# ALMA MATER STUDIORUM
# UNIVERSITÀ DI BOLOGNA

---

## DIPARTIMENTO DI INGEGNERIA CIVILE, CHIMICA, AMBIENTALE E DEI MATERIALI

*LAUREA MAGISTRALE IN INGEGNERIA CIVILE*


**TESI DI LAUREA**

in

LABORATORIO DI INFRASTRUTTURE VIARIE E TRASPORTI

# CREAZIONE DI UN MODELLO DI MICROSIMULAZIONE DEL TRAFFICO NELLA CITTÀ DI SAN FRANCISCO


CANDIDATO:

**Luca Vignoli**

RELATORE:

**Chiar.mo Prof. Joerg Schweizer**

CORRELATORE:

**Chiar.mo Prof. Federico Rupi**



Anno Accademico 2023/2024

Sessione I

# Sommario

# 1. INTRODUZIONE

## 1.1. INTRODUZIONE AL PROBLEMA

La domanda di trasporto di persone e di merci nelle principali città e metropoli è in continuo aumento e le reti di trasporto che le compongono risultano sempre più congestionate. La centralizzazione dei servizi e delle principali attività lavorative nei centri urbani sicuramente è la causa principale di questa tendenza. Inoltre, la non piena affidabilità e comodità del trasporto pubblico nelle ore di punta fa propendere i lavoratori o altre categorie di utenti all'utilizzo della propria automobile, il che aggrava ancora di più la situazione.

A causa di questi fattori, risulta necessaria una pianificazione urbana intelligente e sostenibile, la quale possa individuare quali siano i punti critici della rete di trasporto per poterli studiare e migliorarli. A tal proposito, è sempre più diffuso l'utilizzo di modelli basati sulla creazione di una popolazione virtuale e una rete che riproduca quella reale, detti *Digital Twin*.

Lo studio in oggetto si pone l'obiettivo di applicare il concetto di *Digital Twin* alla città di San Francisco e creare una simulazione microscopica del traffico urbano durante le ore di punta della mattina. Si tratta di un qualcosa di nuovo per la città in oggetto poiché dal punto di vista trasportistico esiste solamente una simulazione mesoscopica della San Francisco Bay Area. Quest'area metropolitana nella California settentrionale ospita otto milioni di persone e comprende nove contee, ed è perciò impensabile condurre una simulazione che non sia macroscopica o mesoscopica come fatto dal Berkeley Lab nell'Università di Berkeley.

L'analisi microscopica permette di considerare l'interazione tra i veicoli presenti all'interno della rete e avere riscontri abbastanza precisi in termini di tempi di percorrenza, velocità medie, emissioni e incidentalità, a differenza degli altri tipi di simulazione che restituiscono invece solo stime a grandi linee. Questi dati si possono poi analizzare per calibrare i servizi offerti, prevedere le conseguenze causate da possibili

scenari di traffico futuri e valutare eventualmente l'efficienza di interventi alternativi sulla rete stradale reale.

## 1.2. OBIETTIVI

Il presente studio si pone quindi l'obiettivo di creare una rappresentazione virtuale della rete di San Francisco con il software hybridPY. Trattandosi di una rete molto vasta e non comune per una microsimulazione, il lavoro di elaborazione dell'offerta di trasporto è sicuramente più lungo e dispendioso rispetto a quello fatto sulle città normalmente oggetto di esse.

Il seguente elaborato, dopo un'introduzione relativa ai *Digital Twin* e ai vari tipi di simulazione in ambito di pianificazione urbana, analizza il software utilizzato, illustrando anche i principali elementi che compongono una rete di trasporto urbana all'interno di esso.

La trattazione prosegue con le attività oggetto del tirocinio svolto, vale a dire la modifica della rete di San Francisco per renderla il più fedele possibile alla realtà, l'importazione della domanda di trasporto fornita dal Berkeley Lab, e la simulazione del traffico urbano dopo aver assegnato la domanda alla rete di trasporto. Tale processo non è unidirezionale poiché in base alle simulazioni possono risultare sulla rete dei punti da migliorare, rendendolo più propriamente un processo circolare.

