Alma Mater Studiorum · Università di Bologna

Scuola di Scienze
Dipartimento di Fisica e Astronomia
Corso di Laurea in Fisica

STUDIO DELL'OTTIMIZZAZIONE DI UNA RETE SEMAFORICA

Relatore:
Prof. Armando Bazzani

Correlatore:
Dott. Filippo Dalla

Presentata da:
Veronica Mungai

Anno Accademico 2023/2024

Abstract

In questa tesi viene presentato un algoritmo per ottimizzare il flusso di traffico regolato da una rete semaforica in un reticolo stradale tipo *Manhattan-like* planare omogeneo. L'obiettivo è quello di rendere più scorrevole la rete, vale a dire aumentare il flusso di agenti a parità di densità. La performance dell'algoritmo è stata confrontata con quella di un altro algoritmo di ottimizzazione realizzato precedentemente. Quest'ultimo, infatti, non è in grado di aumentare il flusso medio rispetto a una situazione in cui la rete non è soggetta ad alcuna ottimizzazione. Per verificare il corretto funzionamento dell'algoritmo è stato utilizzato un Framework costruito sulla base di un modello mesoscopico per simulare il network e la sua dinamica. Dalle simulazioni, in cui si propone un network soggetto a una carica adiabatica fino alla sua completa congestione, si evince che l'algoritmo è in grado di aumentare la scorrevolezza della rete e si hanno miglioramenti anche in termini di diminuzione dei tempi di percorrenza e ritardo della formazione della congestione. Sono state inoltre effettuate simulazioni di carico variabile in cui si riscontrano effetti positivi sul flusso medio e sulla densità media da parte dell'algoritmo; i maggiori effetti si possono osservare in particolare nella simulazione a carico alto.

Indice

Elenco delle figure

Introduzione

Spesso quando si parla di "fisica" si tende a pensare a branche come la Meccanica, l'Elettromagnetismo o la Fisica delle Particelle. Ci si potrebbe chiedere che cosa c'entri la "fisica" con lo studio di una rete semaforica o del traffico in generale. La verità è che questa parola può essere intesa in un senso molto più ampio: trae la sua origine dal greco antico φυσικός, -ή, -όν che condivide la radice con φύσις, -εως. Spesso tradotta semplicemente con "*natura*", in realtà φύσις racchiude in sé molteplici significati che si collegano al concetto di proprietà costitutive di un sistema. In un certo senso la fisica si occupa di queste proprietà costitutive indipendentemente dal fatto che il sistema in analisi sia un "sistema fisico" propriamente detto oppure no.

Lo studio del traffico è entrato a far parte dei campi di ricerca della fisica a partire dagli anni '50, quando sono stati ideati i primi modelli capaci di descrivere questo sistema [1]. Dal momento che il traffico ha una dinamica articolata, all'interno di un singolo modello non è possibile considerare tutte le variabili che descrivano nel dettaglio il suo comportamento, pertanto i modelli fisici che lo analizzano prendono in esame solamente gli elementi più essenziali del sistema in maniera tale da fornire una descrizione realistica ma generale dei fenomeni che lo caratterizzano [2].

Il traffico stradale non solo causa disagi durante gli spostamenti, ma è anche associato al tema del consumo delle risorse e dell'inquinamento ambientale [3]. L'ottimizzazione di alcuni aspetti, ad esempio dei tempi di percorrenza o della distanza percorsa per arrivare a una determinata destinazione, permette non solo un miglioramento nella qualità di vita delle persone ma anche una riduzione nei costi e nei consumi, il che porterebbe vantaggi sono solo ai singoli individui ma anche alle istituzioni.

Le persone si possono spostare in vari modi: a piedi, in auto, in treno, attraverso i mezzi pubblici o in aereo. Ciascuna rete di trasporto ha dei limiti, superati i quali si possono verificare eventi indesiderati come le congestioni. Il concetto di congestione in letteratura è spesso associato al concetto di transizione di fase: essa si verifica quando la densità degli agenti nella rete diventa estremamente elevata e la velocità media degli stessi diventa molto minore della velocità media possibile in condizioni di flusso libero [4]. Questa condizione è visualizzabile nel *Macroscopic Fundamental Diagram* (MFD) del network (nel Capitolo 1.2 si discuterà meglio degli MFD): si può infatti osservare che la curva che descrive il flusso in funzione della densità tende rapidamente a zero quando

la densità aumenta oltre una certa soglia. È quindi interessante andare a studiare quali sono gli indicatori di una congestione e come si può agire per prevenirla e/o gestirla nel caso in cui non sia possibile evitarla. A tal proposito sono stati pensati dei modelli matematici in grado di descrivere le proprietà del traffico stradale. A seconda del tipo di proprietà (macroscopiche o microscopiche) prese in esame esistono diversi tipi di modelli [5].

Il modello di Lighthill-Whitham (1955), un esempio di modello macroscopico continuo [6], si basa sull'assunzione che il numero di veicoli si conservi e sul fatto che esista un'equazione di stato che lega il flusso del traffico e la sua densità. Un'implicazione di questa ipotesi è che anche i più piccoli cambiamenti nel flusso si propagano in senso contrario (rispetto alla direzione del traffico) sotto forma di *kinematic waves* ("onde cinematiche"). Le onde cinematiche, se considerate assieme, vanno a costituire le cosiddette "*shock waves* cinematiche" in presenza delle quali si hanno rapidamente variazioni di velocità sufficientemente grandi. Questi fenomeni sono molto comuni nelle strade e si verificano ad esempio in presenza di ingorghi. È interessante notare che modelli di questo tipo si applicano non solo per il flusso di veicoli ma anche nello studio del flusso di pedoni ad alte densità.

Un esempio di modello microscopico, invece, (dove con "microscopico" si intende il fatto che si occupa di modellizzare il comportamento di ciascun veicolo singolarmente) è rappresentato da quello di Robert Herman, Elliott Montroll ed altri (anni '50) in cui si studia il *car-following*, ovvero in che modo un'auto segua la successiva [7]. Esiste inoltre un modello a "velocità ottimale", sviluppato da Masako e i suoi colleghi (1995) in cui ogni guidatore cerca di raggiungere una velocità ottimale (da qui il nome del modello) che è funzione della distanza tra una macchina e quella davanti a sé [8].

Esistono altri tipi di modelli, detti mesoscopici, che costituiscono il punto di incontro tra quelli macroscopici e quelli microscopici [9]: risolvono la dinamica sulle strade assumendo che tutti i veicoli si comportino allo stesso modo. Tra i modelli mesoscopici più noti ci sono i modelli di cluster e i modelli *gas-kinetic* (anni '60). Modelli di questo tipo rappresentano il traffico così come vengono rappresentati i gas secondo la teoria cinetica: il movimento dei veicoli (o delle particelle che costituiscono il gas) non è trattato considerando ciascun veicolo (particella) singolarmente, bensì si utilizzano distribuzioni di densità e velocità per calcolare le densità e velocità attese [10].

Una volta che si è modellizzato il traffico si può passare alla schematizzazione del network in cui andranno a circolare i veicoli. Un network si compone di link, cioè strade, e nodi, ovvero punti di intersezione, di divergenza o convergenza delle strade. Esistono diversi tipi di incroci che, a seconda di come sono strutturati (semaforici e non, con presenza di rotonde), seguono regole specifiche (ci sono precedenze da rispettare a seconda della segnaletica orizzontale, in un incrocio semaforico non si possono fare in versioni a U ecc.). In questa tesi si tratterà nello specifico di incroci semaforici e di come ottimizzare il loro funzionamento per ridurre i tempi di percorrenza, la possibilità di congestioni (o di ritardarla nel tempo nel caso sia inevitabile) e avere miglioramenti nel flusso dei veicoli

all'aumentare della loro densità nel network.

Nelle città e nelle metropoli le persone quotidianamente sono coinvolte nelle dinamiche del traffico nei loro spostamenti ed è per questo che cercare di evitare e/o ridurre la frequenza di certi fenomeni è diventato oggetto di studio. In particolare in questa tesi verrà affrontato il tema dell'ottimizzazione di una rete semaforica, vale a dire la ricerca di un controllo attuato sui parametri che descrivono i semafori al fine di rendere la circolazione più scorrevole e di ridurre e/o ritardare nel tempo le situazioni di congestione. Per questa analisi è stato pertanto realizzato un algoritmo che, agendo sui parametri propri della rete semaforica, ha mostrato risultati soddisfacenti per quanto riguarda l'aumento di flusso medio della rete e la diminuzione del tempo medio di percorrenza degli agenti nel network. La sua performance è stata studiata attraverso simulazioni di carica adiabatica e di carico di traffico variabile ed è stata confrontata con quella di un algoritmo realizzato precedentemente ([19]) che, rispetto a una situazione in assenza di ottimizzazione, non era in grado di apportare sostanziali modifiche al flusso del network.

Il contenuto di questa tesi è organizzato in cinque capitoli:

1. Nel primo capitolo verranno spiegate le principali grandezze e i modelli (microscopico, macroscopico e mesoscopico) che si utilizzano nell'analisi del traffico stradale.

2. Nel secondo capitolo verrà esposta una trattazione matematica di come modellizzare la dinamica dei componenti di un network (strade, giunzioni) e verrà fatta una breve introduzione sui semafori e sulle tipologie di controllo semaforico.

3. Nel terzo capitolo verranno presentate le due versioni dell'algoritmo di ottimizzazione della rete semaforica: quello originale e quello oggetto di studio per questa tesi. Verrà spiegato il processo di realizzazione del nuovo algoritmo e le ragioni alla base delle modifiche implementate.

4. Nel quarto capitolo verranno presentati i risultati delle simulazioni ottenuti dal confronto dell'ottimizzazione tramite algoritmo originale e tramite la sua nuova versione. Verranno discussi sia i risultati riguardanti le simulazioni di carica adiabatica sia quelli riguardanti le simulazioni di carico di traffico variabile.

5. Nel quinto capitolo si concluderà con i commenti riguardanti il confronto di queste due situazioni (algoritmo originale e nuovo algoritmo) sia per quanto riguarda le simulazioni di carica adiabatica che quelle di carico di traffico variabile.

Capitolo 1

Modelli di traffico

Il traffico può essere considerato come un tipo di sistema a molti corpi di veicoli che interagiscono molto fortemente tra di loro, spesso in situazioni di non equilibrio. Dal momento che il comportamento del traffico è complesso ed articolato, esistono una serie di modelli diversi in grado di mettere in luce aspetti distinti di questo sistema a seconda del tipo e grado di analisi proposto dal modello stesso.

Tra i primi a condurre studi scientifici riguardanti il traffico ci fu Greenshields nel 1935, in seguito nel 1955 Lighthill e Whitham presentarono il noto modello di traffico basato sulla teoria fluido-dinamica. Secondo questo modello, in cui si considera il traffico un fluido 1-dimensionale comprimibile, un ingorgo stradale non è altro che un'onda d'urto (*shock wave*). In seguito a questo modello ne sono stati presentati altri: quello di Prigogine (1960) che utilizza un modello *kinetic-gas* basato sull'equazione di Boltzmann, quello microscopico di Newell (1961) della velocità ottimale, che parte dall'assunzione secondo cui la velocità si adatta con un certo ritardo, e molti altri. Sicuramente l'avvento del computer e gli avanzamenti tecnici e concettuali della fisica moderna hanno fatto sì che le teorie del traffico si potessero sviluppare ulteriormente fino a quelle che conosciamo oggi [11].

Per quanto ciascun modello si possa soffermare su alcuni aspetti caratteristici del traffico, tutti quanti i modelli devono rappresentare alcune caratteristiche comuni: un esempio di questo tipo di caratteristiche comuni può essere il fatto che i veicoli si muovono liberamente a basse densità, mentre ad alte densità sono in uno stato congestionato. Nel passaggio da stato libero a stato congestionato, inoltre, il traffico può mostrare alcuni comportamenti particolari come l'isteresi e la metastabilità, dove per isteresi si intende una situazione in cui, a un dato istante di tempo, i valori di una grandezza funzione di altre grandezze non dipendono unicamente dai valori di queste grandezze al dato istante di tempo ma anche dai valori che hanno assunto ad istanti precedenti e per metastabilità si intende una condizione di equilibrio che non corrisponde a un minimo assoluto, situazione che, invece, che caratterizza un sistema stabile.

Se il sistema è in uno stato stazionario si può misurare la probabilità di passare da

un punto i a un punto j, dove questa probabilità è spesso rappresentata attraverso un rate di transizione indicato in questo modo π_{ij} (il pedice ij sta ad indicare la transizione $j \to i$). Assumendo che una strada abbia flusso massimo ϕ_j^* la condizione di bilancio si scrive

$$\sum_j \pi_{ij}\phi_j^* = d_i\phi_i^*$$

con $d_i = \sum_j \pi_{ji}$ rate uscente.

In caso contrario, in media una strada avrà flusso entrante (o uscente) più grande e quindi si congestionerà (o si svuoterà). Se la rete è omogenea, dal momento che ϕ_i^* e ϕ_j^* sono uguali si ottiene la formula

$$\sum_j \pi_{ij} = \sum_j \pi_{ji} = d_i$$

Concretamente questo vuol dire che l'entità del traffico in entrata e quella in uscita da un nodo sono uguali. Se questa condizione vale, allora la mobilità si evolve in maniera tale da minimizzare la formazione di congestioni sul network. Introducendo altre grandezze quali la popolazione del punto i (n_i) e il flusso di veicoli (ϕ) si può scrivere l'equazione di continuità media in questo modo:

$$\dot{n}_i = \sum_j \pi_{ij}\phi(n_j) - d_i\phi(n_i)$$

1.1 Osservabili principali

Nello studio dei fenomeni relativi al traffico si fa riferimento a poche osservabili macroscopiche: la densità, la velocità, il flusso e i tempi di percorrenza. Queste grandezze sono tra loro collegate, come verrà spiegato più avanti.

- **Densità**: la densità di veicoli in una strada ρ si definisce come

$$\rho(n, l, t) := \frac{n(t)}{l} \tag{1.1}$$

  dove l è la lunghezza della strada e $n(t)$ il numero di veicoli presenti su di essa al tempo t. La sua unità di misura può essere data dai veicoli per chilometro veh/km (dove veh sta per l'inglese *vehicles* cioè veicoli) o, più semplicemente, km^{-1}. La densità è un osservabile importante perché non solo fornisce informazioni relativamente al numero di veicoli che si trovano in una strada ma, durante le situazioni di congestione, è anche un indicatore della lunghezza della coda di veicoli in prossimità di un incrocio.

- **Velocità**: la velocità dei veicoli, generalmente espressa in km/h ma in alcuni casi anche in m/s, potrebbe essere anche una variabile casuale; tuttavia in un network che descrive una città, o anche solo una zona ristretta di una città, il numero di veicoli è sufficientemente elevato da rendere più comodo l'utilizzo della velocità media di ogni strada. La velocità media relativa a una certa strada si definisce come

$$\bar{v}(t) := \frac{\sum_{n(t)} v_n(t)}{n(t)} \tag{1.2}$$

  dove $n(t)$ e $v_1(t)$, ..., $v_n(t)$ sono il numero di veicoli presenti sulla strada al tempo t e le loro velocità, rispettivamente.

- **Flusso**: il flusso di una strada si definisce in questo modo

$$\phi := \rho\, v \tag{1.3}$$

  dove ρ e v sono la densità e la velocità, rispettivamente, come definite nei punti precedenti. Il flusso ha come unità di misura veh/h o, più semplicemente, h^{-1}. In realtà questa definizione puntuale di flusso è concretamente inutilizzabile, pertanto si utilizza una definizione operativa di flusso:

$$\phi := \frac{\#\text{ veicoli che transitano su una determinata strada nel tempo } \Delta t}{\Delta t} \tag{1.4}$$

  dove Δt è un lasso di tempo stabilito.

- **Tempi di percorrenza**: il tempo di percorrenza è il tempo che si impiega a raggiungere A partendo da P, dove P ed A sono due nodi sul network. Essendo questa definizione molto generale, il tempo di percorrenza non dipende da altre variabili ed è forse questo il motivo per cui, tra tutte le osservabili appena citate, questa è quella che maggiormente si utilizza nella quotidianità durante gli spostamenti. Generalmente si cerca di ridurre il più possibile i tempi di percorrenza, anche se viaggiare per meno tempo non è necessariamente indice di un viaggio meno costoso dal punto di vista ambientale o di carburante. Tecnicamente, infatti, per andare a ricercare un risparmio bisognerebbe andare a considerare un certo numero di fattori, come il tipo di percorso fatto, il mezzo utilizzato ecc.

1.2 Diagrammi Fondamentali Macroscopici

Recentemente (Daganzo 2007) è stata avanzata l'idea secondo cui il traffico possa essere modellizzato in maniera dinamica in zone urbane di grandi dimensioni (ad esempio quartieri) se queste ultime possiedono un Diagramma Macroscopico Fondamentale (o MFD, dall'inglese *Macroscopic Fundamental Diagram*) riproducibile che mette in relazione il numero di veicoli circolanti con la velocità media (o flusso medio) e una relazione tra il flusso medio della zona e il flusso in uscita totale. Il concetto degli MFD non è recente, infatti già Godfrey l'aveva introdotto nel 1969 ([12]), ma la verifica della loro esistenza e delle loro caratteristiche invece lo è. Dal momento che mettono in relazione il flusso, la velocità e la densità medi del network, gli MFD sono interessanti da studiare per valutare come agire sulla rete poiché cambiamenti come la riprogrammazione dei semafori o l'aumento percentuale del numero di strade dedicate al trasporto pubblico si riflette sul diagramma stesso [13].

