

Ottimizzazione della Distribuzione dei Parcheggi Mobike

Michela Lapenna
Giordano Paoletti
Oleksandr Olmucci Poddubnyy
Davide Ravaglia

06 Gennaio 2019

1 Abstract

L'obiettivo del progetto è lo studio dell'efficienza dell'attuale distribuzione dei parcheggi offerti da Mobike nella sua area di interesse di Bologna.

Attualmente sono disponibili 70 parcheggi e l'azienda prevede di aumentarne il numero per raggiungere un totale di 240.

Si propone quindi anche una distribuzione dei 240 parcheggi totali.

Tra i risultati ottenuti, uno facilmente prevedibile è l'estrema intensificazione dei parcheggi nella zona compresa tra Piazza Maggiore e le Due Torri, seppure probabilmente non sia applicabile.

Altri risultati mostrano alcune zone nelle quali non è attualmente previsto un parcheggio, come zona Saragozza e le aree nei pressi della Sede di Ingegneria ed Architettura in località Terracini, dell'Ippodromo, dei Padiglioni di Bologna Fiere, del Giardino Valentino Borgia (zona San Donato/Pilastro) e dei Giardini Ivan Pini (zona Fossolo), per le quali se ne propone un aumento.

Inoltre si è mostrato come il flusso di biciclette abbia un carattere intensivo verso il centro della città ed estensivo in periferia, si pensa dunque che potrebbe essere utile un servizio di ridistribuzione che eviti l'agglomerazione dei mezzi in centro.

2 Introduzione

Il progetto mira alla realizzazione di un simulatore che riproduca il flusso di biciclette nel bacino di utenza di Mobike a Bologna (Fig.1) e che fornisca in output i dati relativi alla sosta delle medesime.

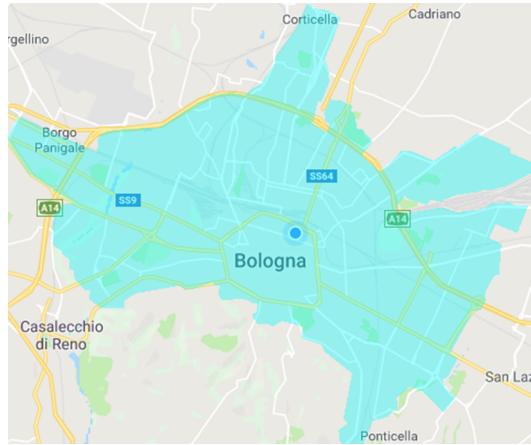


Figura 1: Bacino di utenza Mobike.

Lo scopo finale è proporre una distribuzione dei parcheggi ottimizzata, da confrontare con quella ad ora prevista da Mobike, e l'allocazione di nuovi parcheggi, nell'ottica dell'espansione del servizio programmata dall'azienda. Il dataset è costituito dai dati generati da un simulatore al Dipartimento di Fisica, i quali sono stati successivamente filtrati (vedasi par. 3.1).

Il codice numerico **ActivityId** visibile nei dati in Fig.2 identifica uno spostamento, costituito da più campionamenti, di cui sono note le informazioni temporali, le coordinate spaziali e la velocità istantanea. Sono qui mostrati solamente i campionamenti di partenza ed arrivo per ogni spostamento.

	ActivityId	Time	Latitude	Longitude	Type
0	364758	2017-04-01 00:19:36	44.48921	11.34050	D
1	364763	2017-04-01 00:22:22	44.48688	11.34036	A
2	364763	2017-04-01 00:08:54	44.50052	11.34526	D
3	364782	2017-04-01 00:23:58	44.49160	11.31413	A
4	364782	2017-04-01 00:06:37	44.49517	11.34236	D
5	364869	2017-04-01 00:38:07	44.50036	11.31987	A
6	364869	2017-04-01 00:51:22	44.49478	11.34366	D
7	364935	2017-04-01 00:59:05	44.50286	11.36601	A
8	364935	2017-04-01 01:02:10	44.50013	11.34921	D
9	364971	2017-04-01 01:09:17	44.51091	11.34989	A

Figura 2: Esempio di dati generati e filtrati.

Per ottenere solo i dati interni all'area geografica coperta da Mobike, il suo contorno è stato approssimato attraverso una poligonale di 60 segmenti. In questo modo è stato possibile rimuovere ogni percorso contenente almeno un campionamento esterno alla zona d'interesse, così da ottenere un sistema chiuso che meglio riproduca il servizio di bike-sharing.

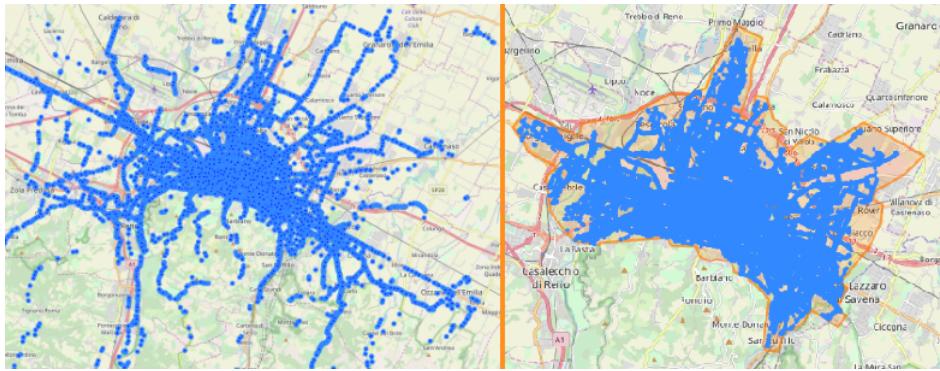


Figura 3: Poligono.

In seguito a questo filtro, sono stati mantenuti solamente i campionamenti di partenza e di arrivo per ogni percorso.

Gli eventi sono stati infine ordinati in maniera crescente rispetto all'orario, rimuovendo l'informazione relativa al giorno, in modo da averli tutti nell'intervallo di 24 ore (una giornata).

Ad ogni run il simulatore distribuisce stocasticamente le biciclette, in base al numero ed alla distribuzione scelti dall'utente. La mappa è stata suddivisa in quadrati di lato 100m dei quali si conosce il numero di biciclette disponibili all'interno.

A questo punto il programma prova a far avvenire in ordine cronologico un evento alla volta; una partenza viene eseguita se e solo se lo stato del quadrato da cui avviene lo permette, ovvero se sono presenti biciclette al suo interno, mentre un arrivo se e solo se è stato precedentemente eseguito il corrispettivo evento di partenza.

Ad intervalli di tempo regolari (parametro della simulazione), viene registrato lo stato della matrice di quadretti, che comprende il numero di entrate e di uscite avvenute in questi intervalli.

Alla fine di ogni run, i dati così registrati vengono salvati sulla memoria secondaria in formato JSON, in modo da poter essere successivamente visualizzati su browser tramite un applicativo scritto in linguaggio JavaScript (par. 3.2.6).

Eseguendo la simulazione un numero elevato di volte, si mira a trovare i 70 e i 240 quadratini che sono stati identificati più volte come parcheggi ideali. I possibili criteri di scelta considerati per l'identificazione di un parcheggio sono:

- **Ridistribuzione:** Si minimizza lo spazio totale, dato dalla somma delle distanze tra ogni bicicletta ed il parcheggio più prossimo, in modo da ottimizzare il lavoro di collezione di biciclette effettuato da Mobike.
- **Nodi:** Si individuano i principali nodi generati durante le simulazioni.