Successivamente, dopo aver raggiunto un grado di definizione soddisfacente della rete e della simulazione di traffico, si presentano i parametri utilizzati e i risultati ottenuti per ogni tipo di simulazione svolta all'interno del software.

Infine, dopo aver concluso tutti i passaggi relativi alla microsimulazione del traffico, al fine di dare un'impronta grafica al progetto, si illustrano brevemente i passaggi per creare una visualizzazione tridimensionale della città di San Francisco con il software Unreal Engine, il quale riproduce parzialmente ciò che è contenuto all'interno della simulazione del traffico urbano.

## 1.3. *DIGITAL TWIN*

Un *Digital Twin* (tradotto Gemello Digitale) è una rappresentazione digitale di un oggetto, di un processo o di un sistema del mondo reale. Questa rappresentazione digitale viene creata attraverso la raccolta e l'analisi di dati provenienti da sensori, dispositivi IoT (*Internet of things*), software di monitoraggio e altre fonti e riflette in modo accurato lo stato e il comportamento dell'oggetto o del sistema reale in tempo reale o in modalità simulata.

L'idea dei Gemelli Digitali è stata espressa per la prima volta nel 1991, con la pubblicazione di *Mirror Worlds* di David Gelernter. Tuttavia, la prima applicazione nel settore manifatturiero nel 2002 e l'annuncio formale di un software incentrato sui gemelli digitali sono attribuiti al Dr. Michael Grieves (all'epoca docente presso la University of Michigan).

L'idea di base di utilizzare un *Digital Twin* come mezzo per studiare un oggetto fisico è però riconducibile a molto tempo prima, quando la NASA durante le sue missioni esplorative nello spazio degli anni '60 realizzò una replica fedele destinata a rimanere a terra, utilizzata per analisi e simulazioni dal personale NASA impegnato come equipaggio.

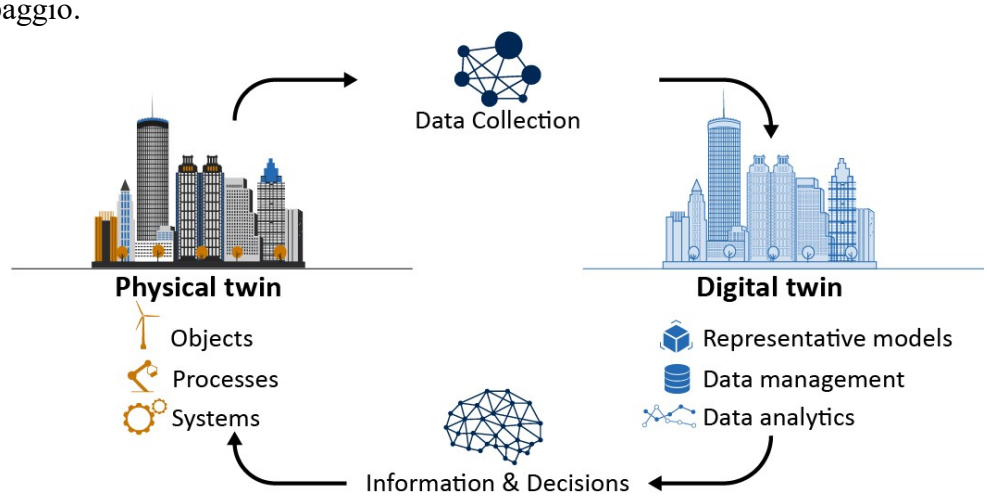


*Figura 1.1: Schema esplicativo Gemello Digitale*

Come si può notare dalla figura, per la realizzazione di un Gemello Digitale sono necessarie tre dimensioni: prodotti fisici nello spazio reale, prodotti virtuali nello spazio

virtuale e sistemi di collegamento dei flussi di dati e informazioni che uniscono lo spazio fisico a quello virtuale e ai suoi sottospazi.

