigenico vengono ad essere identificate due zone fondamentali: una in cui viene effettivamente eseguito il test e un'altra in cui aspettare i risultati rispettando tutte le norme di sicurezza e prevenzione. Per quanto riguarda quest'ultima area di attesa, le modalità con cui l'aeroporto può scegliere di realizzare e gestire tale zona non sono state considerate nel modello.

Le configurazioni considerate per la coda di test sono le seguenti:

- Coda singola
- Coda multipla

Come è possibile notare da quest'ultima lista, per l'area di test non è stata inserita la configurazione sita, in quanto la suddivisione in base al tipo di check-in non ha nessuna relazione con la probabilità di avere o non avere un green pass o con il tasso di positività.

2.2 PARAMETRI

Di seguito vengono riportati tutti i parametri che sono stati considerati nei vari modelli e la relativa descrizione.

- $\boldsymbol{\lambda}$ - tasso di arrivo dei passeggeri
- $\boldsymbol{\mu_{online}}$ - tasso di servizio dei passeggeri con check-in online
- $\boldsymbol{\mu_{national}}$ - tasso di servizio dei passeggeri per voli nazionali
- $\boldsymbol{\mu_{international}}$ - tasso di servizio dei passeggeri per voli internazionali
- $\boldsymbol{\mu_{online}}$ - tasso di servizio dei passeggeri con check-in online
- $\boldsymbol{p_{online}}$ - probabilità che il passeggero abbia effettuato un check-in online
- $\boldsymbol{p_{national}}$ - probabilità che il passeggero debba effettuare un check-in per un volo nazionale
- $\boldsymbol{p_{international}}$ - probabilità che il passeggero debba effettuare un check-in per un volo internazionale
- **max_wait** - attesa massima che un passeggero attenderà nella coda prima di ritirarsi ed annullare il suo volo

- $\boldsymbol{\mu_{test}}$ - tasso di servizio dei tamponi rapidi
- $\boldsymbol{p_{greenpass}}$ - probabilità che il passeggero sia provvisto di green pass
- $\boldsymbol{p_{positive}}$ - probabilità che il passeggero risulti positivo al test rapido
- **test_wait** - tempo che il passeggero deve attendere per avere il risultato del test rapido

3 MODELLO DELLE SPECIFICHE

Grazie alle informazioni riportate dall'U.S. Customs and Border Protection (CBP) [1] sono stati reperiti diversi dati riguardanti l'aeroporto di Los Angeles per utilizzarlo come esempio, in quanto risulta essere abbastanza grande ed un punto di partenza/arrivo sia per voli nazionali che internazionali. Inoltre, sono stati presi in considerazione i dati relativi all'anno 2019, in quanto, durante il periodo della pandemia, i voli risultano essere notevolmente ridotti e si vuole fornire un'analisi che risulti effettivamente utile in tempi di normalità, sperando di lasciare presto questo periodo alle nostre spalle.

3.1 ANALISI DEL DATASET

Il dataset reperito ha fornito diverse informazioni (Figura 1), ma di queste solo alcune sono state prese effettivamente in considerazione per l'analisi. Delle colonne presenti si è deciso di non considerare la

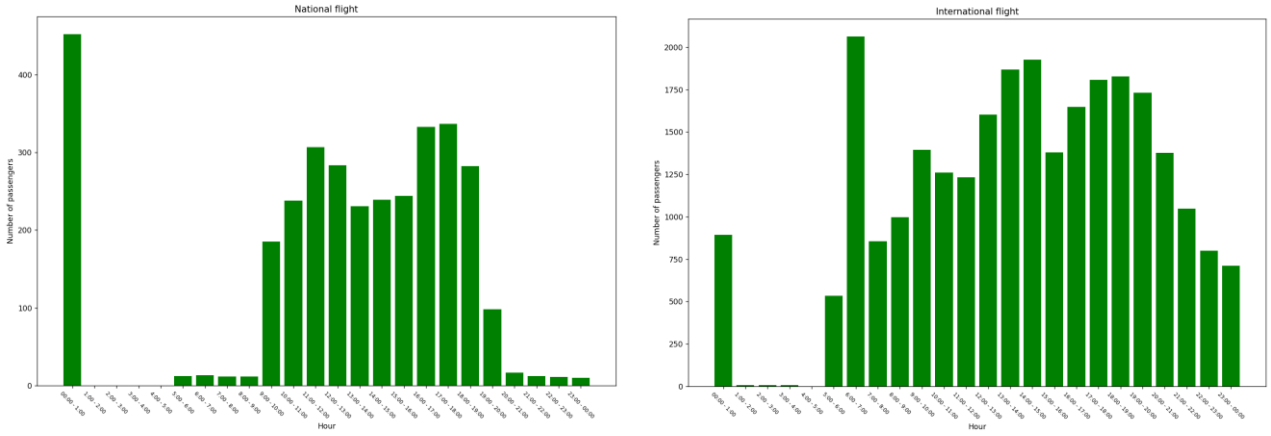
distinzione tra cittadini americani e non americani riguardante l'average wait time e il max wait time, inoltre non è stata neanche presa in esame la colonna "Excluded" che indica il numero di passeggeri che sono stati esclusi dall'analisi, perché i valori riportati in questa colonna risultano essere abbastanza esigui rispetto al valore totale dei passeggeri.