- **Densità:** Si attribuiscono le biciclette in base alla densità di popolazione delle diverse aree.
- **Modello di Mercato:** Si segue la logica che favorisce l'offerta proposta da Mobike agli utenti.

Il criterio scelto è stato quello relativo al modello di mercato, ovvero di disporre i parcheggi nei quadrati in cui è stato registrato il maggior numero di entrate avvenute, favorendo il più possibile i consumatori.

A sostegno di questa scelta è stata considerata la politica utilizzata da Mobike, la quale prevede una certa percentuale di sconto come premio per gli utenti che terminano la corsa in un parcheggio.

3 Svolgimento

3.1 Filtraggio dei dati

I dati di partenza, essendo stati generati da un simulatore, hanno richiesto un'elaborazione ed un adattamento in modo da diventare utilizzabili nel progetto corrente.

In particolare, si è cominciata questa analisi con lo studio delle distribuzioni degli intervalli spaziali (Fig.4) e temporali (Fig.5) ottenute mediante la differenza tra due rilevazioni successive.

Considerando la natura dei dati, che utilizzano longitudine e latitudine, è stata utilizzata la matrice metrica per convertire le differenze da gradi a metri.

La distanza tra due punti successivi è dunque definita tramite:

$$dS = R \sqrt{\sin^2\left((90 - Lat_i)\frac{2\pi}{360}\right) \left[(Long_{i+1} - Long_i)\frac{2\pi}{360}\right]^2 + \left[(Lat_{i+1} - Lat_i)\frac{2\pi}{360}\right]^2}. \quad (1)$$

Dove R indica il raggio medio terrestre, Lat_j e Long_j i dati di latitudine e longitudine del punto j.

In Fig.6 viene riportata la distribuzione delle velocità istantanee, in quanto relative a due rilevamenti prossimi, ottenute dagli intervalli spaziali e temporali discussi sopra. La velocità istantanea riportata nei dati è stata confrontata con la quella calcolata utilizzando la rilevazione corrente e quella precedente.

In Fig.7 viene riportata la differenza tra la velocità fornita dal simulatore e quella calcolata.

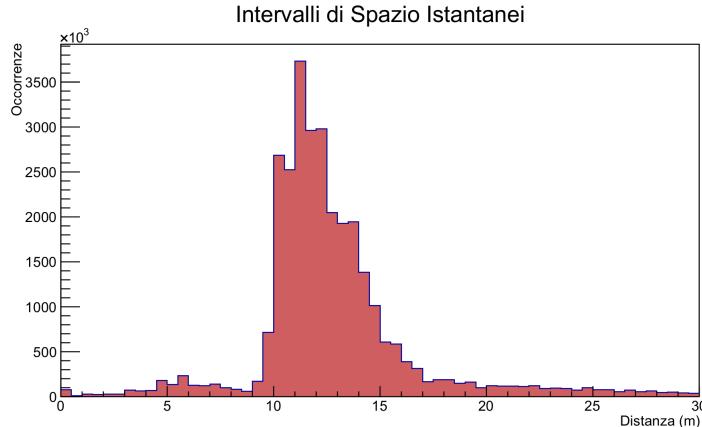


Figura 4: Intervalli spaziali ottenuti dai campionamenti consecutivi della stessa traccia.

Dalla distribuzione degli intervalli spaziali riportata in Fig.4, si osserva come il simulatore generi dati ad una distanza media compresa tra i 10 m ed i 15 m, la maggior parte degli spostamenti infatti appartiene a questo intervallo.

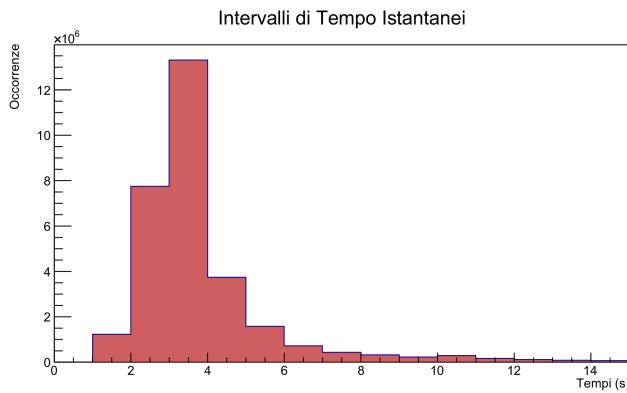


Figura 5: Intervalli temporali ottenuti dai campionamenti consecutivi della stessa traccia.

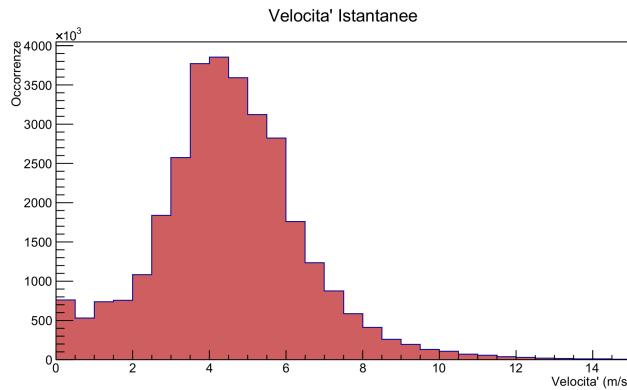


Figura 6: Velocità ottenute dai campionamenti consecutivi della stessa traccia.

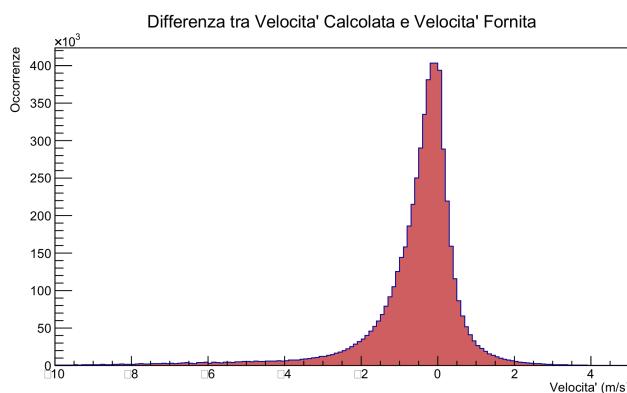


Figura 7: Differenze tra la velocità fornita dal simulatore e quella ottenuta dagli intervalli spaziali e temporali.

Si osserva che la differenza di velocità riportata in Fig.7 si distribuisce approssimativamente in maniera gaussiana attorno a 0 m/s, dunque le due velocità risultano compatibili.

Questa compatibilità può essere attribuita all'abbondante segmentazione del percorso curvilineo effettuata dal simulatore, che prevede un nuovo dato ogni 10-15 m, risultando sufficientemente precisa.

Date le caratteristiche del simulatore descritto nel par. 3.2, le uniche informazioni rilevanti di un percorso risultano essere le posizioni ed i tempi di partenza e di arrivo, ovvero le informazioni riportate in Fig.2. Dunque è stato considerato necessario applicare i filtri sulle caratteristiche complessive dei percorsi invece che su quelle puntuali. Queste caratteristiche sono presentate in Fig.8, Fig.9 e Fig.10, nelle quali vengono riportati rispettivamente gli spazi totali ed i tempi totali per traccia, ottenuti dalla somma degli intervalli spaziali e temporali tra due dati successivi, e le velocità medie per traccia, ottenute dal rapporto tra le due grandezze appena citate. Sui grafici vengono anche mostrate le fasce di valori che sono state rimosse per avere dati più rappresentativi. Più precisamente, sono stati mantenuti soltanto i valori compresi nei seguenti intervalli:

- **250m - 10km:** per lo spazio totale percorso.
- **120s - 1.5h:** per il tempo totale trascorso.
- **0.5m/s - 6.5m/s:** per la velocità media di percorrenza.