### 1.3.1. Impieghi

Un *Digital Twin* mira a essere una replica accurata dell'oggetto o del sistema reale che rappresenta, il che significa che dovrebbe riflettere non solo l'aspetto fisico dell'oggetto, ma anche il suo comportamento, le sue prestazioni e le sue interazioni con l'ambiente circostante.

I dati raccolti dai sensori e da altre fonti vengono continuamente utilizzati per aggiornarlo in tempo reale in modo da monitorare costantemente lo stato e il comportamento dell'oggetto e di prevedere e rispondere prontamente a eventuali cambiamenti o anomalie, ovviamente prima di implementarle nel mondo reale.

I Gemelli Digitali hanno applicazioni in una vasta gamma di settori, tra cui manifatturiero, energia, trasporti, edilizia, salute e molto altro ancora. A titolo di esempio, nel settore manifatturiero, possono essere utilizzati per monitorare e ottimizzare le linee di produzione, mentre nel settore della salute per simulare il comportamento di organi o tessuti nel corpo umano.

### 1.3.2. Mobilità

Tra gli ambiti citati, quello più di interesse per questo studio è sicuramente quello legato ai trasporti e alla mobilità urbana. In questo senso, sono nati gli *Urban Digital Twin*, i quali rappresentano uno degli strumenti più avanzati per la comprensione e l'analisi dell'ecosistema cittadino. Essi integrano varie fonti di informazioni, sia statiche che dinamiche, provenienti da contesti complementari e diversificati, per esempio utilizzano dati storici e aggiornati per fornire una visione dettagliata di infrastrutture, edifici, sistemi di trasporto, spazi verdi cittadini e altre risorse urbane, includendo anche informazioni da sensori in tempo reale per aggiungere una dimensione dinamica alle proprie rappresentazioni.

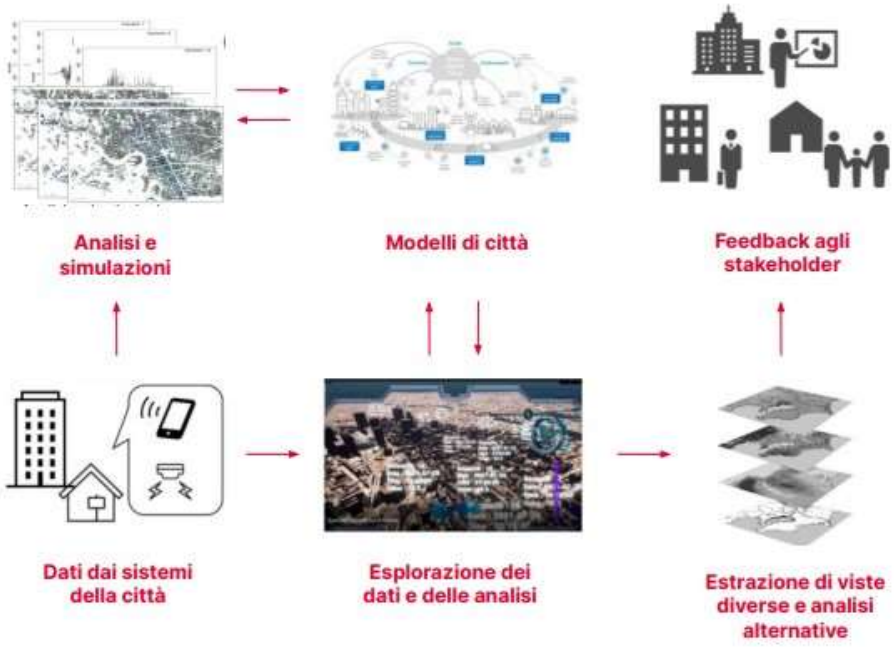


*Figura 1.2: Possibili sviluppi dell'Urban Digital Twin*

Tra gli esempi di utilizzo dell'*Urban Digital Twin*, i più importanti legati alla mobilità urbana sono:

- Simulazioni del flusso di traffico → le autorità municipali li utilizzano per simulare il flusso del traffico in varie condizioni, inclusi orari di punta, eventi speciali e lavori stradali. Queste simulazioni consentono di valutare l'impatto di nuove infrastrutture stradali, modifiche al layout stradale o cambiamenti nei modelli di trasporto pubblico sulla circolazione stradale e sulle congestioni.