Airport	Terminal	Date	Hour	U.S. Citizen		Non U.S. Citizen		All Wait Times		Number Of Passengers Time Interval										
				Average Wait Time	Max Wait Time	Average Wait Time	Max Wait Time	Average Wait Time	Max Wait Time	0-15	16-30	31-45	46-60	61-90	91-120	120 plus	Excluded	Total	Flights	Booths
LAX	Satellite 2	01/01/2019	0600 - 0700	7	9	22	22	9	22	10	2	0	0	0	0	0	0	12	1	0
LAX	Satellite 2	01/01/2019	0700 - 0800	7	11	0	0	7	11	25	0	0	0	0	0	0	1	26	1	0
LAX	Satellite 2	01/01/2019	0900 - 1000	11	13	0	0	11	13	8	0	0	0	0	0	0	1	9	1	0
LAX	Satellite 2	01/01/2019	1000 - 1100	15	43	18	48	16	48	424	219	114	4	0	0	0	22	783	5	6
LAX	Satellite 2	01/01/2019	1100 - 1200	13	42	17	42	15	42	146	94	12	0	0	0	0	5	257	2	6
LAX	Satellite 2	01/01/2019	1200 - 1300	11	45	16	44	12	45	339	138	22	0	0	0	0	19	518	4	6
LAX	Satellite 2	01/01/2019	1300 - 1400	7	22	15	24	9	24	161	27	0	0	0	0	0	1	189	1	6
LAX	Satellite 2	01/01/2019	1600 - 1700	7	14	11	16	8	16	149	6	0	0	0	0	0	4	159	1	6
LAX	Satellite 2	01/01/2019	1700 - 1800	10	35	12	36	10	36	467	53	12	0	0	0	0	17	549	3	7
LAX	Satellite 2	01/01/2019	1800 - 1900	7	18	9	14	7	18	121	2	0	0	0	0	0	4	127	2	3

Figura 5 - Dataset aeroporto di Los Angeles

L'analisi effettuata si è concentrata principalmente sui dati riguardanti i voli Nazionali e Internazionali, considerando come voli internazionali solamente quelli in partenza dal Tom Bradley International Terminal e come voli nazionali tutti gli altri. La decisione di effettuare questa sola distinzione nasce dal fatto che il dataset non fornisce informazioni riguardo i diversi tipi di check-in che è possibile effettuare in aeroporto e anche dal fatto che reperire tali dati risulta essere abbastanza complicato.

Un'analisi preliminare ha permesso di andare a individuare il numero di passeggeri che possono susseguirsi in un'area di check-in nell'arco di una giornata. Nelle figure riportate di seguito è possibile notare come per entrambe le tipologie di voli, nazionale e internazionale, per tutta la giornata è presente un alto numero di passeggeri tranne che per la fascia oraria tra 1:00 e le 5:00. Per semplicità nella simulazione sono stata prese in considerazione solamente le ore di maggior afflusso, poiché è di interesse lo scenario in cui sono presenti delle file al desk del check-in.



3.2 MODELLI ANALITICI

Di seguito vengono riportati i modelli analitici a cui corrispondono i diversi tipi di configurazione definiti nel modello concettuale del sistema:

- **Coda singola**: è una coda singola con serventi multipli; quindi, corrisponde ad una *multi server queue (M/M/m)*

$$\rho_i = \rho_{glob} = \frac{\lambda}{m\mu} \qquad P_Q = \frac{(m\rho)^m}{m!(1-\rho)} p(0)$$

$$E(T_S)_{Erlang} = \frac{P_Q E(S)}{1-\rho} + E(S_i) \qquad E(T_Q)_{Erlang} = \frac{P_Q E(S)}{1-\rho}$$

$$E(S_i) = 1/\mu \qquad E(S) = 1/m\mu = E(S_i)/m$$

- **Coda multipla**: è un sistema costituito da m serventi ognuno con la propria coda di attesa; quindi, abbiamo m diverse single server queues (M/M/1)

$$\rho = \lambda / \mu \qquad E(T_S) = \frac{\rho E(S)}{1-\rho} + E(S) = \frac{E(S)}{1-\rho} \qquad E(T_Q) = \frac{\rho E(S)}{1-\rho}$$

- **Coda sita**: 3 diverse code in cui i passeggeri vengono suddivisi in base al tipo di operazione che deve essere eseguita. È quindi possibile rappresentare questa configurazione come un 3 diverse multi server queues (M/M/m). Le formule di riferimento sono le stesse della coda singola.