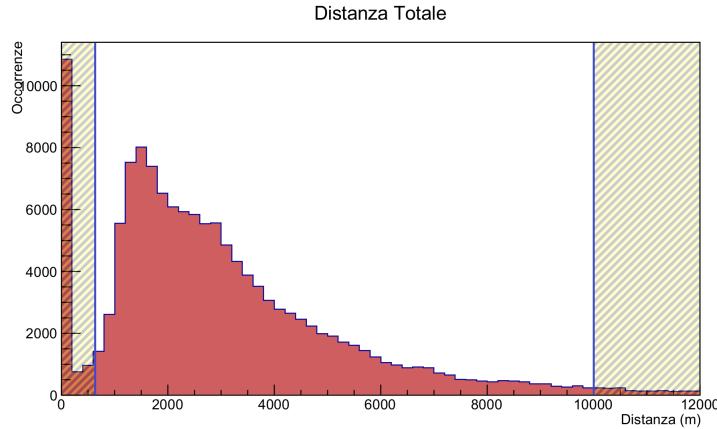


Figura 8: Distanza totale percorsa per traccia; sono evidenziati i dati successivamente filtrati.

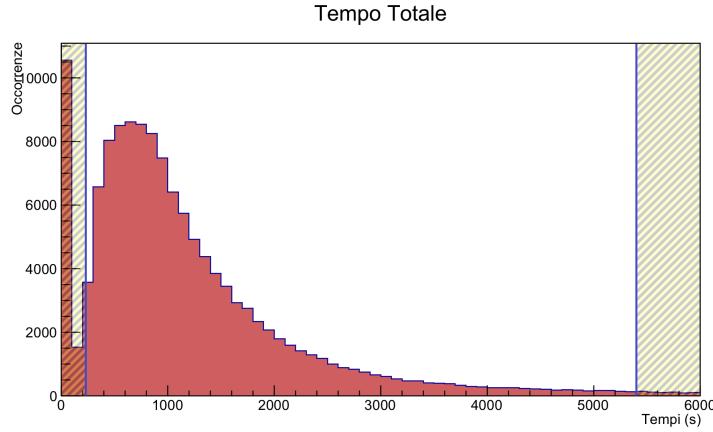


Figura 9: Tempo totale trascorso per traccia; sono evidenziati i dati successivamente filtrati.

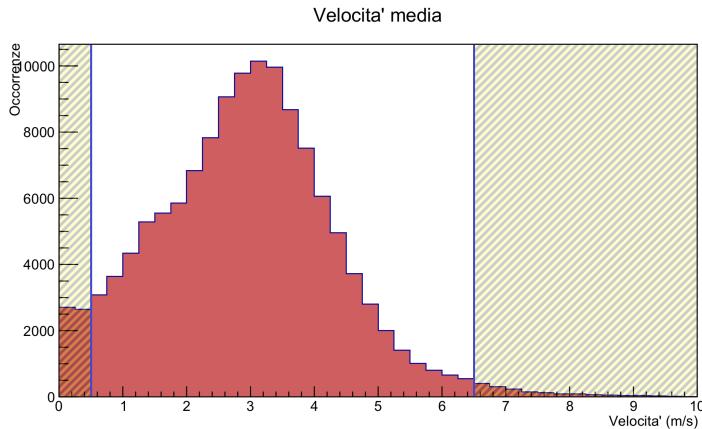


Figura 10: Velocità media per traccia; sono evidenziati i dati successivamente filtrati.

Dai grafici riportati in Fig.8 e Fig.9 si nota un picco di spostamenti aventi distanza totale e tempo totale nulli.

Queste caratteristiche sono probabilmente accoppiate in quanto non è presente un picco di spostamenti aventi velocità media nulla in Fig.10, questo poiché per gli spostamenti con tempo totale nullo non è stata calcolata la velocità media.

3.2 Simulatore

Considerando la natura discreta dei dati generati ed elaborati come descritto nel par.3.1, si è deciso di utilizzare un modello di simulazione ad eventi discreti (DES), ovvero un modello in cui le variabili della simulazione possono avere cambiamenti di valore soltanto in corrispondenza di eventi discreti.

Il modello ha dunque il compito di prendere i dati relativi agli eventi avvenuti, caratterizzati da istante temporale, identificativo, posizione e tipo di evento

(partenza o arrivo), e di simulare a partire da essi i singoli spostamenti da una posizione all'altra della mappa, tenendo traccia per ciascuna di esse del numero di partenze e di arrivi soddisfatti, chiamati anche uscite ed ingressi.

3.2.1 Rappresentazione delle zone

Al fine di diminuire il numero di posizioni per cui viene effettuato il conteggio, si è deciso di cambiare la rappresentazione delle zone, convertendo le posizioni di tutti gli eventi dal sistema di coordinate GPS ad un sistema di coordinate discreto spaziato di 100 m per unità e con l'origine nel punto (44.45216343349134, 11.255149841308594) (Fig.11), la cui precisione è data dalla risoluzione nell'evento onClick della libreria LeafletJS¹, la libreria utilizzata per la parte grafica (par.3.2.6).

In questo modo viene generata una griglia (Fig.12) di cui viene considerato un numero finito di righe e colonne, detti bordi della simulazione.

La conversione di un punto espresso in latitudine e longitudine è effettuata mediante le seguenti formule

$$\begin{aligned} \text{row} &= \left\lfloor \frac{\text{Latitude} - \text{OriginLatitude}}{\text{coef}} \right\rfloor \\ \text{column} &= \left\lfloor \frac{\text{CosLatitude} * (\text{Longitude} - \text{OriginLongitude})}{\text{coef}} \right\rfloor \end{aligned}$$

dove $\text{coef} = 100 * 0.0000089$ è un coefficiente moltiplicativo corrispondente a 100 m espressi in gradi e $\text{CosLatitude} = \cos((\text{OriginLatitude} + \text{coef}) * 0.018)$ è un'approssimazione della latitudine all'origine.

La procedura che utilizza le due formule descritte permette di calcolare gli indici del quadrato contenente il punto originale, espressi in riga e colonna.

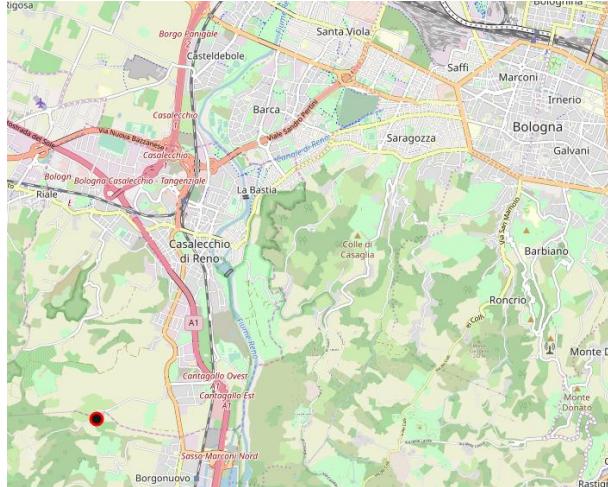


Figura 11: Punto di origine della griglia, con coordinate (44.45216343349134, 11.255149841308594), in basso a sinistra evidenziato in rosso e nero.