Esistono varie tipologie di MDF a seconda di che grandezze sono presenti negli assi delle ascisse e delle ordinate (densità-flusso, velocità-flusso, etc). Dalla definizione di flusso (Eq.1.3) è evidente come esista una relazione sia tra queste grandezze sia tra i valori medi di queste grandezze [14].

Se si assume di conoscere il numero dei veicoli che transitano da una strada grazie agli appositi strumenti di misurazione, di fatto è come se i flussi ϕ fossero noti (li si ottiene tramite l'Eq.1.4) e quindi si può calcolare l'unica grandezza non nota (che sia la velocità o la densità) attraverso l'Eq.1.3. Gli MFD, inoltre, mostrano un certo numero di punti dalle proprietà significative, ad esempio si può vedere che un flusso nullo ($\phi = 0$) può avvenire in due situazioni molto diverse tra loro:

- Quando non ci sono veicoli la densità è nulla, pertanto anche il flusso lo è. In questo caso la velocità ha un valore solamente teorico e generalmente la si assume pari alla velocità limite v_l, ovvero la velocità massima possibile prevista per quella determinata strada a seconda della sua natura: urbana, provinciale, statale, autostrada etc.

- Quando la densità è così elevata che tutti i veicoli sono costretti a fermarsi (e quindi la loro velocità è nulla) anche il flusso diventa nullo. La densità a cui avviene questo fenomeno è detta densità critica ρ_c.

Tra questi due punti in cui $\phi = 0$ si ha che all'aumentare della densità anche il flusso aumenta dal momento che aumentano i veicoli sulla strada, ma mentre questo accade la velocità comincia a diminuire a causa dell'interazione tra i veicoli. Questa diminuzione è trascurabile fintantoché i flussi e le densità sono medio-bassi, tuttavia al continuo aumentare della densità, la diminuzione tende a diventare significativa. In Fig.1.1 si possono osservare gli andamenti precedentemente decritti.

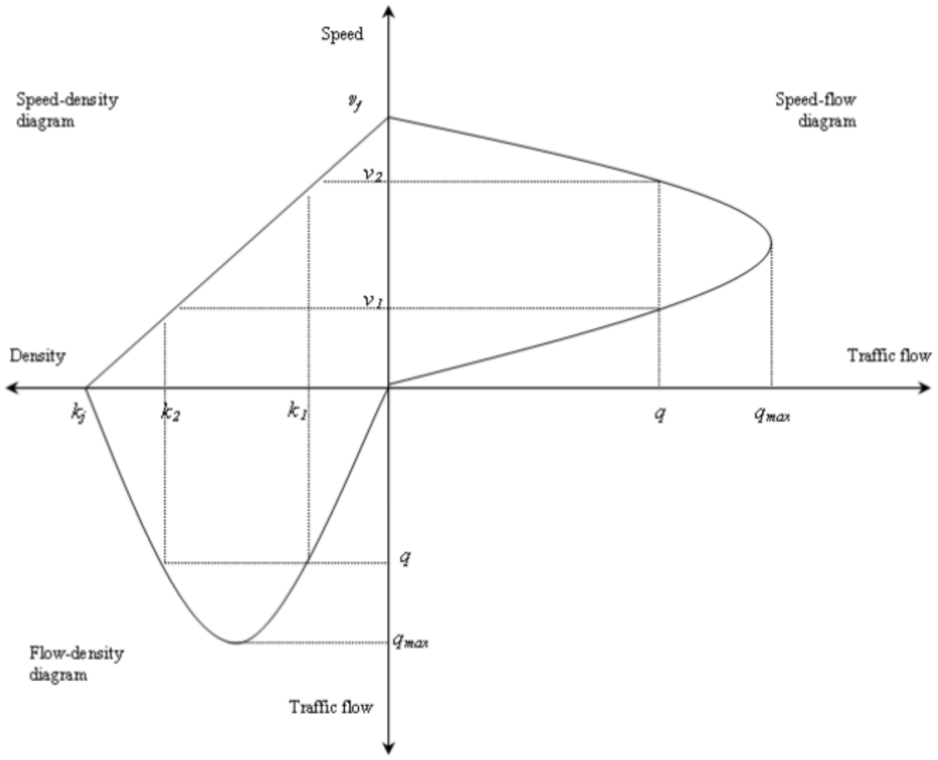

Figura 1.1: Diagramma Fondamentale Macroscopico che mette in relazione velocità, densità e flusso (qui indicate come v, k e q rispettivamente) [14].

1.3 Modelli di code

A seconda del tipo di flusso di traffico i metodi che si possono utilizzare per la sua analisi sono distinti. Innanzitutto esistono due categorie di flussi di traffico: flussi interrotti e ininterrotti.

- Nei flussi di traffico ininterrotti si ha che il flusso è regolato dalle interazioni tra veicoli e tra quelli tra veicoli e le strade. Un esempio di flussi di questo tipo è quello dei veicoli che viaggiano nelle autostrade.
- Nei flussi di traffico interrotti, invece, il flusso è controllato da agenti esterni, ad esempio i semafori; in questo caso le interazioni tra veicoli e tra veicoli e strade hanno un ruolo secondario.

Andando a trattare unicamente dei flussi di traffico ininterrotti e dei modelli di coda, si può dire che in letteratura sono state identificate tre tipologie di modelli di coda a

seconda del livello di dettaglio di analisi del flusso: microscopico, macroscopico e mesoscopico. In sintesi, i modelli microscopici prendono in considerazione ciascun veicolo, quelli macroscopici i flussi (hanno quindi una visione più ampia rispetto a quelli microscopici) e quelli mesoscopici sono una sintesi in un unico modello dei due precedenti. Il principale svantaggio dei modelli microscopici e mesoscopici sta nel fatto che sono estremamente complessi e non risolvibili analiticamente, pertanto la loro risoluzione (via computer) richiede molto tempo. Sotto questo punto di vista i modelli macroscopici sono i più adatti per trovare strategie di controllo, dal momento che il traffico sotto forma di flusso può essere trattato in maniera analitica e questo richiede meno tempo nell'elaborazione dei dati per un computer [14].

1.3.1 Modelli Microscopici

Come anticipato, i modelli microscopici analizzano ciascun veicolo singolarmente e si basano sull'utilizzo delle teorie che studiano il movimento dei veicoli nel traffico. Alcune caratteristiche rilevanti di questi modelli sono:

- Fanno utilizzo di teorie di tipo *car-following*, cioè teorie che descrivono come un veicolo segue il successivo.
- Utilizzano diagrammi spazio-tempo dove il tempo viene generalmente rappresentato sull'asse delle ascisse e lo spazio (ovvero la distanza rispetto a un punto di riferimento) sull'asse delle ordinate. In questo modo si ha che le traiettorie dei veicoli sono rappresentate come nelle curve nel piano, in particolare le traiettorie dei veicoli fermi sono delle linee orizzontali. Come si sa molto bene dalla meccanica classica, dove grafici di questo tipo sono molto comuni, la pendenza di tali curve rappresenta la velocità dei veicoli e in presenza di curve non rettilinee si può capire che il veicolo è andato incontro a situazioni di accelerazione o decelerazione.
- Fanno uso di microsimulazioni attraverso dei software come MicroSim o VISSIM [14].

1.3.2 Modelli Macroscopici

I modelli macroscopici, a differenza dei precedenti, non considerano i veicoli singolarmente ma li raggruppano in flussi. Anche per questo tipo di modelli esistono alcune caratteristiche interessanti da sottolineare:

- Studiano la capacità: attraverso questi modelli si cerca di analizzare la capacità e il livello di servizio di autostrade e altri servizi.
- Utilizzano gli MFD, o Diagrammi Fondamentali Macroscopici, (Capitolo 1.2) per analizzare le relazioni tra densità, velocità e flusso.

- Analizzano le *shock waves* dal momento che la loro formazione è legata a cambiamenti nelle condizioni del traffico e potrebbe essere interessante studiarle per comprendere quando si verificano determinati fenomeni [14].

1.3.3 Modelli Mesoscopici

I modelli mesoscopici rappresentano una sintesi tra i due modelli precedenti in un unico approccio: gli enti del traffico sono descritti con un grado di dettaglio elevato mentre le interazioni tra gli enti sono descritti con un minore livello di dettaglio. Ad esempio un cambio di corsia viene rappresentato da un evento istantaneo per un singolo veicolo (secondo l'approccio dei modelli microscopici) ma la scelta se effettuare il cambio o meno si basa sulle densità e velocità relative della corsia (grandezze tipiche dei modelli macroscopici). Esistono varie tipologie di modelli mesoscopici:

- Modelli di distribuzione degli intervalli: sono modelli che descrivono come si distribuisce la differenza dei tempi di passaggio di due veicoli consecutivi senza, però, mai considerare i due veicoli singolarmente.
- Modelli di cluster: descrivono i cluster, le loro proprietà e la loro formazione. Un cluster è costituito da un gruppo di veicoli che hanno tutti le stesse proprietà. I veicoli appartenenti a un cluster vengono trattati come un'entità unica.
- Modelli di Navier-Stokes: questi modelli descrivono la dinamica delle distribuzioni di velocità e sono il punto di contatto tra il comportamento del singolo veicolo microscopico e l'approccio macroscopico [14].

Capitolo 2

Dinamica di un network e dei suoi elementi

In questo capitolo si propone un approccio per modellizzare dal punto di vista matematico un network stradale con un modello mesoscopico. Le equazioni ottenute saranno poi inserite nella dinamica della simulazione. Si anticipa che il Capitolo 2.2 è presente solo per completezza della trattazione. Quelle equazioni non vengono strettamente utilizzate nella simulazione.

Alla base di una simulazione di traffico c'è sempre un network. Per questioni di praticità verranno presi in esame solo network planari, cioè grafi rappresentabili in uno spazio 2-dimensionale in maniera tale che nessun link si intersechi con un altro. Si considerino N nodi ed M link, si definisce la densità di link in questo modo

$$D := \frac{M}{N}$$

D'ora in avanti si utilizzeranno i termini *giunzioni* (o *intersezioni*) e *strade* per indicare i nodi e i link rispettivamente.

2.1 Dinamica di una strada

In un network lo stato di una strada è strettamente legato alla sua densità ρ, definita secondo l'Eq.1.1. Dalle osservazioni empiriche si osserva che la velocità decresce all'aumentare della densità [15], pertanto una buona e semplice parametrizzazione della velocità come funzione della densità può essere [16]:

$$v(\rho) = v_0\left(1 - \alpha\frac{\rho}{\rho_{max}}\right) \tag{2.1}$$

dove v_0 è la velocità libera (la velocità limite della strada), α ($\in]0, 1]$) è un parametro che fa sì che la velocità sia sempre sempre non negativa e all'interno dell'intervallo di estremi 0 e v_0 (compresi), mentre ρ_{max} è la densità massima della strada.

Definendo il flusso secondo l'Eq.1.3, anch'esso risulta funzione della densità in questo modo:

$$\phi(\rho) = \rho v(\rho) = \rho v_0 \left(1 - \alpha \frac{\rho}{\rho_{max}}\right) \qquad \rho \leq \beta \rho_{max} \tag{2.2}$$

dove β ($\in \left[\frac{1}{2\alpha}, 1\right]$) è un parametro che definisce il flusso medio di traffico durante gli stati congestionati.

Empiricamente si osserva che il flusso di traffico medio tende a un valore costante ϕ^* negli stati congestionati, pertanto l'Eq.2.2 può essere estesa anche a densità maggiori di $\beta\rho_{max}$ in questo modo:

$$\phi(\rho) = \begin{cases} \rho v_0 \left(1 - \alpha \frac{\rho}{\rho_{max}}\right) & \rho \leq \beta \rho_{max} \\ \phi^* = \beta v_0 \left(1 - \alpha \frac{\beta}{\rho_{max}}\right) & \rho > \beta \rho_{max} \end{cases} \tag{2.3}$$

Dalla sua derivata si può vedere che $\phi(\rho)$ ha un massimo locale per un valore di densità critica ρ_c:

$$\rho_c = \frac{\rho_{max}}{2\alpha} \tag{2.4}$$

Per densità maggiori di ρ_c, infatti, il flusso diminuisce e si va a formare la congestione. Noto il flusso in ingresso di una strada (che si assume avere una lunghezza pari a l), indicato come $\phi_{in}(t)$, è possibile scrivere l'equazione che governa la dinamica della strada in questo modo

$$\dot{\rho} = \frac{1}{l}(\phi_{in}(t) - \phi(\rho)) \tag{2.5}$$

Tuttavia all'interno di questa equazione ci si aspetterebbe di trovare un termine che faccia diminuire il flusso in ingresso durante gli stati congestionati (dal momento che si verificano interazioni tra veicoli in corrispondenza degli incroci). Un termine che abbia un effetto simile lo si può ottenere moltiplicando $\phi_{in}(t)$ per una funzione a soglia dipendente dalla densità. In questo modo l'Eq.2.5 diventa

$$\dot{\rho} = \frac{1}{l}\left(\theta\left(1 - \frac{\rho}{\rho_{max}}\right)\phi_{in}(t) - \phi(\rho)\right) \tag{2.6}$$

Per ragioni di convenienza la densità e il tempo (e conseguentemente anche il flusso) possono essere riscalati in questa maniera

$$\rho \to \frac{\rho}{\rho_{max}} \qquad t \to \frac{t v_0}{l} \qquad \phi \to \frac{\phi l}{v_0}$$

dove, come riportato precedentemente, con v_0, l e ρ_{max} si indicano la velocità libera, la lunghezza e la densità massima della strada rispettivamente. L'Eq.2.3 con queste grandezze riscalate diventa

$$\phi(\rho) = \begin{cases} \rho\,(1-\alpha\rho) & \rho \leq \beta \\ \beta\,(1-\alpha\beta) & \rho > \beta \end{cases} \tag{2.7}$$

e il valore massimo che il flusso può assumere si riscrive come

$$\phi_{max} = \rho_c\,(1-\alpha\rho_c) \tag{2.8}$$

e l'Eq.2.6 si trasforma in questo modo

$$\dot{\rho} = \theta(1-\rho)\phi_{in}(t) - \phi(\rho) \tag{2.9}$$

Inoltre si può notare che in condizioni di flusso in ingresso costante pari al più a ϕ_{max} esiste una soluzione stabile dell'Eq.2.9, ρ_{eq} che risolve la seguente equazione

$$\rho_{eq}(1-\alpha\rho_{eq}) = \phi_{in} \tag{2.10}$$

2.2 Dinamiche delle intersezioni

Si ribadisce che questo capitolo è presente solo per completezza della trattazione, le equazioni riportate di seguito non sono state implementate nel modello.

Chiaramente un network non è costituito solamente da strade, ma anche da giunzioni, è pertanto opportuno parametrizzare anche queste giunzioni per poter studiare la dinamica del network complessivamente. Si introduca la matrice bi-stocastica $\pi_{ij}(t)$ che definisce la probabilità con cui il flusso uscente dalla i-esima strada entri nella j-esima strada (chiaramente queste due strade devono essere connesse tra loro) e si ponga $\pi_{ii} = 0$ per tutte le strade i-esime. La dipendenza dal tempo di questa matrice simula le fluttuazioni di flusso agli incroci. Vale la relazione

$$\sum_i \pi_{ij}(t) = \sum_j \pi_{ij}(t) = 1 \tag{2.11}$$

Per semplicità si consideri un network omogeneo, dove tutte le strade hanno la stessa lunghezza e la stessa velocità libera (anche per questa tesi è stato utilizzato un network omogeneo) così che non ci siano problemi nel riscalamento delle variabili.

La dinamica del network può essere descritta a partire dalla dinamica di una strada, utilizzando quindi l'Eq.2.9, tuttavia al posto di $\phi_{in}(t)$ si inserisce un termine che descrive

il flusso in uscita dalle strade connesse alla strada considerata. L'equazione che si ottiene è pertanto questa:

$$\dot{\rho}_i = \sum_j \theta(1-\rho_i)\pi_{ji}(t)\rho_j(t)(1-\alpha\rho_j(t)) - \sum_j \theta(1-\rho_j)\pi_{ij}(t)\rho_i(t)(1-\alpha\rho_i(t)) \quad (2.12)$$

In assenza di fonti esterne valgono queste relazioni

$$\sum_i \dot{\rho}_i(t) = 0 \quad (2.13)$$

$$\sum_i \rho_i(t) = \text{cost} \quad (2.14)$$

L'Eq.2.14 è un integrale del moto il cui valore è sempre minore o uguale a βM, dove con M si indica il numero di strade del network (come riportato all'inizio del capitolo), e che mostra che il numero totale di veicoli si conserva [16].

2.2.1 I semafori

Concettualmente parlando, gli incroci più semplici sono quelli regolati dalla sola segnaletica stradale (e dalle usuali regole di precedenza), tuttavia in città molto popolose o in zone molto trafficate spesso si tende a evitare incroci unicamente regolati dalla segnaletica stradale per ragioni di vario genere come la pericolosità dell'incrocio stesso o la conformazione del network. La presenza di rotonde e/o semafori riduce questa complessità dal momento che entrambi questi elementi con le loro regole sono, in qualche modo, in grado di "auto-regolare" il traffico.