3.3 Ulteriori considerazioni

Per quanto riguarda il tasso di interarrivo si è deciso di utilizzare una distribuzione esponenziale, così come per il tasso di servizio degli operatori del check-in e dell'area di test. Tale decisione è stata presa dopo aver analizzato alcuni articoli scientifici, [2] e [3], nei quali si sostiene che tale scelta sia appropriata e che possa simulare in modo opportuno lo scenario di nostro interesse.

È tuttavia doveroso ripotare che nell'articolo "Airport Passenger Arrival Process: Estimation of Earliness Arrival Functions" [4] gli autori ritengono che la distribuzione di Weibull sia più rappresentativa dell'interarrivo dei passeggeri ad un aeroporto rispetto all'esponenziale. Questo potrebbe essere considerato come un valido punto di partenza per eventuali miglioramenti futuri.

Infine, per quanto riguarda i diversi parametri che descrivono tale sistema, si è cercato di trovare dei valori da usare nelle simulazioni che fossero il più vicini possibile a quelli reali. Per questo motivo, per la percentuale di passeggeri provvisti di green pass è stato usato il valore trovato sul sito del governo italiano [5] (Totale ciclo vaccinale), mentre per la probabilità di risultare positivi al test si è considerato il tasso di positività riportato dal Lab del Sole 24 ore [6].

4 Modello Computazionale

Il modello computazionale è stato sviluppato tramite una Next-Event Simulation, dove la modifica dello stato del sistema è caratterizzata da un avanzamento del tempo in cui ogni volta si passa direttamente dall'istante di arrivo del passeggero precedente a quello del successivo.

4.1 Generazione dei Passeggeri

All'inizio di ogni simulazione viene generata una lista di passeggeri ordinata in base al tempo di arrivo nell'area di check-in. Per gestire al meglio i dati qui generati è stata utilizzata una struttura **passenger** che tiene traccia delle informazioni relative al singolo passeggero, quali, ad esempio, il tipo di check-in che deve effettuare, il possesso del green pass ed i valori relativi ai diversi tempi che risultano essere interessanti ed utili per la gestione dei servizi (check-in e tampone antigenico).

```c
// Struct which identifies a passenger
typedef struct passenger {

    int type;           // Passenger type (ONLINE, NATIONAL, INTERNATIONAL)
    int greenpass;      // Whether the passenger has a green pass or not
    int withdrawal;     // Whether the passenger has withdrawn

    int arrival;        // Arrival time (minutes)
    int begin;          // Begin of service time (minutes)
    int service;        // Service time (minutes)
    int departure;      // Departure time (minutes)

    int test_arrival;   // Covid test arrival time (minutes)
    int test_begin;     // Covid test begin of service time (minutes)
    int test_service;   // Covid test service time (minutes)
    int test_departure; // Covid test departure time (minutes)

    struct passenger *next;

} passenger;
```

Figura 6 - Struttura passenger

Partendo da zero, ad ogni generazione di un nuovo passeggero si va ad incrementare il tempo corrente utilizzando un valore generato con una distribuzione esponenziale che rappresenta il tempo di interarrivo. La generazione viene interrotta non appena il tempo corrente raggiunge una soglia che può essere definita dall'utente che avvia la simulazione.

While *curr_time < stop_time* ***do***
  curr_time += ***Exponential***
  create_passenger *new_pass*
  add_passenger *new_pass to pass_list in arrival order*
end while

Per quanto riguarda la distribuzione esponenziale utilizzata è stata realizzata, come visto a lezione, utilizzando la funzione *Random* della libreria *rngs.c*.

*-m*log(1.0 -Random())*

4.2 GESTIONE DEGLI ARRIVI

Ogni configurazione gestisce gli arrivi all'interno del sistema in maniera differente. Tuttavia, è possibile individuare due procedure fondamentali su cui si basa tale processo: *find_officer* e *find_queue*. La prima viene utilizzata nella configurazione a coda singola per individuare quale fra i serventi disponibili si farà carico di servire il nuovo passeggero.