¹<https://leafletjs.com/>

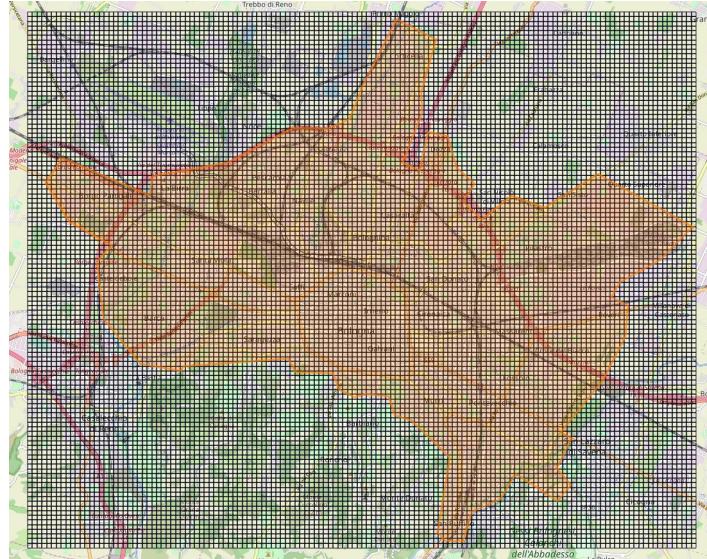


Figura 12: Griglia generata a partire dal punto (44.45216343349134, 11.255149841308594), con spaziatura di 100 m.

3.2.2 Simulazione con risorse illimitate

Si è deciso inizialmente di partire con un modello di simulazione semplificato, che non tenesse conto delle risorse disponibili, con lo scopo di soddisfare tutti gli eventi possibili.

La procedura di simulazione con risorse illimitate prende in input quattro parametri:

1. **Tempo totale:** espresso in unità minima di simulazione (minuti nel nostro caso).
2. **Passo di tempo (Δt):** passo di tempo con cui avanza il clock della simulazione.
3. **Bordi:** bordi dell'area della simulazione espressi in numero di righe e di colonne (si è deciso di tenere fisso questo parametro, mettendo un limite di 140 per x e di 115 per y, al fine di coprire tutta l'area interessata usando il rettangolo più piccolo).
4. **Eventi:** dataset degli eventi.

La simulazione viene eseguita incrementando il proprio clock in base al passo (Δt) passato come parametro, fino al raggiungimento del tempo massimo di simulazione. Ad ogni istante t di clock, viene associato un intervallo di tempo $[t - \Delta t, t]$, utilizzato dal simulatore per capire quali eventi sono avvenuti. Vengono gestiti due tipi di eventi: *partenza* e *arrivo*. La gestione di un evento di *partenza* avviene inserendo l'identificativo dell'evento in una tabella di partenze ed incrementando il contatore di uscite per il quadrato interessato.

La gestione di un evento di *arrivo* avviene verificando la presenza del relativo identificativo nella tabella delle partenze, incrementando il contatore relativo

alle biciclette entrate nel quadrato dell'evento e rimuovendo l'id dalla tabella delle partenze. Una volta che il clock della simulazione ha raggiunto il tempo massimo della simulazione, la procedura restituisce un tensore indicizzato nel tempo, con riga e colonna contenenti per ogni istante o ogni posizione il numero di entrate ed uscite avvenute. Lo pseudocodice della procedura di simulazione con risorse illimitate è illustrato nella Fig.13.

Pur essendo un modello poco realistico, la sua realizzazione è servita inizialmente per verificare la correttezza dell'approccio e per trovare i quadrati che sono soggetti a traffico, ovvero che hanno contatto almeno un ingresso o un'uscita nell'arco della simulazione.

Conoscere il traffico dei singoli quadrati con risorse illimitate può essere utile per fornire i limiti teorici nel caso in cui si utilizzasse uno schema di simulazione a risorse limitate.

```

function SIMULATE(TotalTime, Δt, Bounds, Events)
    location_map  $\leftarrow$  Tensor3D(Bounds.t, Bounds.y, Bounds.x)
    arrivals  $\leftarrow$  Dictionary()
    t  $\leftarrow$  Δt
    while t  $\leq$  TotalTime do
        E  $\leftarrow$  {e  $\in$  Events | e.time  $\in$  [t  $-$  Δt, t]}
        for e  $\in$  E do
            (i, j)  $\leftarrow$  CoordinateConverter(e.latitude, e.longitude)
            if e.type = Arrival and arrivals.contains(e.id) then
                location_map[t, i, j].in $+=1$ 
                arrivals.remove(e.id)
            else if e.type = Departure then
                location_map[t, i, j].out $+=1$ 
                arrivals.insert(e.id)
            end if
        end for
        t  $\leftarrow$  t  $+ \Delta t$ 
    end while
    return location_map
end function

```

Figura 13: Schema di simulazione con risorse illimitate.

3.2.3 Simulazione con risorse limitate

Il modello a risorse illimitate, sebbene semplice, non è realistico per il problema affrontato, si ha quindi la necessità di introdurre un meccanismo di assegnamento delle risorse (biciclette nel nostro caso) alle zone della mappa.

Quest'ultimo è stato realizzato tramite l'utilizzo delle funzioni di piazzamento, ovvero di funzioni da [*Bounds.y*, *Bounds.x*] a N.

Il compito di queste funzioni è quello di prendere uno specifico quadrato della mappa e di assegnare ad esso un certo numero di risorse, senza eccedere un parametro N (numero di risorse totali disponibili).

Utilizzando questo meccanismo, il simulatore a risorse illimitate (Fig.13) viene modificato aggiungendo i controlli, gli incrementi e i decrementi delle risorse disponibili, così che questo ora faccia avvenire gli eventi di partenza solamente

nel caso in cui la zona di quell'evento contenga almeno un'unità di risorse (p.e. almeno una bicicletta).

Lo pseudocodice dello schema di simulazione a risorse limitate è illustrato nella Fig.14.

```

function SIMULATE_LIMITED(TotalTime,  $\Delta t$ , Bounds, Placement, N, Events)
    location_map  $\leftarrow$  Tensor3D(Bounds.t, Bounds.y, Bounds.x)
    available  $\leftarrow$  Matrix(Bounds.y, Bounds.x)
    arrivals  $\leftarrow$  Dictionary()
     $t \leftarrow \Delta t$ 
    {% Allocate N units of a resource according to the Placement function %}
    while  $t \leq TotalTime$  do
         $E \leftarrow \{e \in Events \mid e.time \in [t - \Delta t, t]\}$ 
        for  $e \in E$  do
             $(i, j) \leftarrow$  CoordinateConverter(e.latitude, e.longitude)
            if e.type = Arrival and arrivals.contains(e.id) then
                location_map[ $t, i, j$ ].in += 1
                available[ $i, j$ ] += 1
                arrivals.remove(e.id)
            else if e.type = Departure and available[ $i, j$ ] > 0 then
                location_map[ $t, i, j$ ].out += 1
                available[ $i, j$ ] -= 1
                arrivals.insert(e.id)
            end if
        end for
         $t \leftarrow t + \Delta t$ 
    end while
    return location_map
end function

```

Figura 14: Schema di simulazione con risorse limitate.