Le rotonde fanno parte del tipo di incroci *intersection-free* (lett. "senza intersezioni") [17] perché, come si può osservare in Fig.2.1 le strade non si incrociano mai esattamente, ma al centro dell'incrocio c'è una zona circolare e/o ellittica attorno a cui girano i veicoli che si immettono (o che escono) dalle (in direzione delle) strade comunicanti. Anche per entrare in rotonda è richiesto di fare attenzione alle macchine vicine e alle giuste precedenze ma, siccome vengono tutte dallo stesso lato, è più semplice che attraversare un incrocio in cui i veicoli potrebbero venire da direzioni distinte.

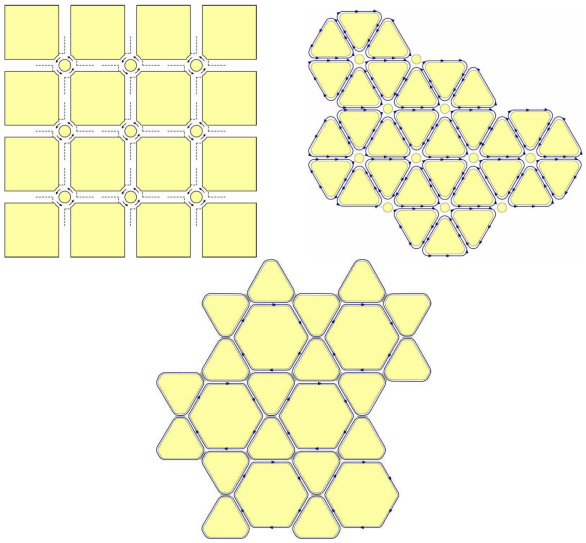

Figura 2.1: Esempi di incroci *intersection-free* realizzabili nelle zone urbane [17].

Sempre in Fig.2.1 sono rappresentati altri tipi di incroci *intersection-free* dove si hanno unicamente immissioni e uscite. In questa maniera i veicoli, nonostante debbano percorrere un po' più di strada, risparmiano tempo di percorrenza e carburante perché questi tipi di incroci non richiedono necessariamente che l'auto si fermi. Pertanto le velocità medie risulteranno superiori e i tempi di percorrenza minori rispetto a strade con incroci standard. Chiaramente questo risparmio lo si ha nel momento il cui il flusso non è troppo basso, altrimenti basterebbe seguire il principio secondo cui il primo che arriva all'incrocio passa o seguire le usuali regole di precedenza secondo cui chi arriva da destra ha diritto di passare per primo.

A differenza delle rotonde, i semafori impongono ai veicoli di fermarsi per un certo tempo (il tempo in cui il semaforo è rosso) e questo aumenta considerevolmente i tempi di percorrenza. Perché un semaforo dovrebbe essere migliore di una rotonda? Chiaramente non esiste un controllo nettamente migliore rispetto agli altri, ma esistono situazioni diverse che richiedono tipi di controlli diversi. Un grande vantaggio dei semafori sta sicuramente nel fatto che modificandone i parametri che lo descrivono (tempo di rosso, tempo di verde, fase) è possibile controllare in modo più preciso l'incrocio di quanto lo si possa fare con una rotonda. Ad esempio è possibile introdurre tempi di verde diversi per le diverse direzioni, una diversificazione non introducibile con una rotonda in cui gli ingressi/uscite non mostrano propriamente una priorità gli uni rispetto agli altri.

Il tipo di controllo più semplice di un semaforo è il *fixed-time control* (lett. "controllo a tempo fissato"), vale a dire un controllo in cui il semaforo diventa rosso e verde (per brevità non si parlerà del tempo di giallo) a intervalli regolari stabiliti a priori. Sono evidenti gli svantaggi di questo tipo di controllo: non prende in considerazione lo stato puntuale del network né si comporta diversamente a seconda del periodo della giornata

e/o dell'anno. Questo tipo di controllo, per quanto semplice, non è per niente efficiente, specialmente alle alte densità a cui si verificano le congestioni. Attualmente si cerca di studiare un tipo di controllo semaforico che sia il più possibile adattabile alla situazione presente del network, infatti questa soluzione è più semplice che cercare di fare predizioni sullo stato futuro del network (predizioni che spesso sono simili ma mai esattamente uguali al reale stato futuro del network) sulla base delle quali regolare i tempi di rosso e verde e la fase di ogni semaforo [18].

Ci sono vari approcci per ottimizzare le prestazioni di una rete semaforica attraverso il controllo dei semafori. La strategia più semplice e più diffusa prevede che il semaforo in esame venga regolato solamente al fine di risolvere le code che si formano a ridosso di quell'intersezione, nella speranza che, risolvendo l'ingorgo locale, l'intera rete ne benefici. Esistono anche approcci più "olistici" nei quali si cerca di trovare un modo per risolvere gli ingorghi locali facendo attenzione alle ripercussioni che l'intervento può avere sugli incroci circostanti, per fare ciò bisogna studiare le proprietà della rete come la formazione di cluster congestionati e inserire all'interno del processo di ottimizzazione le informazioni raccolte su queste proprietà.

Quest'ultimo concetto verrà approfondito nel Capitolo 3 specialmente nel momento in cui verranno paragonati l'algoritmo originale e quello nuovo proposto in questa tesi. Per fare un breve esempio, si consideri una serie di nodi adiacenti congestionati. Se ci si sofferma su ogni singolo nodo si potrebbe trovare una combinazione di rossi e verdi tale da permettere il fluire del traffico di ogni nodo nei nodi adiacenti. Tuttavia, se i nodi adiacenti sono già congestionati, ha senso andare ad aumentare il livello di congestione ulteriormente? Se ciò non ha senso, cosa si può fare altrimenti (sempre ammesso che abbia senso intervenire)? A queste risposte si cercherà di dare risposta nel capitolo sopracitato portando a sostegno della tesi i dati ricavati dalle simulazioni svolte.

Capitolo 3

Ottimizzazione della rete semaforica

Per studiare l'ottimizzazione di una rete semaforica si è scelto di fare riferimento al network planare in Fig.3.1, dove in corrispondenza di ogni nodo è presente un semaforo. Nel network sono presenti 436 strade e 120 nodi. Tutte le strade sono uguali e hanno:

- Lunghezza pari a 2 km.
- Velocità massima pari a 50 km/h.
- Capacità pari a 225 veicoli.

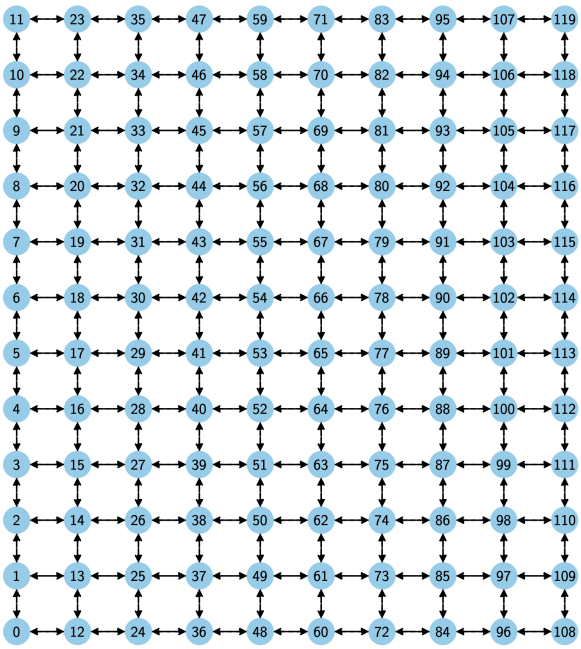

Figura 3.1: Network *Manhattan-like* utilizzato in questa tesi.

La simulazione in cui è stata testata l'efficacia dell'algoritmo è una simulazione in cui, una volta inizializzati i parametri delle strade e dei semafori, all'inizio viene inserito un certo numero di agenti (d'ora in poi ci si riferirà a questo valore con il termine *nAgents*) fissato. Successivamente fintantoché la simulazione non termina, se *nAgents* è positivo, ogni 60 s di simulazione vengono aggiunti uniformemente *nAgents* veicoli. Il valore di *nAgents* subisce variazioni ogni 2400 s di simulazione, infatti, se la differenza tra il numero di agenti nel network nel tempo presente e nel tempo passato (2400 s prima) è minore di 0, si aumenta di un'unità il valore di *nAgents*. Questo tipo di simulazione cerca di riprodurre il fenomeno della "carica adiabatica", ovvero un continuo inserimento di agenti nel network al fine di mantenerne la densità media approssimativamente costante. Quello che si vuole osservare, infatti, è la congestione totale del network e come può influire l'algoritmo di ottimizzazione sul ritardare la congestione e/o mantenere il flusso medio più elevato rispetto a una situazione senza ottimizzazione. Per avere un buon confronto tra le soluzioni è possibile realizzare dei grafici a partire dai dati ottenuti dalla simulazione. Questi grafici verranno presentati nel Capitolo 4.

Nella simulazione, la chiamata alla funzione di ottimizzazione viene effettuata ogni 420 s di simulazione e per tutta la durata della simulazione vengono passati sempre gli stessi parametri alla funzione di ottimizzazione.

Nei capitoli successivi si farà riferimento esplicitamente a oggetti e metodi implementati nel Framework utilizzato per creare il network e la sua dinamica. Per un approfondimento sul Framework si faccia riferimento all'Appendice A.

3.1 Algoritmo originale

L'algoritmo di ottimizzazione originale consiste concretamente in un ciclo su tutti i nodi del *Graph*, in cui se il nodo non è un oggetto *TrafficLight* o se i suoi *Delay* non hanno valore non viene fatto nulla. Per ogni nodo vengono ricavati i valori di tempo di verde e tempo di rosso, a cui, d'ora in poi, ci si riferirà come *greenTime* e *redTime*.

Con *greenTime* e *redTime* non ci si riferisce al tempo di verde e al tempo di rosso rispettivamente, come potrebbe sembrare dal nome delle variabili. Se si osserva la Fig.3.1 si può osservare che ogni semaforo gestisce fino a 4 strade entranti di cui due verticali e due orizzontali. È evidente che, se il semaforo è verde per le strade verticali, le strade orizzontali devono avere il rosso e viceversa. Se si suppone che *greenTime* rappresenti il valore del tempo di verde delle strade verticali, allora esso rappresenta anche il valore del tempo di rosso delle strade orizzontali. Allo stesso modo *redTime* rappresenta il tempo di rosso per le strade verticali e il tempo di verde per le strade orizzontali.

Per prima cosa l'algoritmo valuta la differenza del numero di agenti (sia quelli in attesa che quelli prossimi ad uscire) che si trovano nelle strade verticali e che si trovano nelle strade orizzontali. Si faccia riferimento alla Fig.3.2 per un esempio concreto che potrebbe chiarire i concetti precedenti.

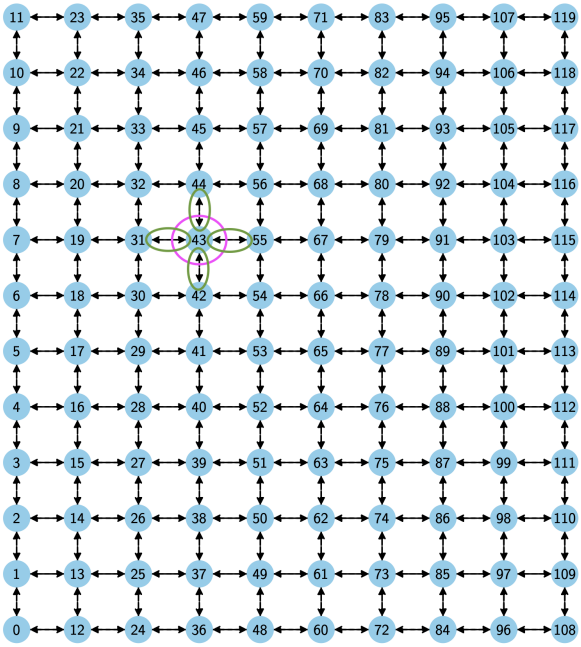

Figura 3.2: Network *Manhattan-like* in cui è stato evidenziato il nodo 43 e le quattro strade entranti rispetto al nodo.

Si consideri il nodo 43 e le quattro strade entranti: le strade verranno indicate tramite i due nodi che le delimitano indicando per primo il nodo di partenza e per secondo il nodo di arrivo. Le strade entranti rispetto al nodo sono: la $44-43$, la $55-43$, la $42-43$ e la $31-43$. L'algoritmo calcola la somma degli agenti che si stanno muovendo verso il nodo 43 dalle due strade disposte verticalmente $44-43$ e $42-43$ (d'ora in poi si supponga che questo valore si chiami $greenSum$) e quella degli agenti che si stanno muovendo verso il nodo 43 dalle due strade disposte orizzontalmente $55-43$ e $31-43$ (d'ora in poi si supponga che questo valore si chiami $redSum$).

Si definiscono inoltre le due variabili $greenQueue$ (sempre relativa alle strade $44-43$ e $42-43$) e $redQueue$ (relativa alle strade $55-43$ e $31-43$) come il numero di agenti in coda per uscire. Chiaramente questo discorso si può estendere a tutte le strade del network dividendo tra quelle verticali e quelle orizzontali.

Definito come $smallest$ il più piccolo tra $greenSum$ e $redSum$, si verifica se vale l'Eq.3.1 o l'Eq.3.2 (ovviamente le due situazioni sono alternative)

$$|greenSum - redSum| < threshold \cdot smallest \tag{3.1}$$

$$|greenSum - redSum| \geq threshold \cdot smallest \tag{3.2}$$

dove $threshold$ è uno dei parametri passati alla funzione *optimizeTrafficLights*.

Se vale l'Eq.3.1 vengono ridefiniti i nuovi valori di *greenTime* e *redTime* come metà del tempo che impiega un semaforo a svolgere un ciclo intero. Un semaforo svolge un ciclo intero quando diventa rosso e poi verde (o viceversa), quindi il tempo di un ciclo è dato dalla somma del tempo di rosso col tempo di verde. Per fare un esempio concreto, si supponga che *redTime* e *greenTime* abbiano i valori $60\,s$ e $40\,s$ rispettivamente. Il tempo di un ciclo è dato da $(60 + 40)\,s = 100\,s$. Se vale l'Eq.3.1 i nuovi *redTime* e *greenTime* valgono entrambi $100/2\,s = 50\,s$.

Se, invece, vale l'Eq.3.2, si definisce a questo punto un oggetto di tipo *Delay* detto *delta* come il più piccolo intero che si avvicini al valore che si ottiene dalla seguente espressione

$$\frac{|greenQueue - redQueue|}{percentage} \tag{3.3}$$

dove *percentage* è l'altro parametro passato alla funzione *optimizeTrafficLights*. Se valgono le opportune condizioni (dove queste condizioni sono state inserite per evitare che un semaforo diventi sempre rosso o sempre verde o che una coda diventi molto più lunga dell'altra) si può incorrere in due situazioni:

- Il valore di *greenTime* viene aumentato di *delta* e il valore di *redTime* viene diminuito di *delta*.
- Il valore di *redTime* viene aumentato di *delta* e il valore di *greenTime* viene diminuito di *delta*.

Per ulteriori dettagli sull'algoritmo e sul suo funzionamento si faccia riferimento a [19].

3.2 Nuovo algoritmo

Lo scopo del miglioramento dell'algoritmo precedente non sta tanto nel ritardare la formazione della congestione nel tempo (la cui formazione è inevitabile visto come è strutturata la simulazione) ma quanto piuttosto nel rendere più scorrevole la rete. Questa maggiore scorrevolezza può essere studiata utilizzando gli MFD ottenuti dalla simulazione, in particolare quello dove si esprime il flusso medio in funzione della densità media. Una maggiore scorrevolezza corrisponde a un flusso medio maggiore a parità del valore di densità media.

Si è partiti dall'algoritmo originale descritto nel capitolo precedente e sono state fatte alcune considerazioni: l'algoritmo originale lavora molto bene su un singolo incrocio tuttavia non prende minimamente in conto i nodi adiacenti e il network in generale. Come anticipato nel Capitolo 2.2.1 non è detto che dalla risoluzione di un ingorgo locale

ne benefici l'interno network, ci si può infatti chiedere se abbia sempre senso che intervenga l'algoritmo dal momento che potrebbero presentarsi due casistiche: ci si potrebbe ridurre a spostare agenti da una zona congestionata a un'altra zona congestionata (fondamentalmente spostando il problema senza risolverlo) oppure potrebbe capitare che, pur fornendo un tempo di verde maggiore, gli agenti non si possano muoversi perché la strada o il nodo in cui devono andare sono congestionati.

Per questo motivo all'interno della nuova versione di questo algoritmo è stato inserito un controllo che tenga conto dello stato complessivo del network, ulteriori dettagli verranno forniti nei capitoli successivi.

3.2.1 Introduzione del controllo sulla densità media

In questo paragrafo si utilizzeranno i termini *greenTime*, *redTime*, *greenQueue*, *redQueue*, *greenSum* e *redSum* con lo stesso significato del capitolo precedente.

Ci sono vari modi con cui interpretare il concetto di cluster all'interno di un network, il più comune dei quali è l'aggregazione di nodi che condividono proprietà morfologiche all'interno del network (ne è un esempio il concetto di componente, che in *network science* indica un insieme di nodi tutti collegati tra loro che non condividono nessun link con i nodi non appartenenti a questo insieme). Tuttavia, all'interno di questa tesi, i cluster a cui si farà riferimento non saranno cluster di nodi, bensì di strade. Quindi con la parola "cluster" si indicheranno tutte quelle strade con valori di flusso, densità o velocità simili tra loro e in particolare, nei capitoli successivi, si farà riferimento a situazioni di congestione.