procedure *find_officer*
num = -1
dep = 1000000 // High value
For *i:=0 to off_num* ***do***
  If *dep > last dep from officer i* ***then***
    dep = last dep officer i
    num = i

end
return *num*

La seconda viene utilizzata nella configurazione a coda multipla per inserire un passeggero appena arrivato nella coda che in quel momento risulta essere la “migliore”, ovvero quella con il minor numero di utenti.

```
procedure find_queue
num = -1
lowest_pass_num = 1000000 // High value
For i:=0 to queue_num do
 If queue i is empty then
  return i
 pass_num = 0
 For e:= elem in queue i do:
  If e inside queue at arrival then
   pass_num++
 end
 If pass_num < lowest_pass_num then
  lowest_pass_num = pass_num
  num = i
end
return num
```

4.3 Simulazione delle Configurazioni

Tutte le configurazioni vengono simulate scorrendo una lista collegata di strutture passenger ordinate per il tempo di arrivo nell’area check-in. Così facendo è possibile andare a servire i passeggeri e verificare se i tempi di attesa hanno portato a delle perdite oppure no.

4.3.1 Coda singola

Nella simulazione della coda singola per ogni passeggero viene scelto un operatore basandosi sul tempo di partenza dell’ultimo utente che quest’ultimo ha servito. Una volta trovato il servente che per primo si libererà, viene gestito il nuovo passeggero calcolando il tempo di inizio e di fine del servizio presso quell’operatore. Nel caso in cui il tempo di attesa nella coda diventi troppo lungo, il passeggero può lasciare la fila producendo un aumento del numero degli abbandoni.
Le medie dei valori del tempo di attesa, servizio e risposta dei singoli passeggeri calcolate alla fine della simulazione vengono poi inserite in una struttura result.

```
procedure simulate_single_queue

num_pass, num_withdrawal, mwait, mresponse, mservice = 0

For p:=pass in list do
 choose officer and update passenger values

 If passenger waits to long then
  num_withdrawal++
 Else
  num_pass++
  update mean values
end

add results to the result structure
```

```c
// Structure which keeps the test results
typedef struct result{

    char *type;             // Check-in area configuration type (SINGLE, MULTI, SITA)
    int num_officers;       // Total number of officers
    double mwait;           // Average passenger wait
    double mresponse;       // Average passenger response time
    double mservice;        // Average service time
    double withdrawal;      // Percentage of passenger who withdrew from the queue

    // SITA configuration only
    int num_off_small;      // Number of officers for small jobs
    double mwait_small;     // Average passenger wait for small jobs
    double mresponse_small; // Average passenger response time for small jobs
    double mservice_small;  // Average passenger service time for small jobs
    double withdrawal_small;// Percentage of passenger who withdrew from the small jobs queue

    int num_off_med;        // Number of officers for medium jobs
    double mwait_med;       // Average passenger wait for medium jobs
    double mresponse_med;   // Average passenger response time for medium jobs
    double mservice_med;    // Average passenger service time for medium jobs
    double withdrawal_med;  // Percentage of passenger who withdrew from the medium jobs queue

    int num_off_big;        // Number of officers for big jobs
    double mwait_big;       // Average passenger wait for big jobs
    double mresponse_big;   // Average passenger response time for big jobs
    double mservice_big;    // Average passenger service time for big jobs
    double withdrawal_big;  // Percentage of passenger who withdrew from the big jobs queue

    // Covid test area
    char *test_type;        // Test area configuration type (SINGLE, MULTI, NONE)
    int test_officers;      // Total number of covid test officers
    double mwait_test;      // Average passenger covid test wait
    double mresponse_test;  // Average passenger covid test response time
    double mservice_test;   // Average passenger covid test service time

} result;
```

Figura 7 - Struttura result

4.3.2 Coda multipla

Nel caso della coda multipla non è necessario effettuare una scelta del servente in quanto ve ne è uno per ogni coda. Il passeggero può tuttavia scegliere la coda in qui accodarsi, che probabilmente sarà quella con il numero minore di utenti al momento del suo arrivo.

procedure *simulate_multi_queue*

num_pass, num_withdrawal, mwait, mresponse, mservice = 0

For *p:=pass in list* ***do***
 find_queue *and update passenger values*
 If *passenger wait to long* ***then***
  num_withdrawal++
 Else
  create new element
  add_element *to the queue*
end

For *i:=0 to off_num* ***do***
 free_elements in queue i
end

add results to the result structure

Per effettuare questo calcolo è stata definita una piccola struttura *element* che contiene il tempo di inizio di servizio per un passeggero. Andando quindi a realizzare una lista collegata di element per ogni servente, ogni volta che vi è un nuovo arrivo si può scegliere la coda giusta andando calcolare il numero di passeggeri ancora in attesa in quel momento.

```
typedef struct element{
    int begin;
    struct element *next;
} element;
```