3.2.4 Dettagli di implementazione

Lo pseudocodice degli schemi di simulazione nasconde delle particolarità implementative che possono influire parecchio sul tempo e sullo spazio richiesti dal programma che li implementa. Sono state rilevate tre operazioni principali, la cui ottimizzazione ha ridotto nettamente il consumo delle risorse computazionali:

1. Selezione degli eventi.
2. Conversione delle coordinate.
3. Rappresentazione di *location_map*.

1) *Selezione degli eventi*: Il tempo di esecuzione di questa operazione dipende fortemente dalla rappresentazione e dall'ordinamento dei tempi degli eventi. Essendo il tempo degli eventi contenuto tra 0 e 24 ore (dato che non viene tenuta in considerazione la data) e il clock del simulatore rappresentato in minuti, una facile ottimizzazione è stata quella di convertire tutti i tempi in minuti passati

dalla mezzanotte e dividere questi valori per il passo della simulazione. Si ottiene così il numero dell'intervallo in cui un evento dovrebbe accadere, in modo da costruire dei gruppi di eventi in base all'intervallo in cui occorrono. Questi gruppi vengono poi considerati dal simulatore in ordine crescente, evitando così ogni volta di dover effettuare una ricerca nell'insieme degli eventi.

2) *Conversione delle coordinate*: La procedura di conversione delle coordinate viene chiamata ancora più spesso di quella della *selezione degli eventi*, perciò è preferibile averla calcolabile in tempo $O(1)$ (costante), come spiegato precedentemente in par.3.2.1.

3) *Rappresentazione di location_map*: L'approccio naïf alla rappresentazione di *location_map* prevederebbe l'utilizzo di un tensore con $O(T \times Y \times X)$ elementi, di cui la maggior parte uguali a 0. La tecnica per ridurre l'occupazione di memoria qui utilizzata è quella della memorizzazione dei tensori sparsi tramite LIL (list of lists), in modo da portare il caso ottimo, con zero eventi realizzati, da $\Omega(T \times Y \times X)$ a $\Omega(1)$, riducendo il consumo di memoria di diversi ordini di grandezza. La tecnica List of Lists consiste nel salvare in una lista di liste solamente gli elementi diversi da zero, i quali nel caso del nostro simulatore sono quadrati della mappa che hanno avuto traffico.

Applicando queste ottimizzazioni, si è ottenuto un miglioramento di diversi ordini di grandezza sia in tempo che in spazio, rendendo possibile l'esecuzione di molteplici run della simulazione.

3.2.5 Output della simulazione

Una volta terminata la simulazione, la procedura restituisce un tensore memorizzato come spiegato in par. 3.2.4, contenente le informazioni di transito nelle celle per ogni intervallo temporale.

Questi dati vengono salvati insieme ad altri metadati, tra cui i bordi ed il tempo totale di simulazione, in un file di log in formato JSON.

Un esempio di file di log è mostrato in Fig.15

.
La scelta di questo formato deriva dal fatto che è un formato standard, il quale permette di rappresentare in modo chiaro anche oggetti complessi ed è in più supportato nativamente in JavaScript, linguaggio in cui è stata scritta l'applicazione per visualizzare questi dati.

```
"meta_data": { "time_delta": 15, "bounds": { "t": 96, "y": 115, "x": 140 } }, "map": [ { "48-68": { "in_bikes": 0, "out_bikes": 3 }, "54-70": { "in_bikes": 0, "out_bikes": 2 } } ] }
```

Figura 15: Esempio di file salvato dopo un run di simulazione.

3.2.6 Visualizzazione della simulazione

Effettuato un run di simulazione, il log prodotto dal simulatore può essere visualizzato tramite un applicazione per il browser scritta in JavaScript. L'applicazione, come illustrato in Fig.16, permette di visualizzare graficamente su una

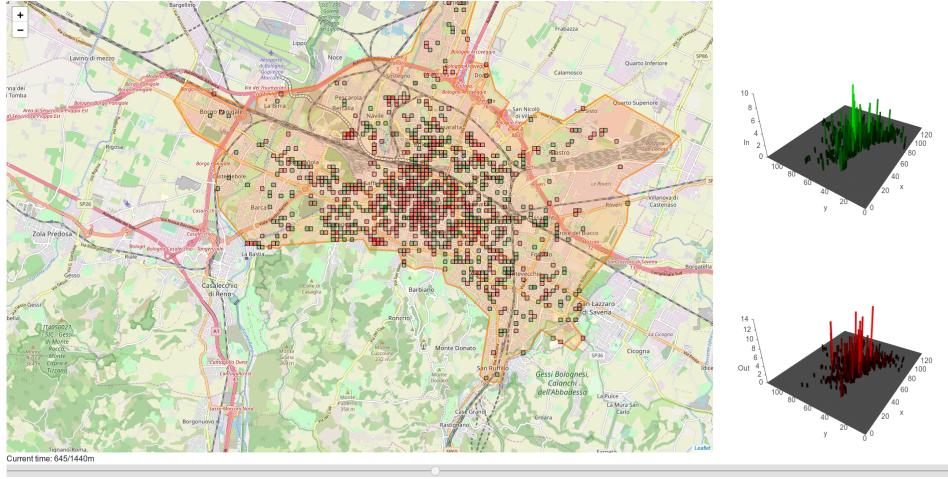


Figura 16: Visualizzatore dei log della simulazione.

mappa le zone che sono state soggette a traffico, colorando in verde i quadrati con più entrate che uscite e in rosso il caso contrario. La densità di traffico è invece rappresentata dalla trasparenza dei quadrati e varia tra 0.2 per il traffico minimo e 0.9 per il traffico massimo. Oltre alla rappresentazione sulla mappa, i dati relativi al traffico vengono rappresentati su due istogrammi, rappresentanti singolarmente le entrate e le uscite. È possibile inoltre cambiare l'istante temporale della visualizzazione tramite uno slider.

Nonostante tutti i vantaggi offerti dall'utilizzo di JavaScript per questo tipo di visualizzazione, il problema principale che si è riscontrato è stato quello dell'impossibilità di accesso al filesystem da parte del browser nel caso in cui si utilizzi soltanto JavaScript lato client. Questo significa che un'applicazione che non utilizza un server, ma rappresenta soltanto pagine con JavaScript, non può accedere né elencare i contenuti delle directories in cui sono contenuti per esempio i log della simulazione. Per poter visualizzare un log diverso, si rende così necessario l'utilizzo dello stesso nome per tutti i file di log, oppure la modifica di una stringa dentro al codice del programma. Una soluzione a questo problema potrebbe essere l'aggiunta di un semplice server che, tramite una chiamata AJAX, restituisca il contenuto della directory interessata, permettendo al browser di vedere i nomi di tutti i file contenuti in quella directory.

3.3 Scelta dei parametri utilizzati nella simulazione

La distribuzione iniziale scelta per le biciclette è stata quella proporzionale al numero di entrate registrate nei quadratini, utilizzando il simulatore con un numero illimitato di risorse. Tramite questa distribuzione si mira a compiere una prima approssimazione della disposizione geografica delle biciclette.

Per la scelta del numero di biciclette da utilizzare nella simulazione sono state applicate due richieste: garantire una variazione nelle distribuzioni iniziali delle biciclette nei diversi run ed allo stesso tempo avere una rappresentazione fedele del sistema. Partendo da simulazioni aventi durata di 1 giorno, si è studiato il numero di percorsi totali realizzati ed il numero medio di percorsi realizzati

per bicicletta, al variare del numero di biciclette. Per questi due studi la distribuzione iniziale scelta non è stata randomizzata, ma le biciclette sono state distribuite semplicemente in modo proporzionale al numero di entrate totali.