Le strade all'interno del cluster sono congestionate, vale a dire che la densità di agenti presenti è molto elevata. Si è riflettuto se continuare a far agire l'algoritmo originario anche sui semafori che regolano le strade all'interno del cluster dal momento che fornire un verde di maggiore durata alle strade dove la coda è più lunga può permettere al singolo incrocio di liberarsi ma se l'incrocio è circondato da strade congestionate, concretamente, o si stanno spostando agenti da una zona congestionata a un'altra zona ugualmente congestionata o, nel caso in cui il numero di veicoli nelle strade sia pari alla loro capacità, gli agenti non sono nemmeno in grado di muoversi. In entrambi i casi l'intervento dell'algoritmo non sembrerebbe apportare alcun vantaggio, pertanto, in un primo momento, si è deciso di far intervenire l'algoritmo originale solamente in questi due casi:

- Nel caso in cui valga l'Eq.3.1 e, in questo caso, l'algoritmo agisce senza alcun tipo di modifica.
- Nel caso in cui valga l'Eq.3.2 con l'aggiunta dell'ulteriore condizione che la densità media delle strade uscenti dal nodo considerato sia minore della densità media totale del network. Richiedere che valga quest'ultima condizione equivale a chiedere all'algoritmo di agire solo sulle strade che si trovano fuori dal cluster o, tuttalpiù, sul bordo del cluster.

Impedire all'algoritmo di agire nel caso in cui le strade siano all'interno del cluster di fatto consiste nel richiedere che i tempi di verde e di rosso dei semafori nella zona congestionata rimangano gli stessi (salvo che non intervenga il controllo previsto quando vale l'Eq.3.1, il cui unico scopo è quello di riequilibrare i tempi di rosso e verde in presenza di code abbastanza simili in lunghezza).

Se si assume che la direzione verticale sia quella a cui si riferiscono le grandezze con prefisso "*green*" e quella orizzontale la direzione a cui sono associate tutte le grandezze con prefisso "*red*", a questo stadio l'algoritmo funziona in questa maniera:

1. Se vale l'Eq.3.1 i valori di *greenTime* e *redTime* vengono posti uguali alla metà della durata di un intero ciclo semaforico.

2. Se vale l'Eq.3.2 e le strade uscenti del nodo hanno una densità media minore rispetto alla densità media dell'intero network, allora i valori di *redTime* e *greenTime* vengono modificati secondo quanto segue:

   - Se il numero di agenti in coda nelle strade in direzione verticale è maggiore di quello nelle strade in direzione orizzontale e fintantoché il tempo di verde nella direzione più congestionata (*greenTime*) è minore o uguale di quello nella direzione meno congestionata (*redTime*), vengono apportate queste variazioni: il valore di *greenTime* viene aumentato di *delta* e il valore di *redTime* viene diminuito di *delta*.
   - Se il numero di agenti in coda nelle strade in direzione orizzontale è maggiore di quello nelle strade in direzione verticale e fintantoché il tempo di verde nella direzione più congestionata (*redTime*) è minore o uguale di quello nella direzione meno congestionata (*greenTime*), vengono apportate queste variazioni: il valore di *redTime* viene aumentato di *delta* e il valore di *greenTime* viene diminuito di *delta*.

   dove con *delta* si intende lo stesso valore descritto nel capitolo precedente.

3. Nel caso non siano soddisfatte le condizioni precedenti l'algoritmo non apporta alcuna modifica.

3.2.2 Threshold dinamica

Si è fatta in seguito un'ulteriore considerazione: in corrispondenza del bordo del cluster si ha il punto di contatto della zona congestionata, dove l'algoritmo modificato non agisce, e la zona non congestionata dove l'algoritmo modificato agisce. Si è riflettuto sul fatto che un tipo di controllo diverso effettuato solamente in corrispondenza dei bordi del cluster avrebbe potuto facilitare il passaggio degli agenti dalle strade congestionate a quelle non congestionate. Chiaramente per gli stessi motivi spiegati precedentemente,

questo tipo di controllo viene effettuato solo sul bordo del cluster o, comunque, nella zona interna del cluster ma mantenendosi comunque in prossimità del suo bordo.

Considerando lo schema del funzionamento dell'algoritmo esposto nel Capitolo 3.2.1 e in particolare il punto 2, nel caso di validità dell'Eq.3.2 se la densità media delle strade uscenti dal nodo è maggiore della densità media del network ma minore della densità media del network più una determinata percentuale (il cui valore verrà discusso in seguito) interviene un ulteriore controllo simile al precedente ma con due variazioni degne di nota:

- I valori di *greenTime* e *redTime* non vengono aumentati/diminuiti della stessa quantità, come invece accadeva nei casi precedenti.
- I valori di *greenTime* e *redTime* vengono modificati anche nel caso in cui il valore del tempo di verde della direzione più congestionata sia maggiore del tempo di verde della direzione meno congestionata, qualora permanga una differenza tra le code. Sia l'algoritmo originale che la versione presentata nel capitolo precedente intervengono solo fintantoché, in presenza di una differenza tra il numero di agenti in coda nelle due direzioni, il tempo di verde della direzione più congestionata è minore o uguale del tempo di verde nell'altra direzione. Una volta che il tempo di verde della direzione congestionata è maggiore dell'altro, l'algoritmo non interviene più, anche nel caso sia rimasta una differenza tra il numero di utenti in coda nelle due direzioni. L'idea alla base di questo ulteriore controllo è quella di dare una sorta di *boost* all'effetto originale dell'algoritmo proprio nella zona dove gli agenti passano dalla zona congestionata a quella meno congestionata.

Per quanto riguarda il concetto della percentuale di cui viene aumentata la densità media soglia, si è scelto di introdurre un valore che variasse in base ai dati del network nodo per nodo. È stato definito il rapporto (a cui d'ora in poi ci si riferirà come *ratio*)

$$ratio = \frac{\text{densità media del network}}{\text{densità media delle strade uscenti}} \tag{3.4}$$

Il valore di *ratio* è stato poi utilizzato per definire il parametro *dyn_thresh* (da *dynamic threshold* o "soglia dinamica") in questa maniera:

$$dyn\_thresh = \frac{3}{10}\tanh(ratio) \tag{3.5}$$

La scelta della tangente iperbolica è dettata dal fatto che è la funzione che descrive, appunto, fenomeni a soglia e la scelta di moltiplicare la sua ampiezza per 3/10 è legata al fatto che si è scelto di includere le strade con una densità maggiore della densità media del network al più di un 30% (per evitare di andare ad agire sulle strade troppo interne al cluster, come da premesse).

Se si assume che la direzione verticale sia quella a cui si riferiscono le grandezze con prefisso "*green*" e quella orizzontale la direzione a cui sono associate tutte le grandezze con prefisso "*red*", a questo stadio l'algoritmo funziona in questa maniera:

1. Se vale l'Eq.3.1 succede quanto descritto nel capitolo precedente.

2. Se vale l'Eq.3.2 e le strade uscenti del nodo hanno una densità media minore rispetto alla densità media dell'intero network succede quanto descritto nel capitolo precedente.

3. Se vale l'Eq.3.2 e le strade uscenti del nodo hanno una densità media maggiore rispetto alla densità media dell'intero network ma minore della densità media del network aumentata di *dyn_thresh*, allora i valori di *redTime* e *greenTime* vengono modificati secondo quanto segue:

   - Se il numero di agenti in coda nelle strade in direzione verticale è maggiore di quello nelle strade in direzione orizzontale e fintantoché il tempo di verde nella direzione più congestionata (*greenTime*) è minore o uguale di quello nella direzione meno congestionata (*redTime*), vengono apportate queste variazioni: il valore di *greenTime* viene aumentato di *delta* e il valore di *redTime* viene diminuito di *dyn_thresh*$\times$*delta*.
   - Se il numero di agenti in coda nelle strade in direzione orizzontale è maggiore di quello nelle strade in direzione verticale e fintantoché il tempo di verde nella direzione più congestionata (*redTime*) è minore o uguale di quello nella direzione meno congestionata (*greenTime*), vengono apportate queste variazioni: il valore di *redTime* viene aumentato di *delta* e il valore di *greenTime* viene diminuito di *dyn_thresh*$\times$*delta*.
   - Se il numero di agenti in coda nelle strade in direzione verticale è maggiore di quello nelle strade in direzione orizzontale e se il tempo di verde nella direzione più congestionata (*greenTime*) è maggiore o uguale di quello nella direzione meno congestionata (*redTime*), vengono apportate queste variazioni: il valore di *greenTime* viene aumentato di *dyn_thresh*$\times$*delta* e il valore di *redTime* viene diminuito di *delta*.
   - Se il numero di agenti in coda nelle strade in direzione orizzontale è maggiore di quello nelle strade in direzione verticale e fintantoché il tempo di verde nella direzione più congestionata (*redTime*) è maggiore o uguale di quello nella direzione meno congestionata (*greenTime*), vengono apportate queste variazioni: il valore di *redTime* viene aumentato di *dyn_thresh*$\times$*delta* e il valore di *greenTime* viene diminuito di *delta*.

   dove con *delta* si intende lo stesso valore descritto nel Capitolo 3.1.

4. Come nel caso precedente, se non vengono soddisfatte le condizioni precedenti l'algoritmo non apporta alcuna modifica.

3.2.3 Introduzione della media

Si è fatta un'ulteriore considerazione riguardante il modo di calcolare i valori di *greenSum* e *redSum*. Nell'algoritmo originale corrispondono alla somma del numero totale di agenti (in attesa e non, ottenuto tramite la chiamata al metodo *nAgents* descritto nel'Appendice A.2) per le strade verticali e per le strade orizzontali (rispetto a un determinato nodo). I valori di *greenSum* e *redSum*, quindi, non tengono alcuna traccia della variazione del numero di agenti tra due chiamate consecutive della funzione *optimizeTrafficLights* ma hanno un valore puntuale.

Si è pensato innanzitutto che fosse più interessante avere un'informazione relativa non tanto a tutti i veicoli presenti sulla strada, quanto piuttosto al numero di veicoli in attesa sulla strada. Infatti potrebbe verificarsi la situazione in cui sono presenti molti veicoli su una strada che, siccome il traffico è scorrevole, sono in grado di passare nei nodi adiacenti e, in tal caso, la presenza di tanti veicoli sulla strada non è necessariamente un problema. Per ottenere il numero dei soli agenti in attesa su una strada al posto di *nAgents* è stato utilizzato il metodo *waitingAgents()*.

Inoltre si è ritenuto importante avere un'informazione mediata nel tempo del numero di agenti (in attesa, vista la modifica appena menzionata) tra una chiamata di funzione e l'altra, pertanto è stata aggiunta la *std::unordered_map* (Appendice A.3) tra le variabili interne della classe *Dynamics* per tenere traccia del numero di agenti in attesa.

Capitolo 4

Risultati

Come anticipato precedentemente, l'obiettivo di questa tesi è la modifica dell'algoritmo di partenza al fine di rendere la rete più scorrevole. La scorrevolezza della rete è studiabile attraverso gli MFD del network e, in particolare, attraverso quello che vede messi in relazione il flusso medio e la densità media. Una maggiore scorrevolezza corrisponde a un flusso medio maggiore a parità di densità media. L'algoritmo precedente era in grado di ritardare nel tempo la congestione, tuttavia, rispetto a una situazione priva di ottimizzazione, non era in grado di aumentare il flusso della rete.

L'algoritmo descritto nel Capitolo 3.2, invece, è in grado di rendere la rete più scorrevole a patto che il numero di agenti inserito all'inizio della simulazione (l'*nAgents* a cui si fa riferimento all'inizio del Capitolo 3) sia minore di 315. Di seguito in Fig.4.1 e in Fig.4.2 viene riportato il MFD (quello flusso-densità) relativo ai casi in cui l'algoritmo funziona. Tutte le simulazioni sono state effettuate considerando che la chiamata alla funzione *optimizeTrafficLights* avviene ogni 420 s e la *threshold* è del 15%.

Per ulteriori dettagli sul funzionamento dell'algoritmo in simulazioni con altri parametri si faccia riferimento all'Appendice C.

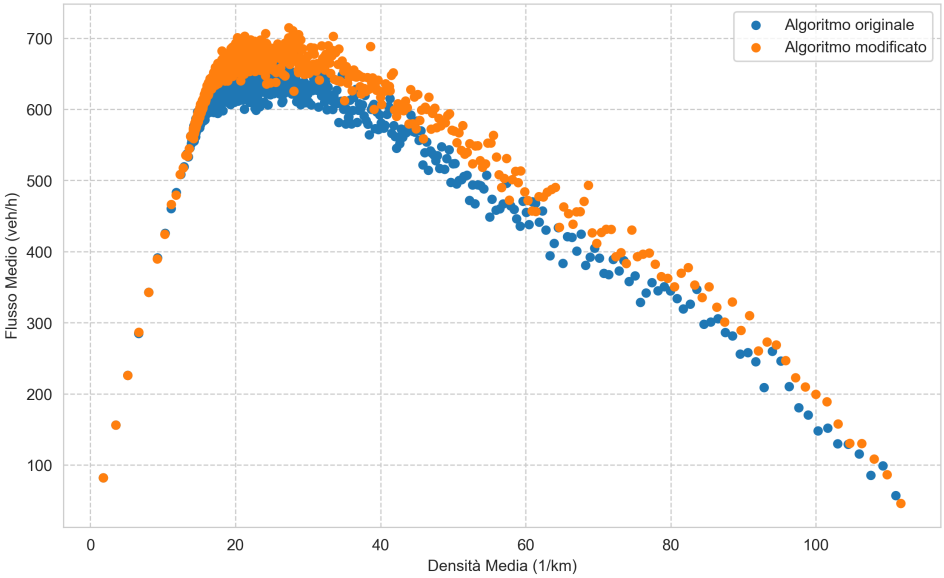

Figura 4.1: MFD flusso-densità per la coppia di simulazioni in cui il valore *nAgents* è di 315 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s.

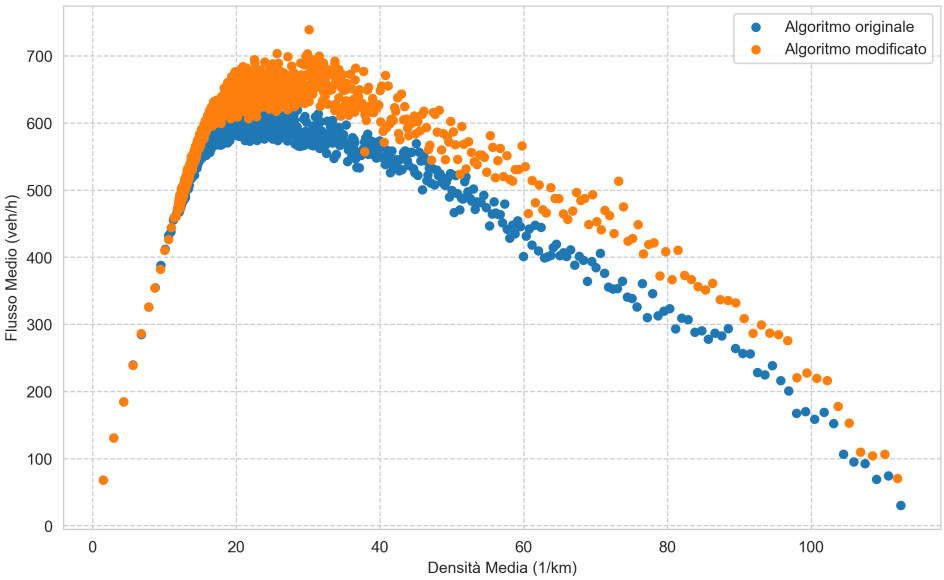

Figura 4.2: MFD flusso-densità per la coppia di simulazioni in cui il valore *nAgents* è di 265 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 80 s e deviazione standard 30 s.

In entrambi i grafici si può osservare la differenza nel valore di flusso medio tra le due situazioni a parità di densità. Il plot arancione rappresenta i valori di flusso medio al variare della densità nel caso della simulazione in cui l'ottimizzazione avviene tramite l'algoritmo modificato mentre il plot azzurro rappresenta i valori di flusso medio nel caso della simulazione in cui l'ottimizzazione avviene tramite l'algoritmo originale. In entrambi i grafici è evidente come i punti del plot arancione, a parità di ascissa, abbiano ordinata maggiore rispetto a quelli del plot azzurro.

Viene inoltre riportato il grafico (Fig.4.3 e Fig.4.4) che mostra l'andamento della densità media del network in funzione del tempo sia nel caso della simulazione ottimizzata tramite l'algoritmo originale e sia nel caso di quella ottimizzata tramite l'algoritmo modificato (curva blu e curva arancione). Inoltre, sempre in funzione del tempo, è stato riportato, per entrambe le simulazioni, il numero di semafori che risultano essere stati modificati in maniera significativa dai due algoritmi (curva verde e curva rossa). Questo è stato possibile grazie all'introduzione della variabile *modTime* (Appendice A.1.2) che tiene conto dell'entità della modifica attuata sui tempi di rosso e verde. Ai fini della realizzazione di questo grafico i semafori con *modTime* uguale a 1 o 2 sono stati considerati come "modificati" (senza fare distinzione tra le due casistiche spiegate in Appendice A.1.2) mentre i semafori con *modTime* uguale a 0 sono stati considerati come "non modificati".