Figura 8 - Struttura element

4.3.3 Coda sita

La gestione della configurazione con coda sita è sostanzialmente identica a quella della coda singola, con l'unica differenza che le code a disposizione sono tre ed i passeggeri si distribuiscono su di esse in base al tipo di check-in richiesto. Perciò, ogni volta che vi è un nuovo arrivo è sufficiente andare a classificare il tipo di passeggero e poi simulare una coda singola con il numero opportuno di serventi.

procedure *simulate_sita_queue*

For *p:=pass in list* ***do***
 If *passenger is online* ***then***
  handle online as single configuration
 Else if *passenger is national* ***then***
  handle national as single configuration
 Else
  handle international as single configuration
end

add results to the result structure

4.3.4 Covid test area

L'aggiunta dell'area adibita all'esecuzione di tamponi antigenici rapidi va a rallentare tutti quei passeggeri che, in assenza della pandemia, si sarebbero semplicemente presentati nell'area di check-in per svolgere i dovuti controlli. Quindi, il tempo di arrivo al desk di un passeggero sprovvisto di green pass deve essere calcolato simulando anche le attese nella coda per il tampone.

Come specificato nelle sezioni precedenti, anche per questa nuova area vengono considerate come configurazioni possibili la coda singola e la coda multipla. Perciò, la simulazione del test covid è sostanzialmente identica a quella dell'area di check-in, con le uniche differenze che:

- Se un passeggero risulta positivo deve essere considerato come una perdita
- Se un passeggero risulta negativo non esce fuori dal sistema ma va considerato nella coda di check-in

While *there are still passengers in list* ***do***

 If *passenger has green pass* ***then***
  handle_covid_test / find_covid_test_queue

If** passenger is positive **then
num_withdrawal++
Else
continue with check-in

end while

4.4 DEMO

Per sfruttare le funzioni appena descritte è stata realizzata una semplice demo presente nel file **demo.c** della cartella lib. Per usarla è necessario andare ad impostare i parametri che si intende utilizzare all'interno del file **config.h**. Tali valori vengono inseriti all'interno di una struttura *config* che viene passata come parametro della funzione che si occupa della generazione dei passeggeri.

Una volta avviata, la demo va a simulare i diversi tipi di configurazione dell'area di check-in e dell'area di test salvando i risultati ottenuti in un file csv di cui è possibile specificare il nome.

```
// Struct which keeps all the configuration parameters
typedef struct config {

    double lambda;           // Passenger arrival rate
    double test_mu;          // Covid test officers service rate
    double online_mu;        // Check-in online service rate
    double national_mu;      // Check-in national service rate
    double international_mu; // Check-in international service rate

    double greenpass_p;      // Probability of having the green_pass
    double online_p;         // Probability of online check-in
    double national_p;       // Probability of national check-in
    double international_p;  // Probability of international check-in

} config;
```

Figura 9 - Struttura config

Tutte le informazioni su come compilare ed eseguire la simulazione sono presenti nel file **README.md** nella root del progetto.

5 VERIFICA

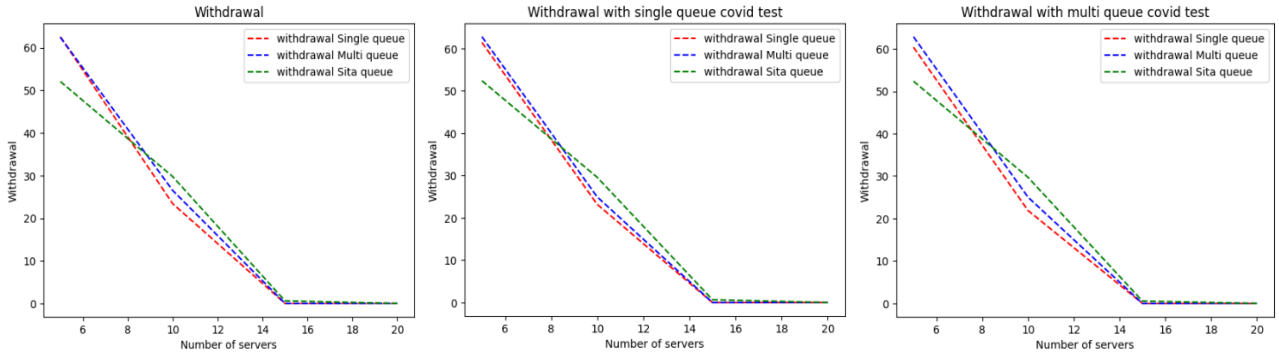
Per realizzare la verifica del modello si è avviata la simulazione con configurazioni differenti, modificando il numero di serventi presenti all'interno del sistema e differenziando come questi vengono suddivisi. In particolare, si è prestata molta attenzione al verificare che il tempo di attesa e di servizio non differissero troppo dai valori ottenuti mediante l'analisi del dataset.