Si è quindi eseguito uno studio sulla ripetizione dei percorsi tra diversi run, in modo da assicurare una variazione tra run successivi. A tal scopo sono stati effettuati 10 run del simulatore per ognuno dei diversi valori del numero di biciclette scelto, quindi sono state studiate le ripetizioni dei percorsi avvenuti nei 10 run, chiamate molteplicità. Considerando i percorsi aventi molteplicità 8, 9 e 10 come nucleo degli spostamenti giornalieri, l'obiettivo è utilizzare un numero di biciclette tale per cui questo nucleo sia considerevole ma non estremamente predominante.

Essendo la simulazione parametrizzabile su più giorni consecutivi, è stato studiato l'andamento del rapporto tra i percorsi realizzati in un giorno ed i percorsi realizzati il giorno precedente, per una durata della simulazione di 15 giorni e per numeri diversi di biciclette.

In base a questi studi si è deciso di utilizzare come parametri della simulazione 900 biciclette per una durata di 1 giorno. Quindi è stata mostrata la fluttuazione del numero di spostamenti generati utilizzando questi parametri. (Vedere sezione 4)

Infine si è scelto di evidenziare la posizione dei quadretti contenenti:

- **Pozzi:** dove è massima la differenza tra entrate avvenute ed uscite avvenute.
- **Sorgenti:** dove è massimo il numero di partenze non soddisfatte.
- **Parcheggi ottenuti tramite l'utilizzo di un numero illimitato di risorse:** dove, fornendo al simulatore un numero sufficiente di biciclette in modo da realizzare ogni spostamento, è massimo il numero di percorsi che vi terminano all'interno.
- **Parcheggi ottenuti tramite i parametri scelti:** dove, utilizzando il simulatore con 900 biciclette per la durata di una giornata, è massimo il numero di percorsi avvenuti che vi terminano all'interno.

4 Risultati

In Fig.17 si può osservare l'effetto che il filtro sul bacino dell'area di Mobike provoca in vicinanza dei bordi.

Avendo infatti eliminato i tragitti per i quali almeno un punto è esterno all'area, in prossimità dei bordi non sono presenti rilevazioni, mentre queste aumentano di densità avvicinandosi verso il centro del bacino.

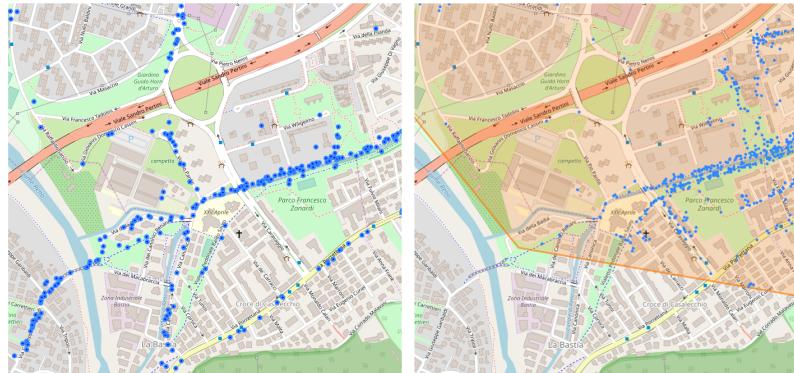


Figura 17: Dettaglio degli eventi generati sulla mappa prima e dopo il filtro geografico.

Nei grafici in Fig.18 e Fig.19 è riportato il numero di spostamenti realizzati in una giornata ed il numero medio di spostamenti realizzati in una giornata per bicicletta al variare del numero di queste.

Conoscendo il numero totale di spostamenti realizzabili in una giornata (233632), nel primo grafico si osserva come il numero di percorsi realizzati ci si avvicini costantemente e, sempre da questo grafico, è possibile risalire alla percentuale di spostamenti realizzati dato il numero di biciclette.

Nel secondo grafico si osserva come il massimo del numero medio di spostamenti per bicicletta avvenga per un numero di biciclette molto basso, compreso tra 153 e 257.

Non è possibile utilizzare il numero di biciclette corrispondente al massimo nella simulazione finale in quanto è troppo poco rappresentativo del sistema, infatti per quei valori di biciclette non avvengono nemmeno il 3% degli spostamenti totali.

Inversamente non verrà scelto nemmeno un numero di biciclette superiori alle 10000, in quanto in tal caso le biciclette utilizzate realizzerebbero meno della metà degli spostamenti medi giornalieri massimi e quindi si perderebbe in efficienza.

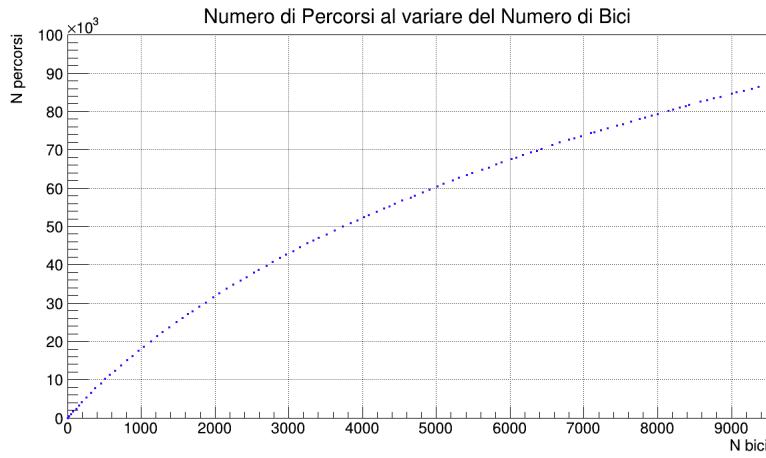


Figura 18: Numero di percorsi realizzati dalle biciclette in un giorno al crescere del numero di biciclette.

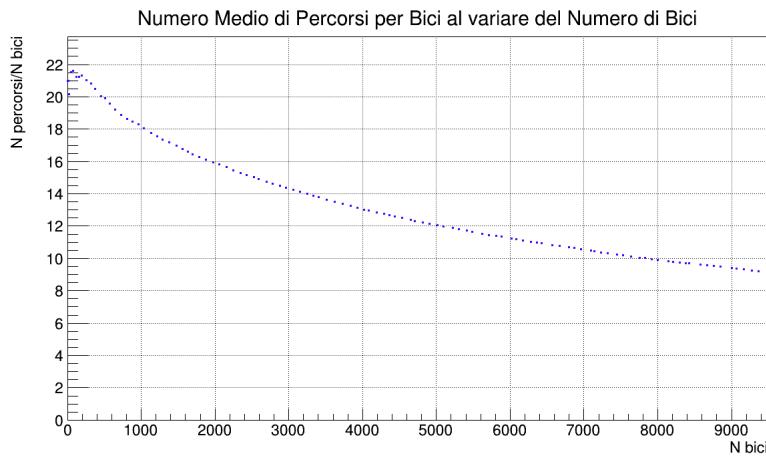


Figura 19: Numero medio di percorsi per bicicletta in un giorno al crescere del numero di biciclette.

Risulta quindi fondamentale per la scelta del numero di biciclette la richiesta di variazione della disposizione iniziale delle stesse; se queste sono distribuite diversamente all'inizio della giornata compieranno diversi spostamenti, quindi è stato affrontato lo studio sulle molteplicità prima citate. In Fig.20 viene rappresentato, per diversi numeri di biciclette, il numero degli spostamenti che sono stati ripetuti N (1,2,...,10) volte su 10 run, diviso per il numero degli spostamenti totali avvenuti nei 10 run.