- Monitoraggio del traffico in tempo reale → integrano dati provenienti da sensori di traffico, telecamere di sorveglianza e altri sistemi di monitoraggio per creare una rappresentazione in tempo reale del traffico urbano e individuare congestioni e incidenti in modo tempestivo per coordinare le risposte agli eventi stradali. Questo può includere la regolazione dei semafori, la modifica delle rotte dei mezzi pubblici o la segnalazione di percorsi alternativi ai conducenti attraverso segnaletica dinamica o app di navigazione.

- Previsione del traffico e analisi dei dati storici → utilizzando dati storici e modelli di previsione, possono essere utilizzati per prevedere il flusso del traffico futuro e

identificare tendenze e nel comportamento del traffico urbano. Tali informazioni vengono utilizzate per pianificare interventi di infrastruttura, ottimizzare la distribuzione dei servizi di trasporto pubblico e migliorare la gestione del traffico nel lungo termine.

### 1.3.3. Applicazioni

I Gemelli Digitali del traffico urbano sono stati utilizzati già in diverse città in tutto il mondo, come per esempio Sidney, Dubai, Londra, Barcellona, Shangai e Singapore. Grazie a questa tecnologia, esse sono state in grado e sono in grado di migliorare la sicurezza stradale, ottimizzare e migliorare il trasporto pubblico, sviluppare soluzioni di mobilità intelligente e gestire il traffico in caso di congestione o lavori stradali.

Per quanto riguarda il panorama italiano, nel settembre 2023 è stato annunciato che anche Bologna avrà un *Digital Twin*, finanziato da un investimento iniziale di 7 milioni di euro provenienti da fondi europei PON Metro. Gli ambiti in cui il Comune si propone di impegnarsi maggiormente sono i seguenti:

- Mobilità → supporto alla città nelle sfide che trasformeranno la mobilità urbana (città a 30 km/h, Tram, Passante, Bicipolitana) e valorizzazione del patrimonio di dati, tra cui integrazione e accesso ai dati, analisi avanzata, nuove sorgenti dati da aziende partner.
- Energia → analisi della risposta energetica del patrimonio edilizio cittadino e supporto alle valutazioni di sostenibilità; simulazione dell'impatto dell'inserimento di nuovi progetti nei piani urbanistici, di alternative progettuali, di politiche e incentivi.
- Cambiamenti climatici → alla luce dell'emergenza che ha colpito l'Emilia-Romagna, sarà sviluppato un caso d'uso legato ai temi del cambiamento climatico e dissesto idrogeologico.

All'interno di questo progetto, l'Alma Mater Studiorum ha il ruolo di leader scientifico e responsabile del comitato scientifico e guida la parte di ricerca.

## 1.4. SIMULAZIONE DI UNA RETE DI TRASPORTO

Una simulazione del traffico costituisce una replica virtuale di situazioni effettive di traffico, offrendo agli esperti un ambiente digitale in cui possono creare modelli dettagliati di incroci, strade e altre infrastrutture viarie. Nelle città convergono pedoni, ciclisti, mezzi pubblici e veicoli motorizzati, gli spazi sono limitati e i sistemi di trasporto multimodali coinvolgono una vasta gamma di utenti, modalità e servizi, per cui le simulazioni del traffico consentono ai pianificatori di comprendere questa intricata rete dinamica, sviluppando strategie efficaci per risolvere le sfide attuali e future legate al traffico. Attraverso la modellazione e la simulazione del flusso di traffico, è possibile testare diverse ipotesi e valutare gli effetti delle misure pianificate sul traffico complessivo, fornendo preziose informazioni per la pianificazione urbana. Questo approccio virtuale è particolarmente vantaggioso poiché progettare e ottimizzare infrastrutture attraverso simulazioni prima dell'implementazione fisica aiuta a ridurre i rischi e i costi associati ai potenziali problemi.