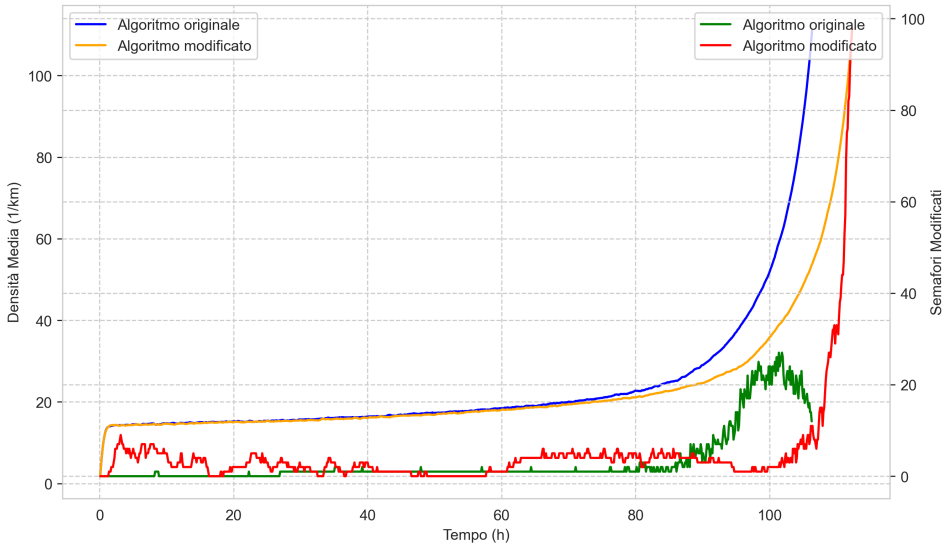

Figura 4.3: Grafico densità-tempo per la coppia di simulazioni in cui il valore di *nAgents* è di 315 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s. Viene inoltre riportata la quantità di semafori "modificati" al variare del tempo per entrambe le simulazioni.

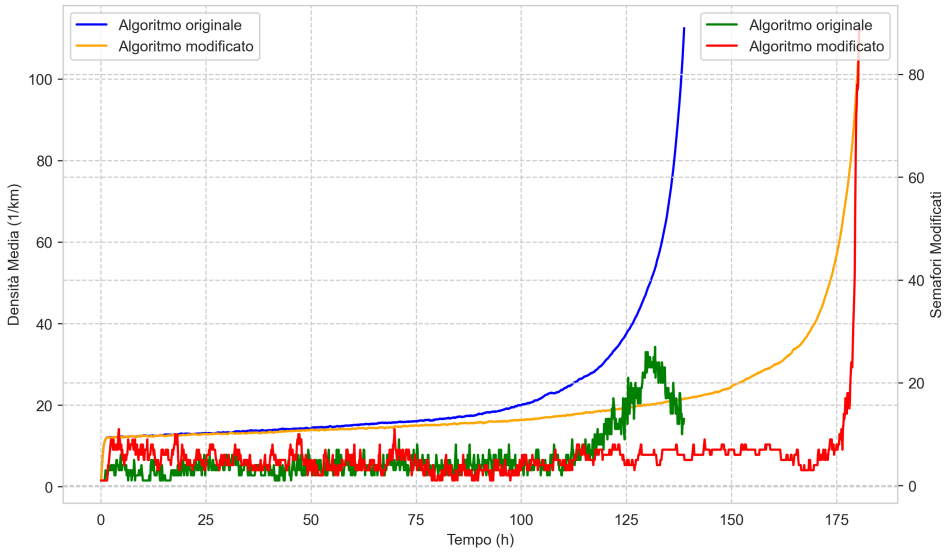

Figura 4.4: Grafico densità-tempo per la coppia di simulazioni in cui il valore di *nAgents* è di 265 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 80 s e deviazione standard 30 s. Viene inoltre riportata la quantità di semafori "modificati" al variare del tempo per entrambe le simulazioni.

In entrambi i grafici si può osservare che, nel momento in cui la densità aumenta rapidamente, in entrambe le simulazioni il numero di semafori i cui parametri sono soggetti a modifiche importanti aumenta. Tuttavia si può notare che, nella simulazione in cui l'ottimizzazione è realizzata tramite l'algoritmo originale, il numero di "semafori modificati" tende a diminuire (dopo aver raggiunto un massimo) nel momento in cui il network si congestiona completamente. Nella simulazione in cui l'ottimizzazione è realizzata tramite il nuovo algoritmo, invece, questo non accade, anzi il numero di "semafori modificati" continua ad aumentare.

Oltre a ciò si può osservare come l'algoritmo modificato impatti positivamente sui tempi di percorrenza medi (vale a dire il tempo medio che gli agenti passano nel network) riducendoli. Di seguito (Fig.4.5 e Fig.4.6) vengono riportati i grafici in cui si confronta l'andamento dei tempi di percorrenza in funzione del tempo nel caso dell'ottimizzazione effettuata dall'algoritmo originale e nel caso di quella effettuata dall'algoritmo modificato. E in Fig.4.7 e Fig.4.8 vengono riportati i grafici dove il tempo di percorrenza medio viene espresso in funzione della densità media del network. Anche in questo caso si può osservare come l'algoritmo modificato riduca, a parità di densità, il tempo medio di permanenza degli agenti nel network rispetto all'algoritmo originale.

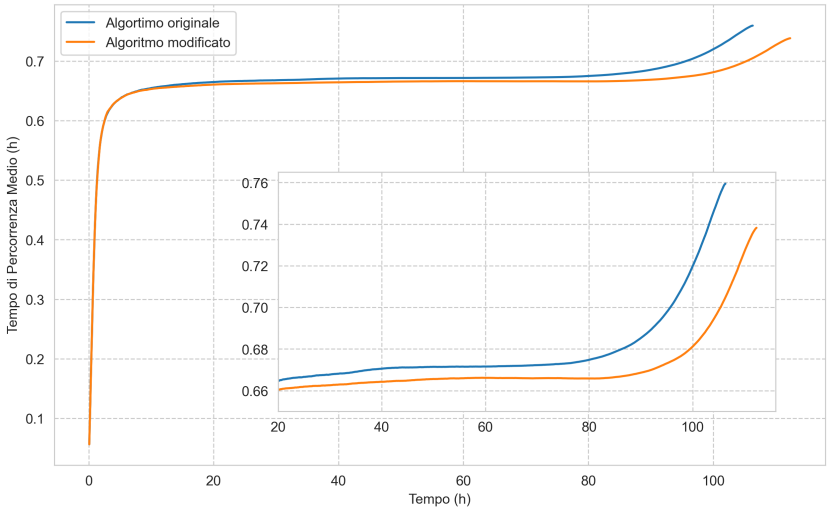

Figura 4.5: Grafico tempo di percorrenza-tempo per la coppia di simulazioni in cui il valore di *nAgents* è di 315 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s. Viene riportato un ingrandimento della curva per visualizzare meglio la differenza tra i tempi di percorrenza.

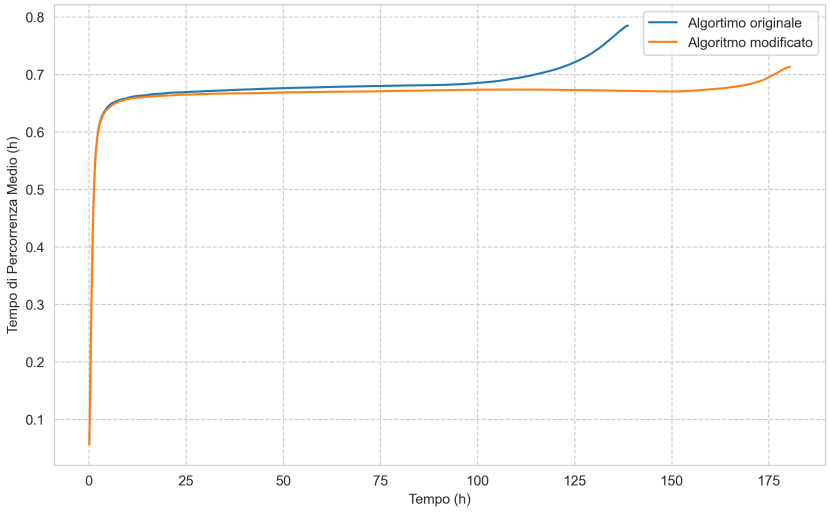

Figura 4.6: Grafico tempo di percorrenza-tempo per la coppia di simulazioni in cui il valore di *nAgents* è di 265 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 80 s e deviazione standard 30 s.

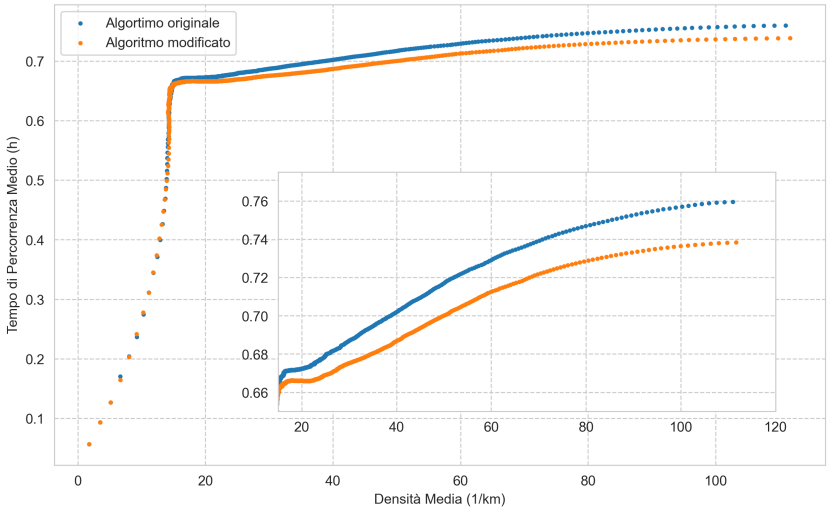

Figura 4.7: Grafico tempo di percorrenza-densità per la coppia di simulazioni in cui il valore di *nAgents* è di 315 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s. Viene riportato un ingrandimento della curva per visualizzare meglio la differenza tra i tempi di percorrenza.

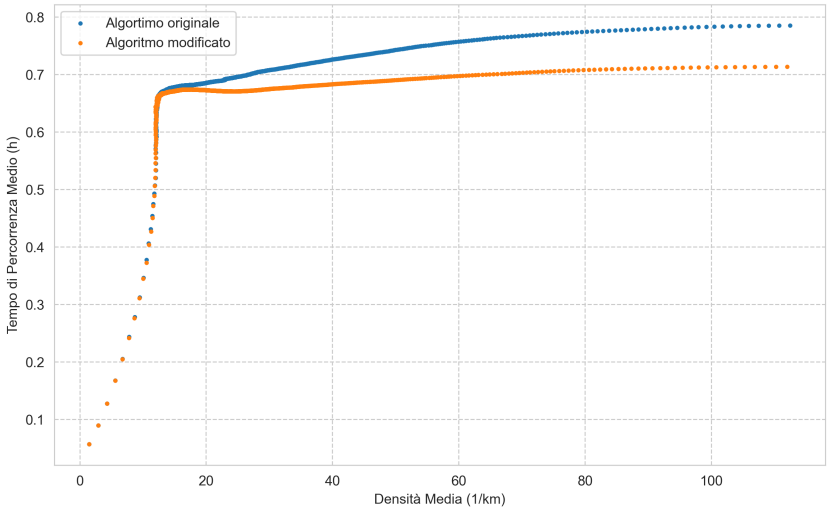

Figura 4.8: Grafico tempo di percorrenza-densità per la coppia di simulazioni in cui il valore di *nAgents* è di 265 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 80 s e deviazione standard 30 s.

4.1 Carico di rete variabile

A questo punto è naturale chiedersi in quale situazione di carico l'algoritmo performi meglio, ovvero in quale situazione di carico l'algoritmo entri in gioco e influisca maggiormente. Capire questo è importante perché permette di comprendere se l'algoritmo è più utile per risolvere congestioni o per rendere il network più scorrevole a carichi più bassi.

In queste simulazioni il network non è stato caricato fino alla completa congestione, ma si è cercato di mantenere il carico costante nelle tre situazioni seguenti: carico basso, carico medio e carico alto. Considerato che all'interno della simulazione la capacità di trasporto di ciascuna strada è di un veicolo al secondo, la simulazione descritta all'inizio del capitolo è stata modificata per poter studiare separatamente queste tre situazioni:

- **Carico di traffico basso**: le code agli incroci sono, in media, al di sotto del flusso di verde, cioè vuol dire che a ogni ciclo la strada si libera completamente dalla coda.
- **Carico di traffico medio**: le code agli incroci non arrivano mai, in media, alla capacità massima della strada, vale a dire che nel network non sono presenti strade congestionate anche se a ogni ciclo la strada non si libera completamente dalla coda.
- **Carico di traffico alto**: sono presenti strade congestionate nel network.

Per "code agli incroci" si intende il numero di veicoli che devono uscire dalla strada, ottenibile tramite il metodo *queue()* (Appendice A.2).

Anche in questo caso il funzionamento della nuova versione dell'algoritmo è riscontrabile nell'aumento della scorrevolezza della rete, pertanto sono stati riportati gli MFD dove è possibile riscontrare la differenza di flusso medio nel network a parità di densità media (l'entità della differenza varia a seconda del caso considerato). È stato scelto un tempo massimo per la simulazione, pertanto i grafici potrebbero non risultare allineati tra di loro. Da questo non allineamento si può osservare ancora meglio la differenza di flusso a parità di densità.

4.1.1 Carico di traffico basso

Per realizzare quanto è richiesto in questa casistica si è modificato il codice originale in maniera tale che il numero di agenti della rete sia tale da permettere che, istante per istante, il valore medio del numero degli agenti in coda agli incroci su tutto il network sia minore di 15. I parametri della simulazione sono i seguenti:

- Numero di agenti iniziale (*nAgents*) pari a 130.

- Valori di tempo di rosso e di verde distribuiti secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s.

Di seguito il MFD che confronta il flusso medio del network a parità di densità media nelle due simulazioni (Fig.4.9) e il grafico che rappresenta la densità media al variare del tempo (Fig.4.10). In entrambi i grafici si può osservare che i due algoritmi lavorano in maniera simile; seppur non ci sia molta differenza tra i valori, nella simulazione con ottimizzazione tramite algoritmo modificato il flusso medio a parità di densità e la densità media al variare del tempo del network risultano, rispettivamente, lievemente maggiore e lievemente minore rispetto a quelli della simulazione con ottimizzazione tramite algoritmo originale.

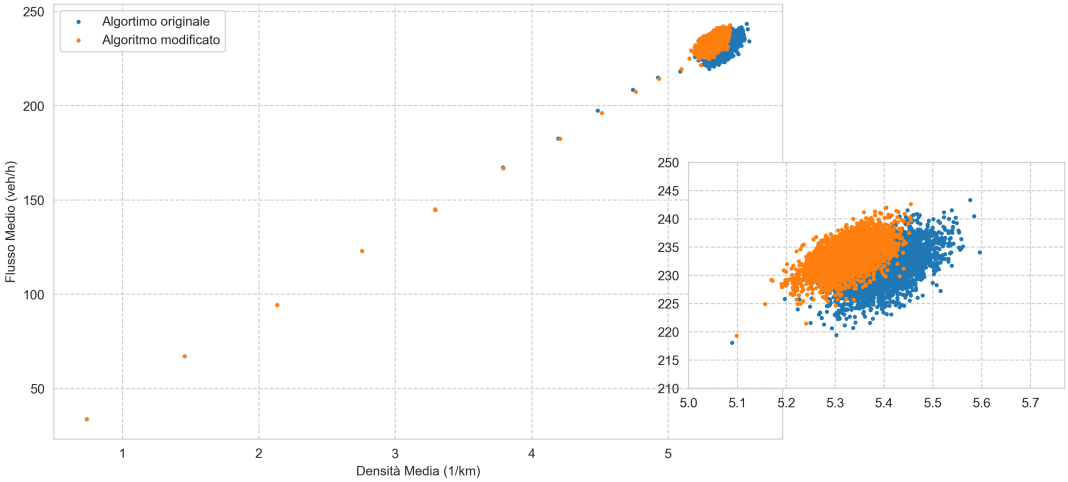

Figura 4.9: MFD flusso-densità per la coppia di simulazioni in cui il valore *nAgents* è di 130 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s. È riportato un ingrandimento del grafico per il confronto dei valori di flusso a parità di densità.