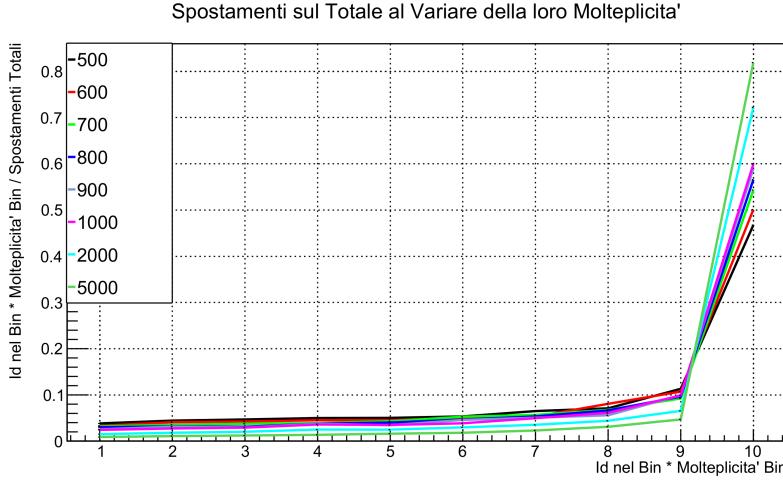


Figura 20: Spostamenti avvenuti N su 10 volte al variare di N.

I valori sono stati normalizzati utilizzando il numero degli spostamenti totali avvenuti, in modo da poter meglio comparare curve che altrimenti sarebbero state separate. Si precisa che il valore presente al bin N non è il numero delle ActivityId che sono state ripetute N volte su 10, ma quel numero moltiplicato per N. Si osserva come, anche per valori relativamente bassi del numero di biciclette, la maggior parte degli spostamenti venga ripetuto 10 volte e come questa percentuale cresca rapidamente al crescere del numero di biciclette.

In Tab.4 vengono riportati in funzione del numero di biciclette: la percentuale di spostamenti aventi molteplicità 8, 9 o 10; la percentuale di spostamenti realizzati sul totale ottenuta dal grafico in Fig.18; il numero di spostamenti medio per bicicletta ottenuto dal grafico in Fig.19.

Biciclette	% Molteplicità 8,9,10	% Spost. Realizzati	Spost./Biciclette
500	65.2%	4.3%	20.2
600	68.9%	5.0%	19.5
700	70.3%	5.7%	19.1
800	72.9%	6.4%	18.7
900	74.5%	7.1%	18.3
1000	75.9%	7.7%	18.0
2000	83.1%	13.7%	15.9
5000	95.4%	26.1%	12.2

In base a questi risultati si è scelto di utilizzare un numero di 900 biciclette per la simulazione finale. Infatti, superando questo numero, più di tre quarti degli spostamenti risulterebbero uguali tra due run consecutivi, riducendo la diversità tra run diversi. Allo stesso tempo, scegliendo un numero di biciclette inferiore, avremmo un sistema sempre meno rappresentativo dell'interità degli spostamenti. Inoltre, come si osserva da Fig.20, la spezzata relativa al numero 900 si trova in una banda di valori tali per cui l'andamento delle molteplicità non subisce grandi variazioni.

Per scegliere il numero di giorni di esecuzione è stato mostrato l'andamento

del numero di spostamenti al giorno N diviso per il numero di spostamenti al giorno $N-1$, riportato in Fig.21. Nonostante sia già stato scelto il numero di biciclette da adoperare, questo andamento è stato studiato per diversi numeri di biciclette, distanziati in maniera quadratica. Si osserva come, nonostante la distanza tra i diversi numeri di biciclette sia elevata, solo la spezzata relativa a 102400 biciclette si separi nettamente dalle altre, in quanto si avvicina al limite del sistema a risorse illimitate e tende quindi a perdere giornalmente meno spostamenti in percentuale. Per i rimanenti numeri di biciclette riportati nel grafico in linea di massima avviene una perdita di percorsi da un giorno al successivo compresa tra il 4% ed il 6%. Questa perdita tende a diminuire nel tempo, indicando una possibile convergenza del sistema. Si osserva come per numeri minori di biciclette questa perdita sia leggermente più consistente. Nonostante la perdita non sia troppo elevata si è deciso di eseguire la simulazione finale per un'unica giornata, al fine di evitare di perdere spostamenti a causa di possibili pozzi ed insieme a questi perdere una rappresentazione fedele del dataset.

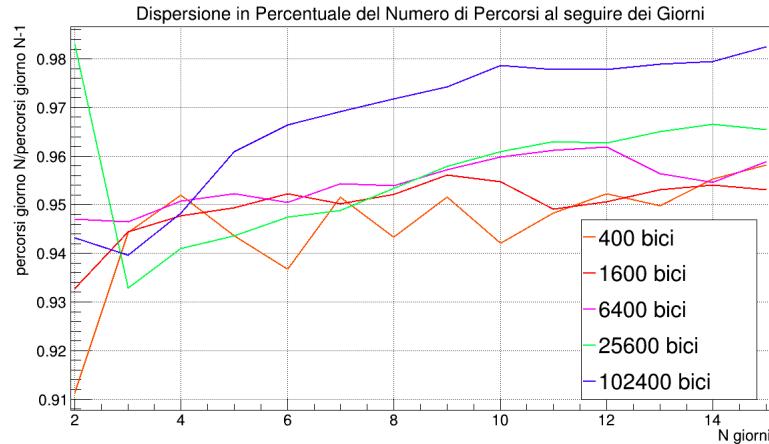


Figura 21: Dispersione in percentuale del numero di percorsi tra un giorno e il successivo su 15 giorni.

Infine, per 20 run diversi, viene mostrato numero di spostamenti avvenuti in una giornata con il valore scelto a 900 biciclette. Questo per verificare che questo valore non vari in maniera significativa. Si osserva come il numero di spostamenti realizzati resta pressocchè costante.

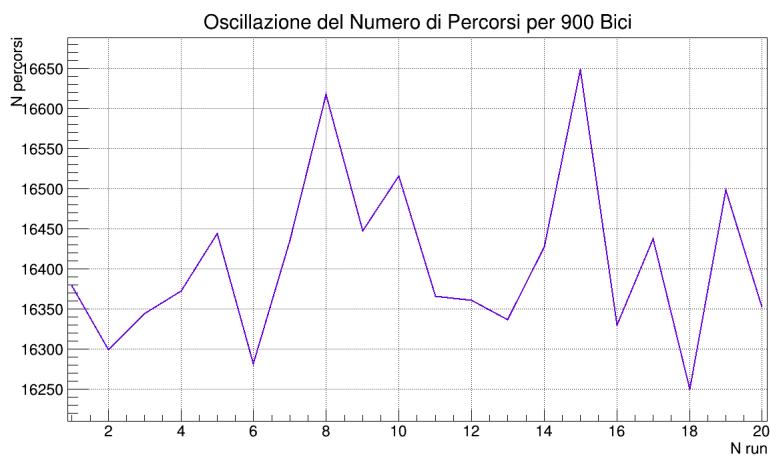


Figura 22: Oscillazione del numero dei percorsi realizzati per run da 900 biciclette con durata 1 giornata.

5 Conclusione

5.1 Parcheggi Individuati

Il programma è stato infine eseguito più volte, fino ad ottenere una soluzione stabile dopo un numero di esecuzioni maggiore di 100.

Come anticipato, sono stati individuati i 70 e i 240 quadrati che per più volte (per più giorni) sono risultati maggiormente transitati, cioè hanno registrato il maggior numero di entrate. I risultati ottenuti sono visibili in Fig.23 .