6 VALIDAZIONE

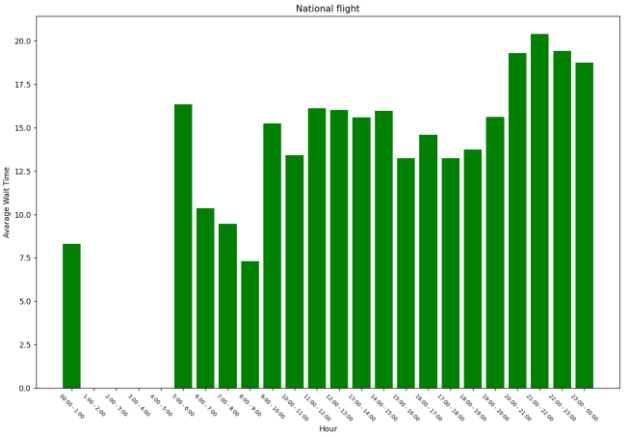
Per andare a validare la simulazione rispetto al modello, si è prima di tutto effettuato un controllo sui valori restituiti del tempo di risposta, attesa e servizio. Infatti, ciò che ci si aspetta è che i valori medi rispettino la seguente formula:

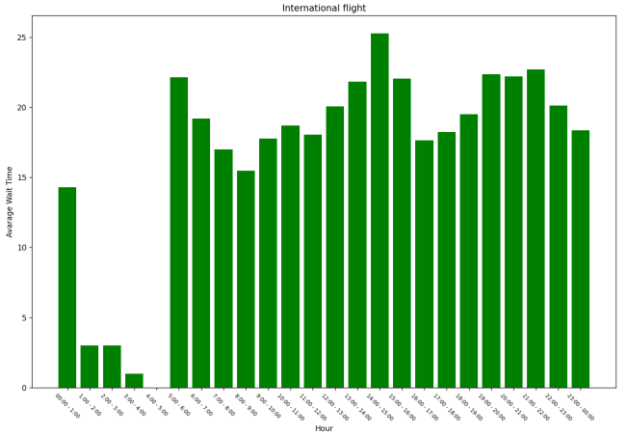
$$E(T_s) = E(T_Q) + E(S)$$

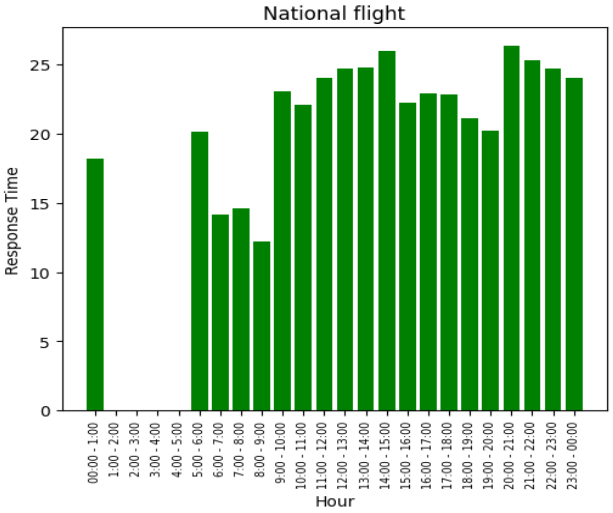
Inoltre, si è deciso di controllare che il numero degli abbandoni ed i tempi spesi nelle code, sia del check-in che dell'aria del test rapido, diminuissero al crescere del numero degli operatori.