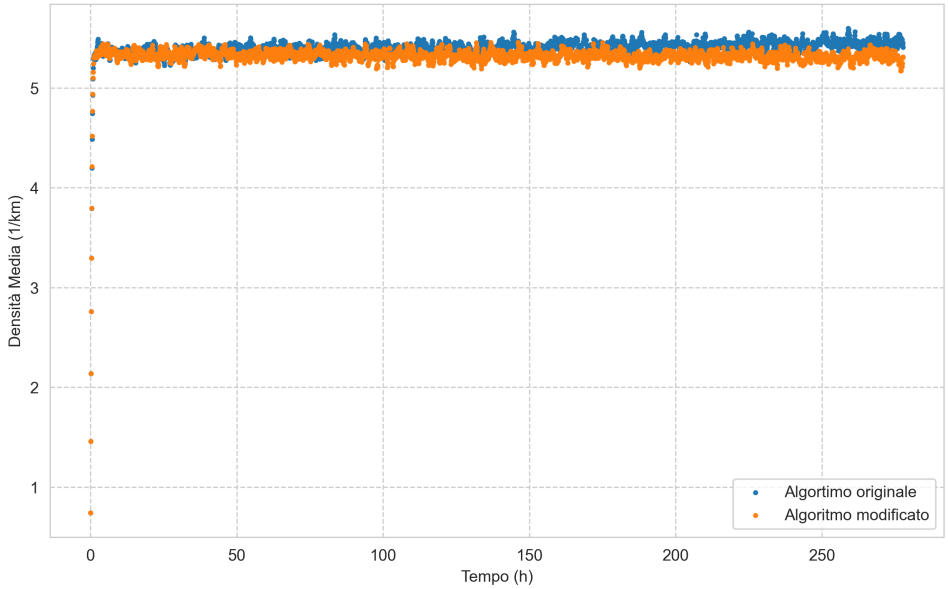

Figura 4.10: Grafico densità-tempo per la coppia di simulazioni in cui il valore *nAgents* è di 130 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s.

4.1.2 Carico di traffico medio

Per realizzare quanto è richiesto in questa casistica si è modificato il codice originale in maniera tale che il numero di agenti della rete sia tale da permettere che, istante per istante, il numero di agenti in coda in ogni strada non superi mai la capacità della strada stessa (in caso contrario la strada sarebbe congestionata). I parametri della simulazione sono i seguenti:

- Numero di agenti iniziale (*nAgents*) pari a 190.
- Valori di tempo di rosso e di verde distribuiti secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s.

Di seguito il MFD che confronta il flusso medio del network a parità di densità media nelle due simulazioni (Fig.4.11) e il grafico che rappresenta la densità media al variare del tempo (Fig.4.12). In entrambi i grafici si può osservare che i due algoritmi lavorano in maniera simile, tuttavia rispetto al caso a carico basso la differenza negli effetti sul flusso medio e sulla densità media è più visibile. L'algoritmo modificato risulta più efficace nell'aumentare la scorrevolezza della rete e nel tenere la densità minore (rispetto all'algoritmo originale).

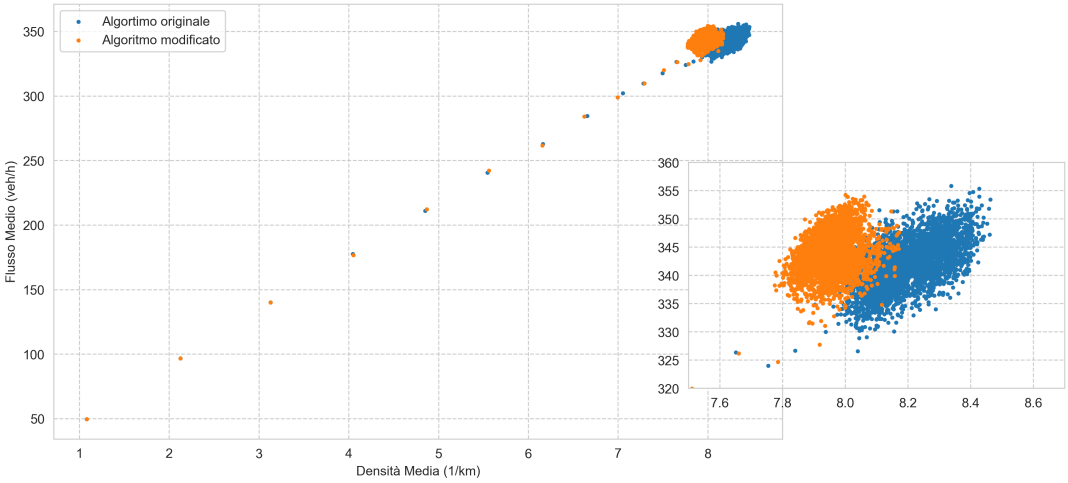

Figura 4.11: MFD flusso-densità per la coppia di simulazioni in cui il valore *nAgents* è di 190 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s. È riportato un ingrandimento del grafico per il confronto dei valori di flusso a parità di densità.

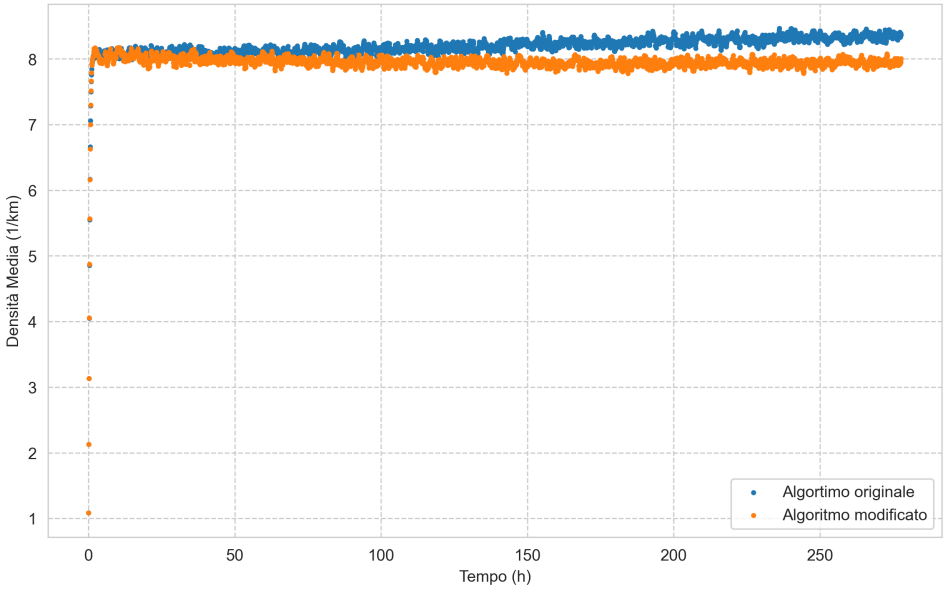

Figura 4.12: Grafico densità-tempo per la coppia di simulazioni in cui il valore *nAgents* è di 190 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s.

4.1.3 Carico di traffico alto

Per realizzare quanto è richiesto in questa casistica si è modificato il codice originale in maniera tale che il numero di agenti della rete sia tale da permettere l'insorgere di congestioni in un certo numero di strade del network, senza congestionarlo completamente. I parametri della simulazione sono i seguenti:

- Numero di agenti iniziale (*nAgents*) pari a 250.
- Valori di tempo di rosso e di verde distribuiti secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s.

Di seguito il MFD che confronta il flusso medio del network a parità di densità media nelle due simulazioni (Fig.4.13) e il grafico che rappresenta la densità media al variare del tempo (Fig.4.14). In quest'ultimo grafico è interessante notare (in base all'andamento della densità media) che nel caso in cui l'ottimizzazione viene realizzata attraverso l'algoritmo originale, il network tende a congestionarsi, mentre, nel caso in cui l'ottimizzazione viene realizzata attraverso l'algoritmo modificato, la densità media del network tende a rimanere costante nel tempo.

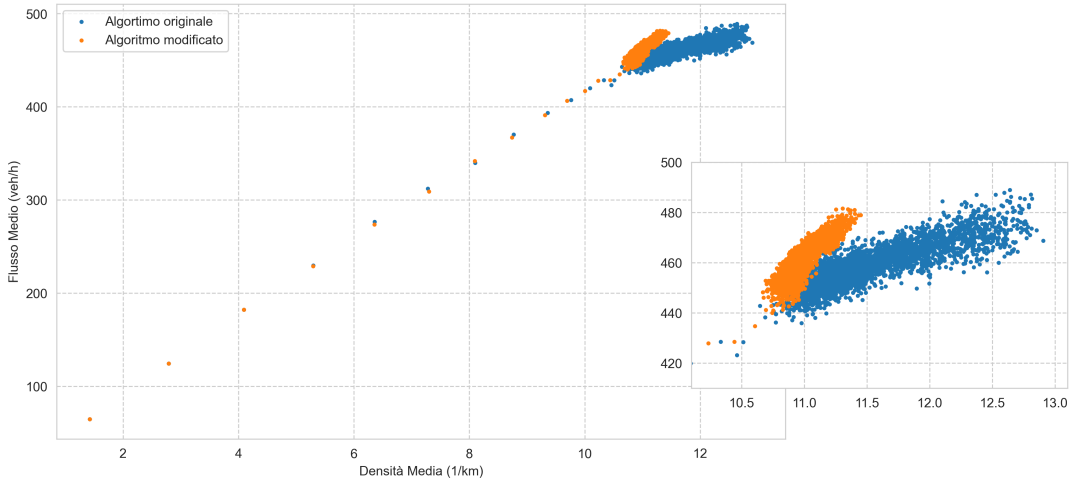

Figura 4.13: MFD flusso-densità per la coppia di simulazioni in cui il valore *nAgents* è di 250 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s. È riportato un ingrandimento del grafico per il confronto dei valori di flusso a parità di densità.

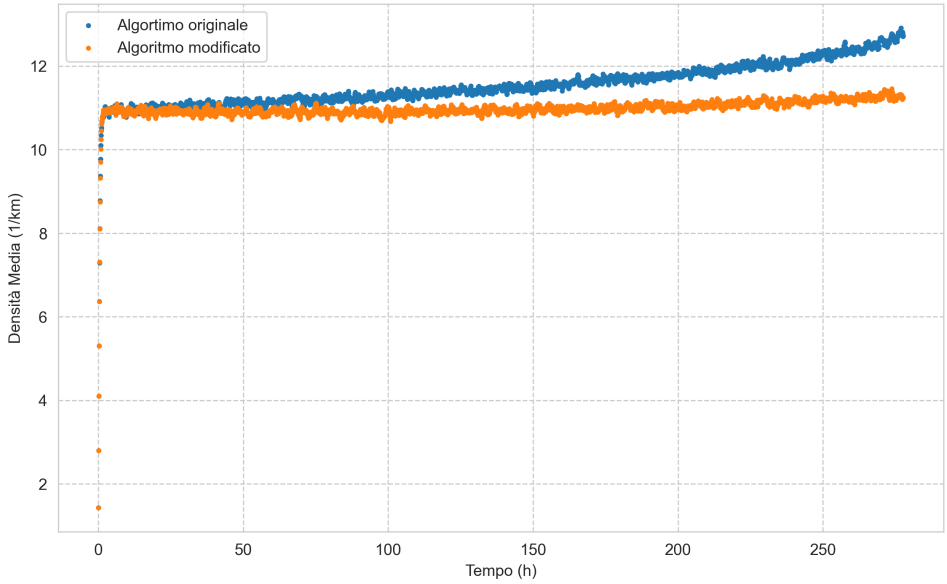

Figura 4.14: Grafico densità-tempo per la coppia di simulazioni in cui il valore *nAgents* è di 250 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s.

Capitolo 5

Conclusioni

Nelle situazioni in cui si è simulata una carica adiabatica del network sono stati messi a confronto gli MFD delle simulazioni (con ottimizzazione tramite algoritmo modificato, tramite algoritmo originale e senza ottimizzazione, per il MFD di quest'ultimo si faccia riferimento all'Appendice D) e si è potuto studiare l'effetto dell'algoritmo modificato nell'aumento del flusso medio del network (rispetto alle altre due situazioni). Questa versione dell'algoritmo fa sì che il network risponda in modo più resiliente alle congestioni, infatti in tutti i grafici si possono notare dei valori più elevati del flusso a pari densità (rispetto all'algoritmo originale).

Inoltre un altro parametro indice dell'efficacia di questo algoritmo è il tempo di percorrenza medio, ovvero il tempo medio che gli agenti passano nel network, anche in questo caso si può notare un'ottima risposta del network in quanto il tempo medio di percorrenza nelle simulazioni con l'algoritmo modificato rimane sempre inferiore (a pari densità) rispetto a quello nelle simulazioni con l'algoritmo originale.

Il motivo del funzionamento dell'algoritmo solo al di sotto di una certa soglia di agenti immessi nel network a inizio simulazione probabilmente non è da ricercarsi tanto nell'algoritmo in sé quanto piuttosto nel fatto che, al di sopra di 315 agenti, la simulazione non sta più riproducendo una carica adiabatica. Al di sopra di tale numero di agenti iniziale, infatti, (questo valore di riferimento è stato ottenuto facendo un certo numero di simulazioni e osservando l'efficacia di entrambi gli algoritmi al variare del numero di agenti immessi nel network a inizio simulazione) sembra che il numero di agenti immesso nel corso della simulazione vari eccessivamente e che il sistema non riesca ad andare all'equilibrio.

Osservando inoltre le modifiche sui tempi di verde e di rosso dei semafori, ottenibili tramite il parametro *modTime* e tramite stampe a schermo di tali valori, si è potuto constatare che l'azione dell'algoritmo modificato può apportare grandi variazioni sia nei tempi di verde e rosso in sé (possono aumentare o diminuire molto rispetto al valore assegnato rispetto a inizio simulazione) sia nella relazione l'uno con l'altro (possono essere anche molto diversi tra loro, ad esempio uno molto piccolo e uno molto grande).

Chiaramente non tutti i valori dei tempi ottenuti sono accettabili in un network reale, tuttavia questo algoritmo può essere uno strumento che, assieme ad ulteriori controlli costruiti ad hoc a seconda del network da studiare di volta in volta, può fornire una guida per l'ottimizzazione della rete.

Per quanto riguarda le situazioni che simulano un carico variabile, si è notato che l'algoritmo interviene maggiormente (nel senso che ha effetti più apprezzabili) quando il carico è alto, infatti quella ottimizzata con l'algoritmo originale vede la densità media del network innalzarsi in un modo simile a quanto accade nelle simulazioni di carica adiabatica mentre quella ottimizzata tramite l'algoritmo modificato la vede mantenersi pressoché costante.

Nelle altre due simulazioni l'effetto dell'algoritmo modificato è meno visibile, anche se comunque presente, infatti la lieve differenza tra le curve (sia quelle flusso-densità che quelle densità-tempo) presente nella simulazione a carico basso diventa più marcata in quella a carico medio. In base ai dati raccolti in questa analisi, l'algoritmo modificato risulta avere effetti positivi in termini dell'aumento della scorrevolezza della rete, tuttavia il maggiore impatto lo ha nella risoluzione (e/o prevenzione) di congestioni.

Appendice A

Dynamical System Framework

La libreria C++ che ha permesso la realizzazione di questa tesi si trova al link `https://github.com/sbaldu/DynamicalSystemFramework` con la relativa documentazione. Tramite questa libreria è stato possibile realizzare un network planare omogeneo di tipo *Manhattan-like* costituito da 436 strade e 120 nodi (si faccia riferimento a quello rappresentato in Fig.A.1) su cui sono stati fatti circolare un certo numero di agenti. Grazie ai metodi implementati è stato possibile andare ad agire direttamente sui parametri della simulazione interessati per ottenere l'ottimizzazione voluta.

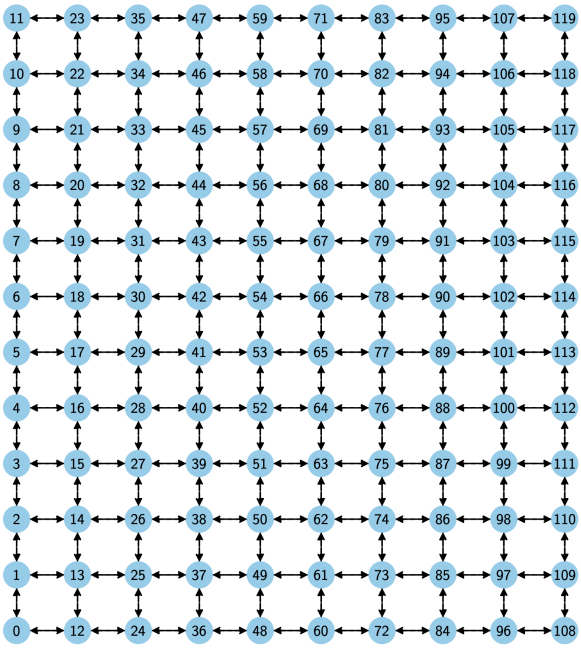

Figura A.1: Il network *Manhattan-like* realizzato tramite il Framework.

Per avere più informazioni riguardo all'implementazione di questa libreria e ai modelli utilizzati per realizzarla si faccia riferimento a [19]. In questa trattazione verranno spiegati solo alcuni elementi della libreria, quelli che sono risultati i più importanti per questa analisi.

Tutte le classi che sono state sviluppate all'interno della libreria andrebbero scritte premettendo *dsm::*, tuttavia, salvo la prima menzione, per ragioni di comodità e brevità i *dsm::* verranno omessi.

A.1 dsm::NodeConcept

Nodi e strade sono gli elementi costitutivi di un network, pertanto hanno due classi specializzate. I nodi sono organizzati a partire da una classe madre base *dsm::NodeConcept*, caratterizzata da alcune variabili interne: il numero identificativo (da qui in avanti chiamato *id*), le coordinate e la capacità. È inoltre corredato di un certo numero di funzioni interne:

- Metodi *getter* e *setter* delle proprie variabili interne, dove con questi termini si intendono funzioni in grado di recuperare il valore presente di una determinata variabile e di impostare a un determinato valore una certa variabile rispettivamente.
- *isFull()*, un metodo che verrà implementato nelle classi figlie come spiegato nel Capitolo A.1.1.
- *isIntersection()*, *isTrafficLight()* e *isRoundabout()*, che se chiamati su un oggetto *NodeConcept* restituiscono *false*, la cui utilità risulterà più evidente quando si parlerà delle classi figlie.