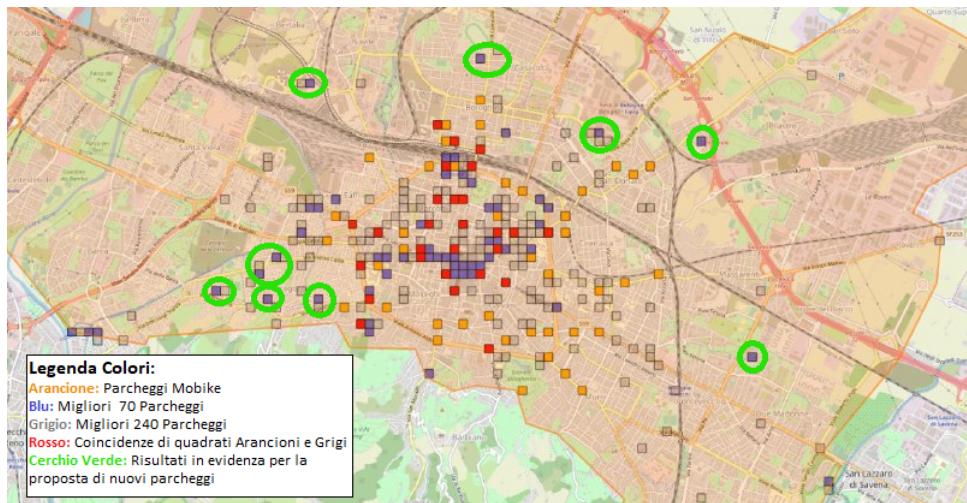


Figura 23: Rappresentazione e Confronto dei parcheggi ottenuti.

Il primo risultato, facilmente prevedibile, è un'estrema intensificazione dei parcheggi nella zona compresa tra Piazza Maggiore e le Due Torri. Questo risultato è tuttavia probabilmente non realizzabile. Notiamo invece come la zona sud-est della città (Via Murri, Giardini Margherita ecc..) ne sia particolarmente fornita, seppure dal nostro elaborato il uso di biciclette non risulti molto elevato.

Le aree cerchiate in verde in figura rappresentano i risultati di maggiore interesse: sono stati evidenziati infatti i punti ottenuti che non presentano nelle vicinanze un parcheggio Mobike. In particolare il quartiere Saragozza risulta essere molto trafficato, ma quasi completamente privo di parcheggi. Altre zone di particolare interesse sono quelle nei pressi di

- Sede di Ingegneria ed Architettura in località Terracini
- Ippodromo
- Padiglioni di Bologna Fiere
- Giardino Valentino Borgia (zona San Donato/Pilastro)
- Giardini Ivan Pini (zona Fossolo)

Si propone quindi di disporre alcuni parcheggi in queste aree ancora poco servite.

5.2 Confronto tra risultati ottenuti tramite simulatore con risorse limitate e senza.

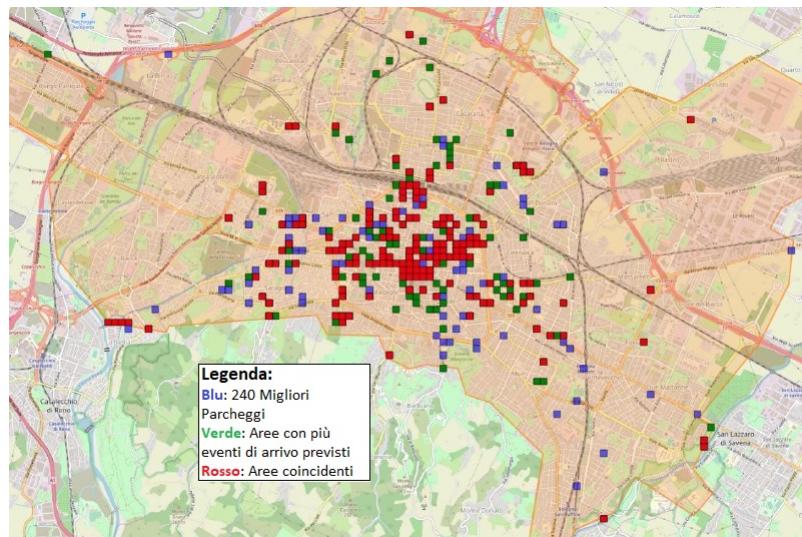


Figura 24: Confronto tra Risultati ottenuti.

La Fig.24 mostra il confronto che è stato eseguito al seguito dell'utilizzo delle due tipologie di simulatore.

Il simulatore a risorse illimitate evidenzia i 240 quadrati che hanno più ingressi a priori, cioè più eventi di arrivo previsti al loro interno.

Si vede che, seppure la maggior parte delle aree trovate impiegando la simulazione con 900 biciclette coincida con quelle citate sopra, il restante 33% circa se ne discosta, proponendo soluzioni più ponderate.

5.3 Aree intensive ed estensive

L'ultima analisi è stata dedicata allo studio delle aree di carattere estensivo ed intensivo. Con i metodi introdotti nel par.3.3 sono stati individuati i pozzi e le sorgenti del flusso di cicli simulato.

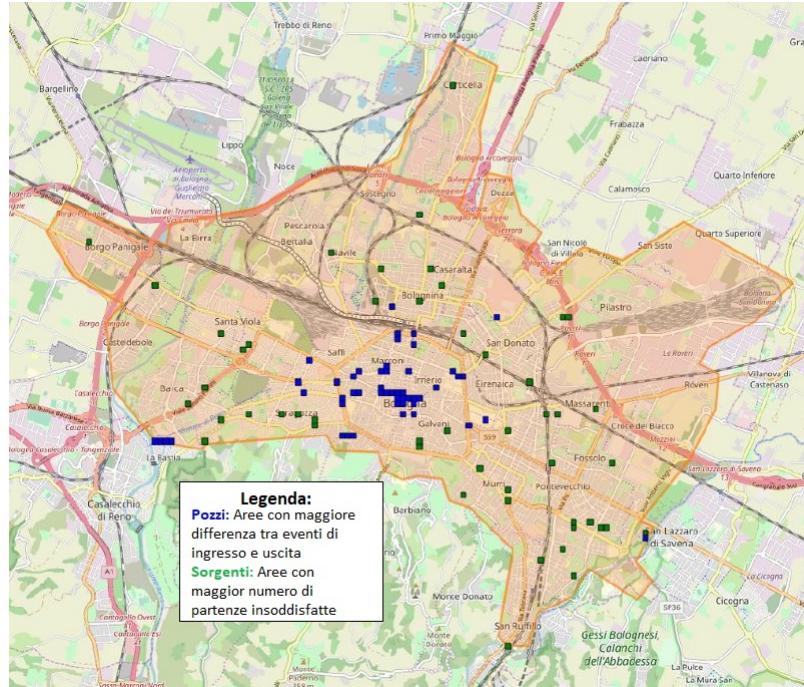


Figura 25: Rappresentazione delle aree estensive (Sorgenti) ed intensive (Pozzi).

Come si può vedere in Fig.25, il centro della città rappresenta il punto di agglomerazione di tutti i mezzi disponibili, mentre le aree esterne fungono da sorgenti, rimanendo molto presto prive di biciclette. Questo risultato potrebbe suggerire che il flusso non sia completamente autonomo e che quindi possa esserci la necessità di un servizio di ridistribuzione che riporti le biciclette di Mobike verso l'esterno della città, così da garantire una maggiore presenza su tutto il territorio.