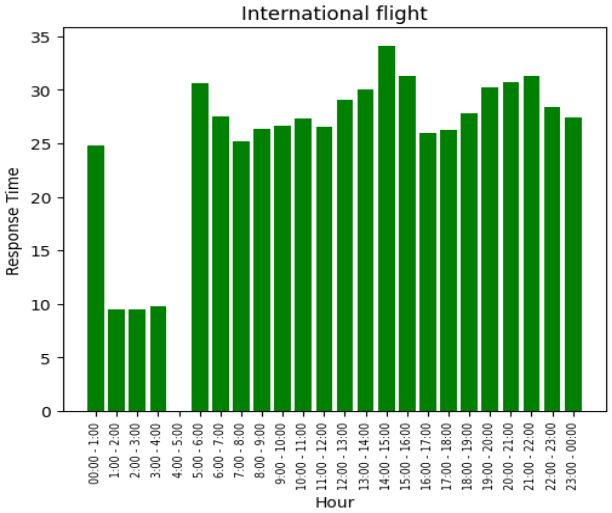


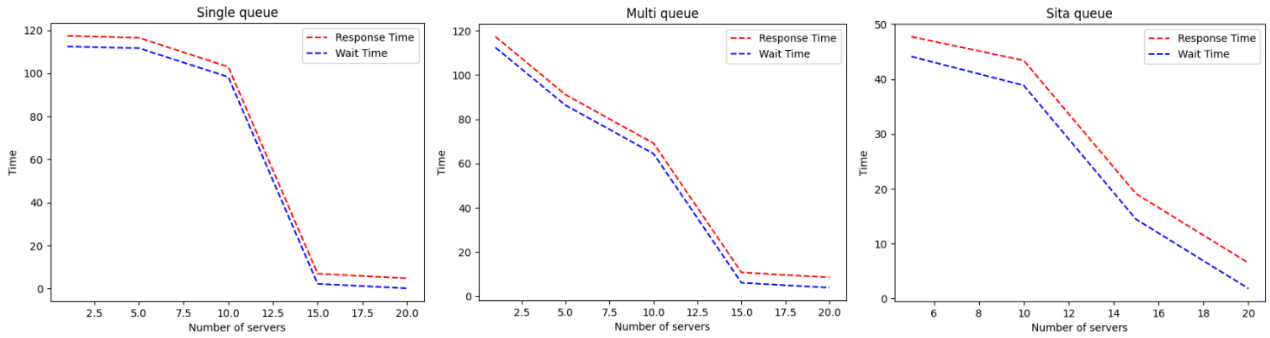
Avendo a disposizione un dataset con valori reali del tempo di attesa e di risposta dei passeggeri, si è andanti a confrontare i valori ottenuti dalla simulazione per vedere se ci fossero o no sostanziali differenze fra i due.











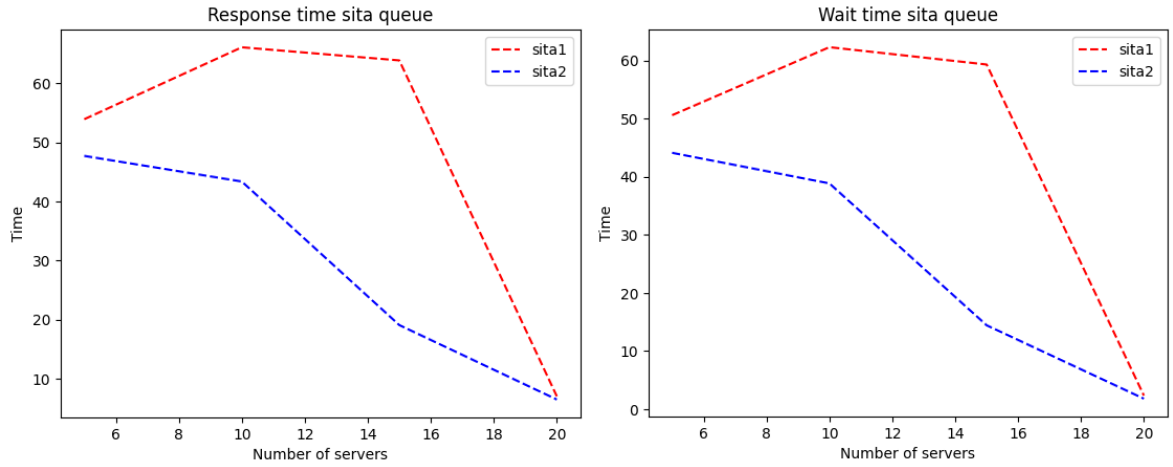
Come possiamo vedere dai grafici l'andamento dei tempi risulta essere quello atteso in quanto all'aumentare del numero degli operatori i tempi di attesa e di servizio diminuiscono. Nonostante questo, è da notare come rispetto ai dati reali forniti dal dataset, i risultati ottenuti iniziano ad essere concordi con un numero di operatori superiore a 10. Ciò è comunque considerato un buon risultato in quanto la presenza di un numero inferiore di operatori, per un aeroporto internazionale, è altamente improbabile.

7 CONCLUSIONE

Osservando i grafici discussi nel paragrafo precedente la configurazione migliore, fra tutte le possibili configurazioni, varia a seconda del numero di operatori che si hanno a disposizione. Per un numero basso di operatori, inferiore a 10, risulta essere migliore la configurazione sita, mentre all'aumentare del numero di operatori la configurazione single e multi risultano preferibili.