A.1.1 dsm::Node

La prima classe figlia di *NodeConcept* è la classe *dsm::Node*, che corrisponde a un vero e proprio nodo del network, infatti il metodo *isIntersection()* restituisce *true* se chiamato su un oggetto di questo tipo. Oltre alle variabili interne tipiche di un *NodeConcept* un *Node* possiede una *std::multimap* che tiene conto del numero di agenti presenti nel nodo, un *std::set* contenente le strade con che hanno la priorità di passaggio rispetto a quel determinato nodo (questa variabile risulterà molto importante nel descrivere i semafori) e un *counter* del numero di agenti (un *counter* è una variabile che viene incrementata e/o diminuita per tenere traccia della variazione di una determinata grandezza).

Di seguito una lista delle funzioni interne:

- Tutti i metodi della classe madre.
- I *getter* e *setter* delle nuove variabili interne.

- Metodi che vengono utilizzati per aggiungere/togliere agenti (dove per "agenti" si intende veicoli in senso generico, in questo Framework non vengono fatte distinzioni tra tipologie di veicoli distinte) dal nodo.
- Metodi per aggiungere una strada all'insieme di strade con priorità per il nodo.
- *isFull()* che restituisce *true* se il numero di agenti nel nodo è uguale alla capacità del nodo, *false* altrimenti.

A.1.2 dsm::TrafficLight

La classe *dsm::TrafficLight* è una classe figlia della classe *Node*, infatti il metodo *isTrafficLight()* restituisce *true* a differenza dei due casi precedenti. Oltre alle variabili interne ereditate da *Node* (e *NodeConcept*), un oggetto di tipo *TrafficLight* possiede anche una coppia di oggetti di tipo *dsm::Delay* (concretamente un *unsigned integral*) che corrispondono ai tempi di rosso e di verde, un *counter* e una fase.

Le funzioni interne relative a questa classe sono:

- Tutti i metodi della classe *Node* e, conseguentemente, tutti quelli della classe *NodeConcept*.
- I *getter* e i *setter* della coppia di *Delay*, i *getter* del *counter* e i *setter* della fase.
- Un metodo che aumenta il valore del *counter*.
- Due metodi, *isGreen()* e *isGreen(streetId)*, che restituiscono *true* se il semaforo è verde.

A seguito dell'analisi effettuata per questa tesi è stata aggiunta un'altra variabile interna, corredata di *getter*, il cui stato viene aggiornato ogni volta che vengono impostati i nuovi valori dei *Delay* per osservare l'intervento dell'algoritmo e l'entità delle modifiche apportate nei parametri. I valori che posso essere assunti da questa variabile sono:

- 0 se la differenza tra i due *Delay* in valore assoluto è minore di 10 e se nessuno dei due *Delay* è minore di 5.
- 1 se la differenza tra i due *Delay* in valore assoluto si trova all'interno nell'intervallo di estremi 10 e 40 (estremi compresi) e se nessuno dei due *Delay* è minore di 5.
- 2 se la differenza tra i due *Delay* in valore assoluto è maggiore di 40 o se almeno uno dei due *Delay* è minore di 5.

A.1.3 dsm::Roundabout

La classe *dsm::Roundabout* è la classe figlia di *NodeConcept* e non di *Node* (come invece era *TrafficLight*), pertanto i metodi *isIntersection()* e *isRoundabout()* se chiamati su un oggetto *Roundabout* restituiscono *false* e *true* rispettivamente. Questo non è sorprendente dal momento che era già stato anticipato nel Capitolo 2.2.1 che le rotonde fossero degli incroci *intersection-free*. Oltre alle variabili interne della classe madre un oggetto di questo tipo possiede anche un oggetto *dsm::queue* (che corrisponde alla parte pubblica di un oggetto *std::queue*) che tiene traccia di tutti gli agenti che si trovano nella rotonda.

Oltre agli usuali metodi della classe madre, un oggetto *Roundabout* possiede anche:

- Metodi per inserire/togliere un agente dal nodo.
- *isFull()* che, come accadeva nel caso della classe *Node*, restituisce *true* se il numero di agenti nel nodo è uguale alla capacità del nodo, *false* altrimenti.

A.2 dsm::Street

Questa è la classe dell'altro elemento fondamentale del network: le strade. Le variabili interne di un oggetto *dsm::Street* sono:

- Un oggetto *queue* che tiene traccia dei veicoli che stanno uscendo dalla strada.
- Un *std::set* che tiene traccia del numero di veicoli che stanno aspettando nella strada.
- Una coppia di numeri identificativi, che sono gli *id* dei nodi alle estremità della strada.
- La lunghezza della strada.
- La velocità massima permessa nella strada.
- L'angolo a cui è posta la strada.
- L'*id* della strada.
- La capacità della strada.
- La capacità di trasporto, ovvero il numero massimo di agenti che possono essere spostati sulla strada in un *time step* (cioè un intervallo di tempo elementare della simulazione, pari a 1 s).

Di seguito un elenco delle funzioni interne:

- I *getter* e *setter* delle variabili interne.
- *nAgents()* , che restituisce il numero di agenti totale presenti sulla strada, indipendentemente dal fatto che stiano uscendo o che siano in attesa.
- Un metodo che restituisce la densità della strada, espressa come il rapporto tra il numero di agenti totale e la lunghezza della strada.
- Un metodo che restituisce la densità normalizzata (la normalizzazione viene fatta dividendo per la capacità della strada).
- *isFull()* che restituisce *true* se il numero di agenti nella strada è uguale alla sua capacità, *false* altrimenti.
- Metodi che aggiungono/tolgono un agente dalla strada.
- *isSpireStreet()*, che chiamato su un oggetto *Street* generico restituisce *false*, la cui utilità sarà più chiara nel prossimo capitolo.

A.2.1 dsm::SpireStreet

Così come la classe dei nodi era organizzata in classi madre e classi figlie, la stessa cosa avviene per la classe delle strade. La classe *dsm::SpireStreet* è la classe figlia della classe *Street*. Ci si potrebbe chiedere da dove venga il nome "*SpireStreet*" e perché sia stato scelto proprio questo nome. Gli oggetti di questo tipo, a differenza di quelli di tipo *Street*, attraverso l'aggiunta di due variabili interne che sono i *counter* di agenti in ingresso e in uscita, tengono traccia del flusso in ingresso e in uscita di veicoli. È importante notare che l'informazione del numero di agenti in ingresso o in uscita da una strada non è un parametro ottenibile solamente durante una simulazione, ma è ottenibile anche sulle strade reali attraverso l'installazione di strumenti di misura chiamati, per l'appunto, spire (induttive). Si rimanda all'Appendice B per un approfondimento sulle spire.

Oltre ai metodi della classe madre ci sono anche:

- *inputCounts(resetValue)* e *outputCounts(resetValue)* che, rispettivamente, recuperano il numero di veicoli in ingresso e in uscita. Se la variabile booleana *resetValue* è *true*, ogni volta che viene chiamata una di queste due funzioni entrambi i *counter* vengono azzerati, in caso sia *false* invece no.
- *meanFlow()* restituisce il flusso medio della strada dato dalla differenza tra il numero di veicoli in ingresso e quello di veicoli in uscita. Il valore in uscita è positivo se il flusso in entrata è maggiore di quello in uscita. A seguito della chiamata di questa funzione vengono riazzerati entrambi i *counter*.

- A differenza dei soli oggetti *Street*, se *isSpireStreet()* viene chiamata su una *SpireStreet* restituisce *true*.

A.3 dsm::Dynamics

L'ultima classe rilevante per questa analisi è stata la classe *dsm::Dynamics*, che, come si comprende dal nome, si occupa di gestire la dinamica del network durante la simulazione. È inoltre la classe all'interno della quale è stata sviluppata la funzione *optimizeTrafficLights(percentage, threshold)*, di cui si discute approfonditamente nel Capitolo 3.

Tutte le strade e i nodi realizzati tramite le classi descritte nei capitoli precedenti vengono organizzati in un grafo rappresentato da un oggetto di tipo *dsm::Graph* che tiene traccia attraverso una mappa dei nodi, una delle strade e una matrice di adiacenza di come sono disposti reciprocamente questi elementi. Una volta costituito questo *Graph* si inizializza un oggetto di tipo *Dynamics* passandogli il *Graph*. Siccome le variabili interne e i metodi di questo oggetto sono molto numerosi, invece di fare un elenco comprendendoli tutti quanti, si citeranno solo quelli più importanti ai fini di questa tesi.

Di seguito vengono elencate le principali variabili interne dell'oggetto *Dynamics*:

- Una *std::unordered_map* che contiene tutti gli itinerari che gli agenti possono seguire.
- Una *std::map* che tiene traccia di tutti gli agenti nel grafo.
- Un oggetto di tipo *dsm::TimePoint* (*long long unsigned int*) che tiene traccia dello scorrere del tempo durante la simulazione.
- Il grafo.

Inoltre a seguito dell'analisi effettuata per questa tesi è stata aggiunta una variabile interna, una *std::unordered_map* che tenga traccia del numero di agenti in coda al variare del tempo. Il motivo di questa aggiunta viene commentato nel Capitolo 3.2.

Tra i metodi principali ci sono:

- *evolve(reinsertAgents)* che si occupa dell'evoluzione della dinamica chiamando i metodi di evoluzione dei singoli componenti del grafo. In base al valore *true* o *false* della variabile booleana *reinsertAgents* si ha che gli agenti che escono dal network possono essere reinseriti o meno nella simulazione.
- I metodi che si occupano di aggiungere/rimuovere agenti dal network. Esistono vari metodi per aggiungere gli agenti, a seconda del metodo chiamato sarà necessario passare alla funzione elementi distinti e si potranno ottenere effetti distinti (ad esempio esiste un metodo per aggiungere gli agenti uniformemente).

- I metodi che restituiscono i valori medi e le deviazioni standard della velocità media, della densità media, flusso medio delle strade del network, del flusso in ingresso/uscita di una *SpireStreet* e del tempo medio di percorrenza degli agenti del network.
- *optimizeTrafficLights(percentage, threshold)*, la funzione che si occupa dell'ottimizzazione della rete semaforica.

Appendice B

Le spire induttive

Una spira induttiva è, fondamentalmente, un avvolgimento di filo elettrico (come le usuali spire viste in Elettromagnetismo nei circuiti) disposto solitamente in una forma quadrata o rettangolare di lato circa di $2 - 3\,m$. Il sistema di misura è costituito da una o più spire posizionate in corrispondenza della carreggiata e collegate ad un apparecchio rilevatore che si trova in prossimità ai margini della strada. La misurazione del flusso veicolare avviene sfruttando la legge di Faraday-Neumann-Lenz riportata in forma integrale nell'Eq.B.1 e in forma differenziale nell'Eq.B.2

$$\mathcal{E}_{indotta} = \int_{\Sigma(\Gamma)} \mathbf{B} \cdot \mathbf{n}\, \mathrm{d}\Sigma = -\frac{\mathrm{d}\Phi(\mathbf{B})}{\mathrm{d}t} \tag{B.1}$$

$$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t} \tag{B.2}$$

Il filo che costituisce la spira viene percorso dalla corrente grazie alla presenza di un generatore e si genera un campo magnetico. Quando un veicolo (che è costituito in gran parte da metallo) transita sopra il punto dove è posizionata la spira si ha una variazione di campo magnetico che risulta in una variazione della corrente circolante nella spira. Questa variazione produce un segnale elettrico che consente la segnalazione della presenza del veicolo e permette che si possano contare i mezzi che circolano su una strada.

Qualcuno si potrebbe chiedere se è possibile riconoscere dove sono posizionate queste spire e la risposta è affermativa, si può vedere dove sono state posizionate. Di seguito in Fig.B.1 viene riportato un esempio di installazione di questi strumenti di misura.

Figura B.1: Manto stradale dove è possibile vedere dove sono state posizionate le spire [21].

Visto il loro posizionamento è intuibile che l'installazione e la manutenzione (quest'ultima necessaria per garantire l'affidabilità del sensore) siano costose specialmente in termini del disturbo del traffico (è necessario chiudere la strada). Pertanto si stanno sviluppando altri strumenti di misurazione del flusso di traffico [22].

Appendice C

Ulteriori casi di funzionamento del nuovo algoritmo

Di seguito verranno presentati ulteriori casi di funzionamento dell'algoritmo descritto nel Capitolo 3.2. Verranno presentati cinque esempi (che differiscono tra di loro solo per il valore del seed) per tre casistiche di funzionamento, dove ogni casistica ha un numero di agenti differente e i tempi di verde e di rosso dei semafori a inizio simulazione sono stati posti secondo una distribuzione gaussiana di media e deviazione differente.

Con "M/N" si intende che i valori di rosso e di verde dei semafori sono stati stabiliti secondo una distribuzione gaussiana di media M e deviazione standard N (entrambe espresse in secondi).

C.1 315 agenti 60/10

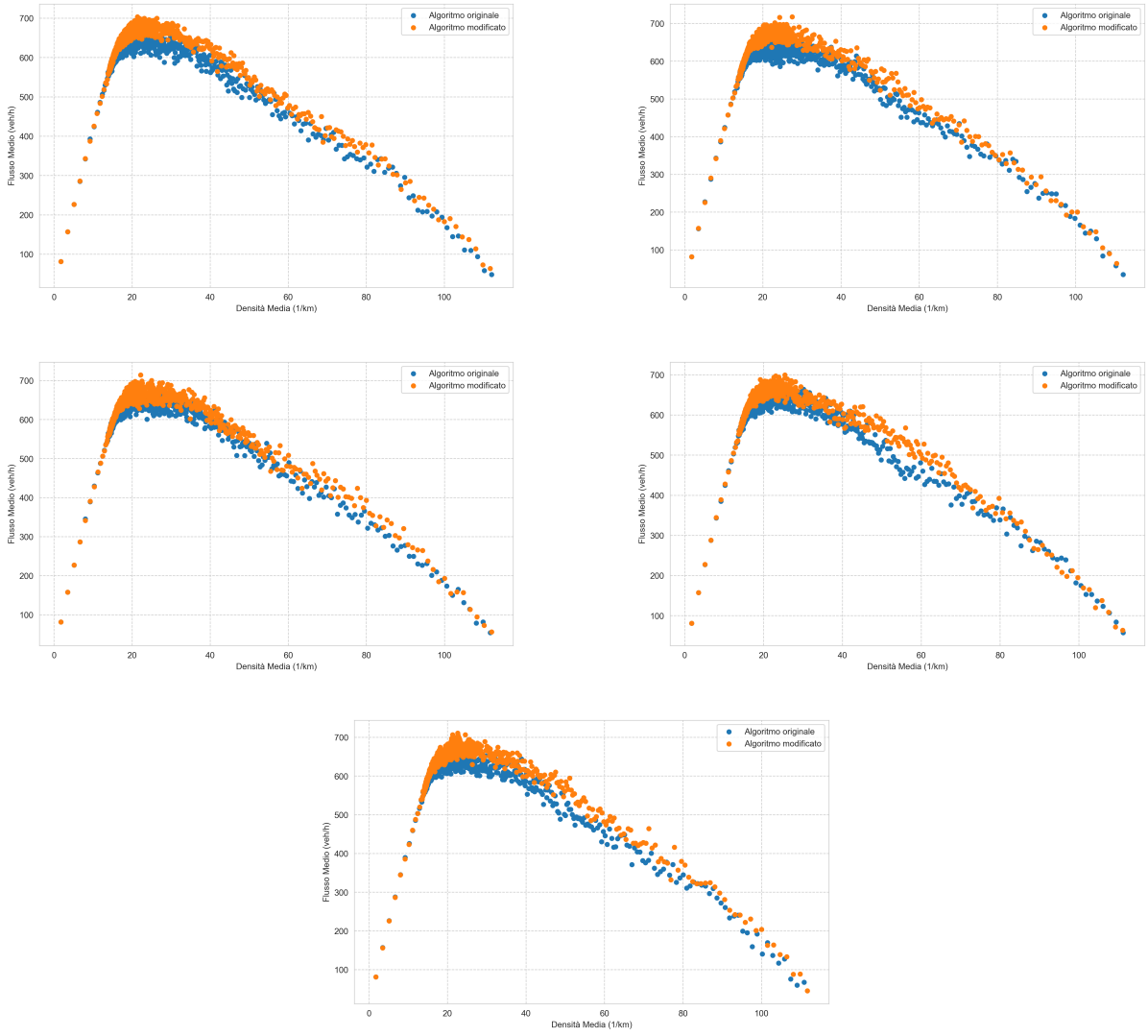

Figura C.1: MFD flusso-densità per le simulazioni in cui il valore *nAgents* è di 315 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s. Tutte le simulazioni possiedono gli stessi parametri, ad eccezione del seed.