Inoltre, la chiara visualizzazione delle simulazioni migliora la comunicazione, consentendo ai pianificatori di presentare in modo convincente le proprie idee e misure ai decisori e al pubblico interessato. In questo modo, le simulazioni del traffico forniscono una solida base per le decisioni di pianificazione e contribuiscono a promuovere soluzioni efficaci per le sfide urbane legate alla mobilità.

Un passaggio cruciale per garantire la bontà di un modello è la calibrazione: non si può semplicemente trasferire un modello da una città all'altra ma deve essere adattato alle specifiche condizioni locali. Per verificare che il modello rifletta fedelmente i dati reali, vengono impiegati metodi statistici standard come le statistiche GEH o la bontà dell'adattamento.

Le simulazioni del traffico, specialmente quando si considerano scenari futuri, operano su diverse ipotesi. Poiché il futuro è intrinsecamente incerto, vengono generati vari scenari per contemplare differenti evoluzioni delle infrastrutture, dell'uso del suolo e dello sviluppo socioeconomico. Pertanto, i risultati del modello devono essere interpretati e valutati tenendo conto di tali premesse.

Tra i tipi di simulazione, se ne distinguono principalmente tre: a carattere macroscopico, mesoscopico, e infine microscopico. Quest'ultimo è il tipo di simulazione utilizzato nel presente studio.

### 1.4.1. Macroscopica

I modelli di macrosimulazione trattano il traffico nella sua totalità, descrivendolo come un flusso guidato da regole comportamentali che coinvolgono principalmente l'interazione dei veicoli tra di loro e con l'infrastruttura stradale. Tali modelli (di tipo aggregato) si concentrano soprattutto sull'evoluzione complessiva del traffico, fornendo previsioni sulla velocità media, il flusso e la densità veicolare.

Possono essere classificati in base alla loro rappresentazione dello spazio e del tempo, assumendo che entrambi siano continui: i modelli a spazio continuo si basano su variabili definite in ogni punto dello spazio e richiedono soluzioni numeriche complesse per situazioni più intricate; d'altra parte, i modelli a spazio discreto considerano le variabili di base a livello dell'intera area, come fatto dai modelli statici, e richiedono una discretizzazione temporale per la loro soluzione.

La teoria matematica alla base dei modelli di macrosimulazione rappresenta il traffico come un fluido continuo e si avvale di equazioni differenziali che esprimono le leggi di conservazione del flusso. Sebbene questi modelli offrano una rappresentazione compatta e generale del problema, possono incontrare difficoltà nel cogliere fenomeni come la formazione di code o la sovrasaturazione del sistema, i quali vengono analizzati molto bene invece dai modelli microscopici e abbastanza bene da quelli mesoscopici.

### 1.4.2. Mesoscopica

I modelli mesoscopici si posizionano tra l'approccio aggregato dei modelli macroscopici e quello più dettagliato dei modelli microscopici. Questi modelli offrono una rappresentazione più dettagliata rispetto ai modelli macroscopici, anche se le prestazioni della rete vengono simulate a livello aggregato utilizzando variabili come la capacità, il flusso e la densità. L'attenzione dei modelli mesoscopici è concentrata sul comportamento

di gruppi di utenti, e gli output ottenuti si riferiscono proprio a tali gruppi anziché ai singoli veicoli, come avviene invece nella microsimulazione.

Ciascun gruppo, noto come *pacchetto*, è composto da veicoli con la stessa origine, destinazione e strategia di scelta durante il loro percorso, e per ognuno di essi viene simulato uno spostamento rigido seguendo la traiettoria nello spazio. La dimensione del gruppo influisce sulla somiglianza della soluzione del modello con l'approccio microscopico o macroscopico: gruppi più piccoli formati da pochi utenti si avvicinano maggiormente ad un'analisi dettagliata perché vengono quasi scelti singolarmente, mentre gruppi più numerosi si avvicinano di più ad un'analisi macroscopica e d'insieme.