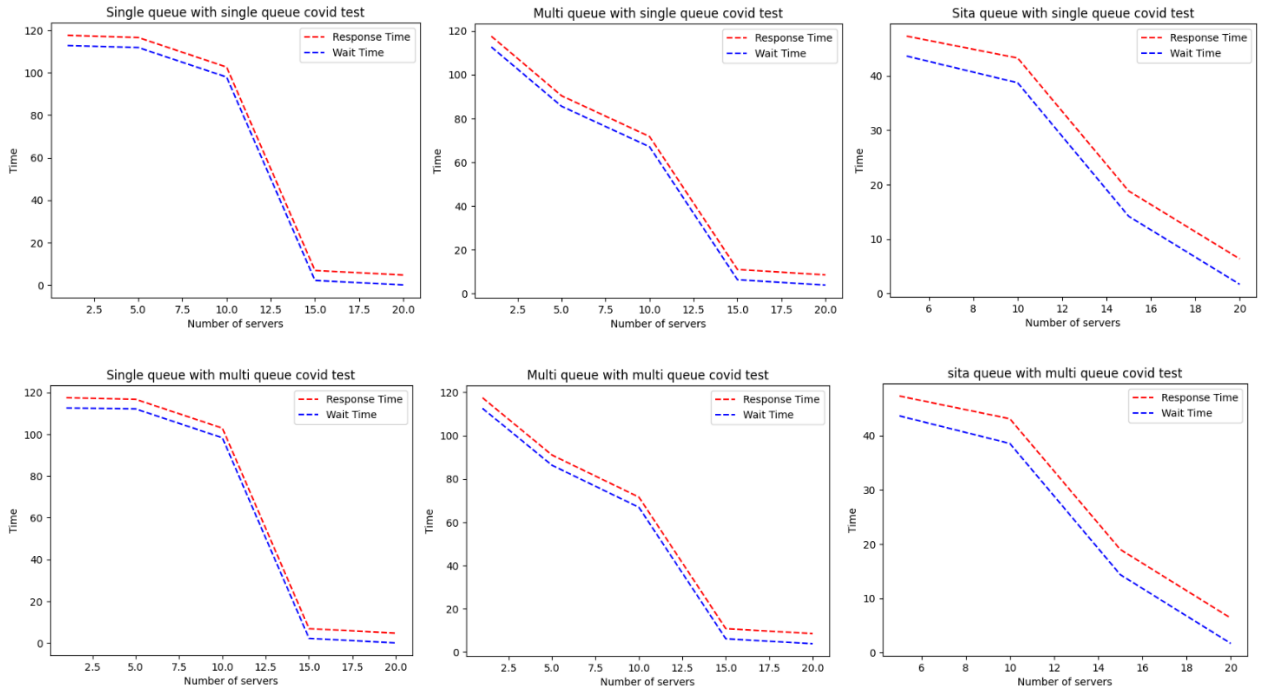
Per quanto riguarda gli abbandoni le tre configurazioni sono abbastanza simili, con un tasso di perdita lievemente inferiore per quanto riguarda la single queue.

Nonostante questo, è doveroso notare come modificando il numero di operatori associato ai diversi tipi di check-in che vengo eseguiti, si ottengano risultati differenti per quanto riguarda la sita queue.



In figura vengono riportati il tempo di risposta e di attesa considerando una configurazione sita in cui sono stati distribuiti in maniera differente gli operatori. Infatti, andando a bilanciare il numero di serventi associati ai passeggeri che devono effettuare un check-in completo, nazionale e internazionale, ovvero quelli che necessitano di un tempo di servizio maggiore, si ottiene un miglioramento dei tempi di attesa e di risposta.

Inoltre, è possibile notare come l'area del test covid non abbia particolare influenza sul tempo di risposta dell'area di check-in, indipendentemente dal tipo di configurazione utilizzata. L'unica differenza rispetto al caso in cui non sia necessario effettuare dei tamponi è possibile notarla nel numero degli abbandoni che ovviamente risente dei passeggeri risultati positivi al test.



8 BIBLIOGRAFIA

[1] «U.S. Customs and Border Protection,» [Online]. Available: https://awt.cbp.gov/.

[2] H. Mehri, T. Djemel e H. Kammoun, «SOLVING OF WAITING LINES MODELS IN THE AIRPORT USING QUEUING THEORY MODEL AND LINEAR PROGRAMMING THE PRACTICE CASE : A.I.M.H.B.,» 2008.

[3] H. Nikoue, A. Marzuoli, D. J.-P. Clarke, D. E. Feron e J. Peters, «Passenger Flow Prediction at Sydney International Airport : a data-driven queuing approach».

[4] M. N. Postorinoa, LucaMantecchini, CaterinaMalandri e FilippoPaganelli, «Airport Passenger Arrival Process: Estimation of Earliness Arrival Functions,» 2019.

[5] «Report Vaccini Anti COVID-19,» [Online]. Available: https://www.governo.it/it/cscovid19/report-vaccini/.

[6] «Coronavirus in Italia, i dati e la mappa,» [Online]. Available: https://lab24.ilsole24ore.com/coronavirus/.