C.2 265 agenti 80/30

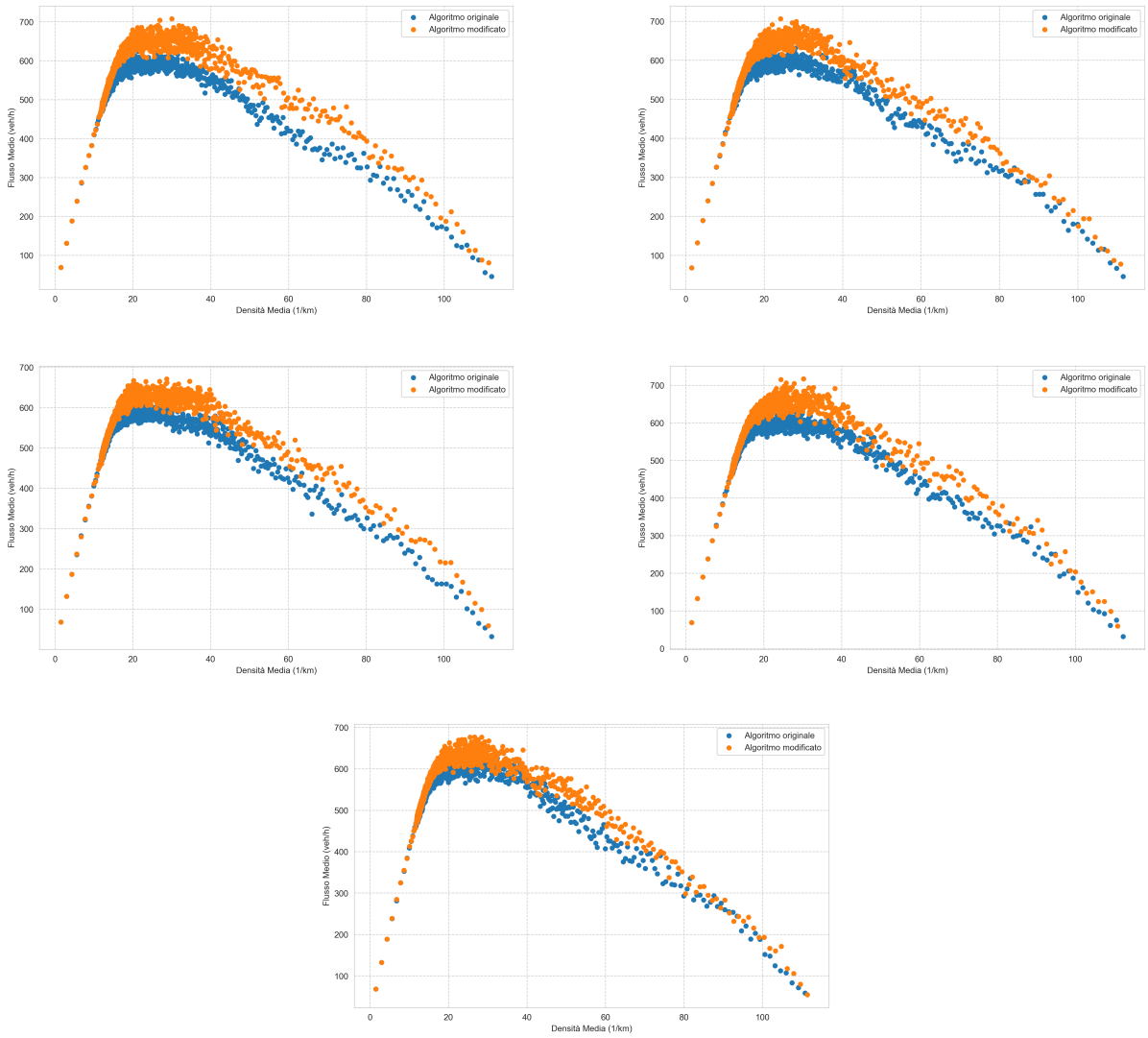

Figura C.2: MFD flusso-densità per le simulazioni in cui il valore *nAgents* è di 265 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 80 s e deviazione standard 30 s. Tutte le simulazioni possiedono gli stessi parametri, ad eccezione del seed.

C.3 215 agenti 100/20

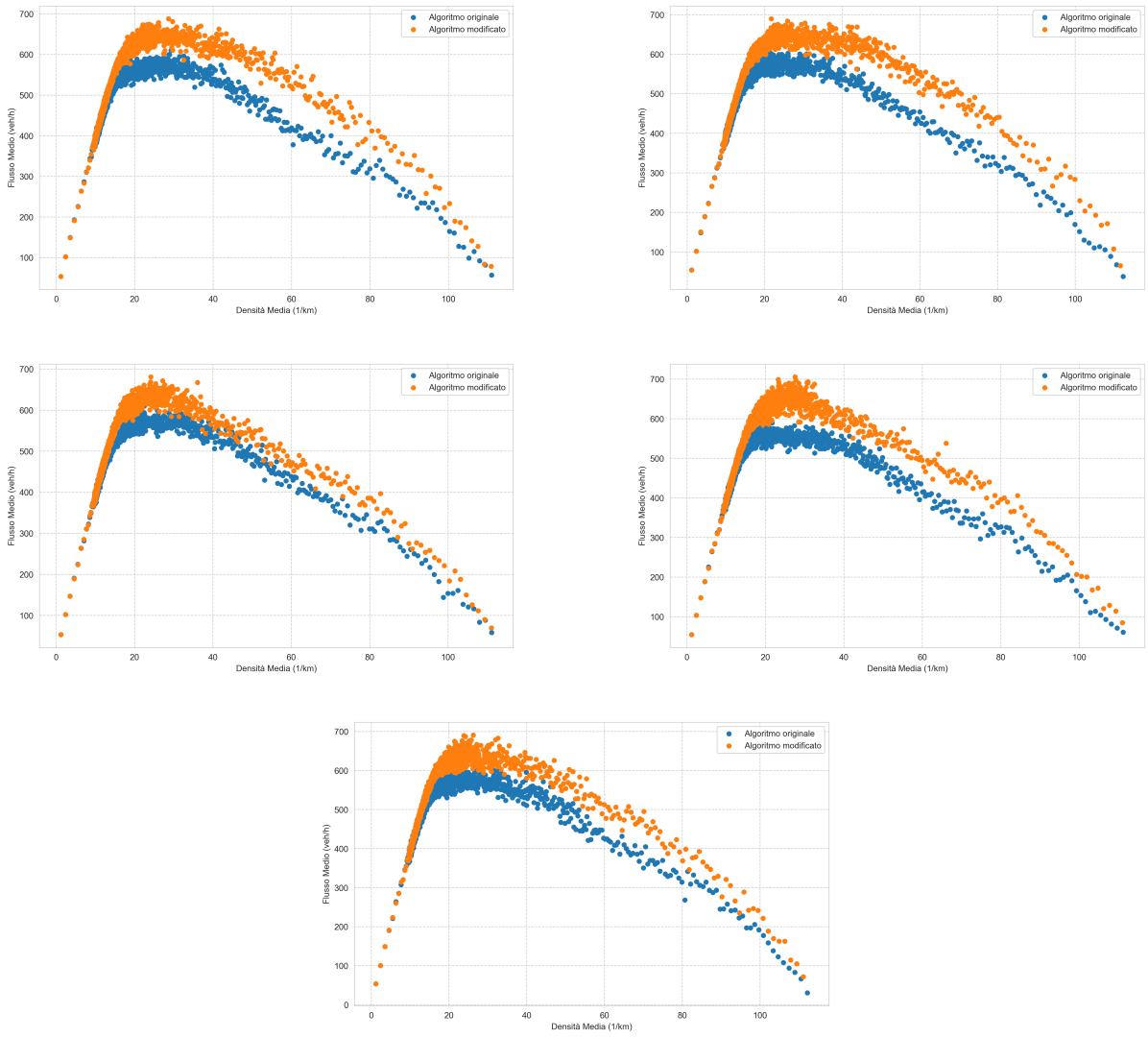

Figura C.3: MFD flusso-densità per le simulazioni in cui il valore *nAgents* è di 215 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 100 s e deviazione standard 20 s. Tutte le simulazioni possiedono gli stessi parametri, ad eccezione del seed.

Appendice D

Risultati della simulazione in assenza di ottimizzazione

Di seguito viene riportato il grafico (Fig.D.1) in cui si confrontano gli MFD delle simulazioni soggette a ottimizzazione (tramite algoritmo originale o modificato) e non soggette ad alcuna ottimizzazione. Sono stati inoltre realizzati dei fit polinomiali per dare un'idea qualitativa di quanto il flusso medio vari tra una situazione e l'altra ed in nero è segnalato il valore massimo del flusso raggiunto nelle tre situazioni.

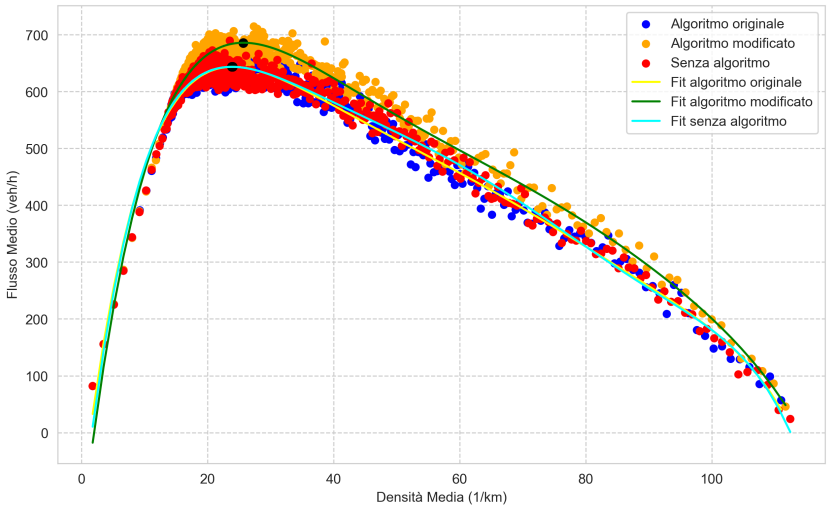

Figura D.1: MFD flusso-densità per le tre simulazioni in cui il valore *nAgents* è di 315 e i valori dei tempi di rosso e di verde iniziali dei semafori sono stati assegnati secondo una distribuzione gaussiana di media 60 s e deviazione standard 10 s.

Si può osservare, infatti, che, nei casi in cui non si ha ottimizzazione (curva ciano) o l'ottimizzazione è realizzata tramite l'algoritmo originale (curva gialla), il flusso medio

non varia molto, a differenza del caso in cui l'ottimizzazione è realizzata tramite il nuovo algoritmo (curva verde). Pertanto, per quanto riguarda l'aumento della scorrevolezza della rete, dal momento che non si riscontrano particolari differenze tra la situazione in assenza di ottimizzazione e quella in presenza di ottimizzazione tramite algoritmo originale, quest'ultima situazione viene utilizzata come punto di partenza e confronto in questa tesi.

La versione originale dell'algoritmo, pur non avendo effetti sulla scorrevolezza della rete, possiede altri effetti positivi sul network, ad esempio è in grado di ritardare la congestione nel tempo. Per un confronto più dettagliato tra l'algoritmo originale e una situazione in assenza di ottimizzazione si faccia riferimento a [19].

Bibliografia

[1] A. Schadschneider, "Statistical physics of traffic flow", *Physica A: Statistical Mechanics and its Applications*, vol. 285, 1-2, 101-120, 2000. Link: https://doi.org/10.1016/S0378-4371(00)00274-0.

[2] D. Chowdhury, L. Santen, A.s Schadschneider, "Statistical Physics of Vehicular Traffic and Some Related Systems", *Physics Reports*, vol. 329, 4-6, 199, 2000. Link: https://doi.org/10.1016/S0370-1573(99)00117-9.

[3] S. Lämmer e D. Helbing "Self-Control of Traffic Lights and vehicle Flows in Urban Road Netwoks" *Physics*, Feb. 2008. Link: https://arxiv.org/abs/0802.0403v2.

[4] N. Geroliminis, C. F. Daganzo, "Existence of urban-scale macroscopic fundamental diagrams: Some experimental findings", *Transportation Research* Part B 42 759–770, 2008. Link: https://doi.org/10.1016/j.trb.2008.02.002.

[5] B. S. Kerner, "The physics of traffic", *Physics World*, vol. 12, no. 8, p. 25, Ago. 1999. Link: https://dx.doi.org/10.1088/2058-7058/12/8/30.

[6] M. J. Lighthill e G. B. Whitham, "On kinematic waves II. A theory of traffic flow on long crowded roads", *Proceedings of the royal society of London. series a. mathematical and physical sciences*, vol. 229, no. 1178, pp. 317–345, 1955.

[7] M. Brackstone, M. McDonald, "Car-following: a historical review", *Transportation Research* Part F 2 181-196, 1999. Link: https://doi.org/10.1016/S1369-8478(00)00005-X.

[8] M. Bando, K. Hasebe, K. Nakanishi e A. Nakayama, "Analysis of Optimal Velocity Model with Explicit Delay", *Phys. Rev. E*, vol. 58, 5429–5435, 1998. Link: https://link.aps.org/doi/10.1103/PhysRevE.58.5429.

[9] E. B. Lieberman, "Brief history of traffic simulation", *Traffic and Transportation Simulation*, vol. 17, 2014. Link: https://onlinepubs.trb.org/onlinepubs/circulars/ec195.pdf.

[10] C. M. J. Tampère, B. van Arem, e S. P. Hoogendoorn, "Gas-Kinetic Traffic Flow Modeling Including Continuous Driver Behavior Models", *Transportation Research Record*, 1852(1), 231-238, 2003. Link: https://doi.org/10.3141/1852-28.

[11] T. Nagatani, "The physics of traffic jams", *Reports on Progress in Physics*, vol. 65, no. 9, p. 1331, Ago. 2002. Link: https://dx.doi.org/10.1088/0034-4885/65/9/203.

[12] J. W. Godfrey, "The mechanism of a road network", *Traffic Engineering and Control* 11 (7), 323–327, 1969.

[13] C. F. Daganzo, N. Geroliminis, "An analytical approximation for the macroscopic fundamental diagram of urban traffic", *Transportation Research* Part B, vol. 42, 771-781, 2008. link: https://doi.org/10.1016/j.trb.2008.06.008.

[14] T. Van Woensel e N. Vandaele, "Modeling traffic flows with queuing models: A review", *Asia-Pacific Journal of Operational Research*, vol. 24, no. 04, pp. 435–461, 2007. Link: https://doi.org/10.1142/S0217595907001383.

[15] B. S. Kerner, *The Physics of Traffic*, Springer, Berlin, 2004.

[16] E. Andreotti, A. Bazzani, S. Rambaldi, N. Guglielmi, e P. Freguglia, "Modeling traffic fluctuations and congestion on a road network" *Advances in Complex Systems*, vol. 18, no. 03n04, p. 1 550 009, 2015.

[17] D. Helbing, J. Siegmeier e S. Lämmer, "Self-Organized Network Flows", *AIMS Journals*, vol. 2, no. 2, Giu. 2007. Link: https://arxiv.org/pdf/physics/0702173.

[18] S. Lämmer e D. Helbing "Self-Stabilizing Decentralised Signal Control of Realistic, Saturated Network Traffic" *SFI WORKING PAPER* 2010-09-019. Link: https://www.santafe.edu/research/results/working-papers/self-stabilizing-decentralized-signal-control-of-r.

[19] G. Berselli, "Advanced queuing traffic model for accurate congestion forecasting and management" Tesi di Laurea Magistrale, 2024. Link: https://amslaurea.unibo.it/32191/

[20] G. Berselli e S. Balducci, *Framework for modelling dynamical complex systems*, 2024. Disponibile online: https://github.com/sbaldu/DynamicalSystemFramework

[21] *Induction Loop*. Link: https://en.wikipedia.org/wiki/Induction_loop.

[22] C. Antoniou, R. Balakrishna e H. N. Koutsopoulos, "A Synthesis of emerging data collection technologies and their impact on traffic management applications", *Eur. Transp. Res. Rev.*, 3:139–148, 2011. Link: `https://link.springer.com/content/pdf/10.1007/s12544-011-0058-1.pdf`.

Ringraziamenti

Vorrei ringraziare il mio relatore Armando Bazzani per avermi permesso di svolgere questo percorso di tesi con lui. Vorrei inoltre fare un ringraziamento speciale al mio correlatore Filippo Dalla per il supporto e la disponibilità costanti durante la realizzazione dell'algoritmo e della stesura della tesi stessa, il cui aiuto è stato per me indispensabile.

Ringrazio inoltre i miei compagni di corso e amici che, più che soli compagni di corso, sono diventati la mia famiglia per questi tre anni e sono persone preziose che faranno sempre parte della mia vita. Loro sono le persone che hanno questa esperienza universitaria la mia *core memory* più bella. In particolare ringrazio Simone Naglieri che è stato sempre al mio fianco pronto a incoraggiarmi ogni qual volta fosse necessario fornendomi nuove prospettive con cui guardare il mondo.

Ringrazio i miei genitori e in particolare mia sorella Vittoria che, seppur da lontano, è sempre stata fondamentale per me, sia come punto di riferimento negli studi che come punto di riferimento nella vita. Ringrazio inoltre Ursula perché mi ha permesso di essere la persona che sono oggi e mi ha permesso di arrivare fino a qui, nonostante tutte le premesse.

Ringrazio tutte quelle persone che, magari anche se da lontano o per poco tempo, hanno contribuito a rendermi la persona che sono oggi, come alcuni miei insegnanti e gli amici che sono rimasti fino ad oggi.

Infine vorrei ringraziare me stessa per essermi data la possibilità di mettermi in gioco in questa task non banale e per aver confidato fino alla fine nelle mie possibilità di riuscire a realizzare qualcosa di cui poter essere fiera.