I modelli mesoscopici sono in grado di rappresentare una vasta gamma di fenomeni dinamici, inclusi la sovrasaturazione, la formazione e la propagazione di code, tuttavia non possono simulare fenomeni come sorpassi, cambi di corsia e altri comportamenti di deflusso, più adatti allo studio di tipo microscopico.

Tra i pacchetti di simulazione mesoscopici, si possono menzionare MATSim e BEAM CORE, i quali collaborano con l'approccio microscopico e hanno un ruolo importante per lo studio in esame.

#### 1.4.2.1. MATSim

MATSim è un framework open-source progettato per simulazioni di traffico su larga scala basate su agenti. Si basa su un algoritmo co-evolutivo in cui gli agenti ottimizzano egoisticamente i loro programmi di attività quotidiane regolando, ad esempio, l'orario di partenza, il mezzo di trasporto utilizzato o il percorso e l'ottimizzazione viene eseguita in un ciclo fino al raggiungimento di uno stato di equilibrio. Il software consente di modellare diverse modalità di trasporto come auto, camminata, bicicletta e trasporto pubblico, singolarmente ma senza interazione. Nonostante le sue limitazioni dovute alle dinamiche semplificate, è in grado di simulare in modo efficiente scenari di grandi dimensioni, privilegiando la velocità rispetto alla precisione delle dinamiche del traffico e della guida.

#### 1.4.2.2. BEAM CORE

BEAM CORE è un software di modellazione completo che simula le dinamiche di passeggeri e merci nelle aree urbane utilizzando la *pipeline* di modelli PILATES. È composto da vari modelli interconnessi che creano una popolazione sintetica, proprietà dei veicoli, attività quotidiane, scelte di modalità e percorsi. All'interno di esso sono presenti vari strumenti che svolgono compiti diversi al fine di creare la domanda di trasporto:

- SynthPop, DEMOS e Urbansim generano una popolazione sintetica con la sua evoluzione nel corso degli anni, tenendo conto della demografia, dell'uso del suolo e delle caratteristiche delle famiglie.
- ADOPT, FASTSim e ATLAS prevedono la proprietà e le caratteristiche dei veicoli nel corso degli anni.
- ActivitySim modella le attività quotidiane, le posizioni e le modalità di trasporto, fornendo informazioni sui piani.
- BEAM CORE poi simula i viaggi utilizzando un approccio mesoscopico basato su eventi per tutte le modalità di trasporto, adattabile a studi su larga scala. Il servizio di trasporto pubblico su BEAM CORE è riprodotto attraverso file GTFS, mentre i servizi on-demand sono modellati da un gestore di richieste di veicoli interno.
- RouteE calcola il consumo energetico dei veicoli.

Ogni sottomodello di BEAM CORE può essere calibrato indipendentemente, consentendo regolazioni specifiche per scenario senza la necessità di ricostruire l'intero modello. Inoltre, itera la simulazione dei piani di viaggio con BEAM CORE e la regolazione delle scelte di modalità su ActivitySim in base alle stime di trasporto, con l'obiettivo di raggiungere un equilibrio nei modelli di viaggio.

Dopo una prima calibrazione del caso di studio, le simulazioni successive possono iniziare da stime già calibrate, riducendo drasticamente le iterazioni necessarie per convergere nuovamente sui risultati. Tuttavia, BEAM CORE è intensivo dal punto di vista computazionale e tipicamente richiede più potenza di calcolo rispetto a un laptop standard, soprattutto per simulazioni su larga scala. In questo senso, può eseguire studi su scala ridotta concentrandosi su un sottoinsieme della popolazione e scalando i risultati nel

*post-processing*. Questa scalabilità si applica anche alle reti stradali, alle flotte di *ride-hail* e alle capacità di trasporto pubblico per simulare accuratamente gli effetti della congestione.

Il modello BEAM CORE ha già alcuni processi di post-elaborazione integrati, e uno in particolare aggrega i risultati basati su eventi come viaggi porta a porta per valutare una misura di accessibilità chiamata INEXUS. Questo database è particolarmente importante per questo studio poiché è compatibile con altre piattaforme di simulazione come SUMO.

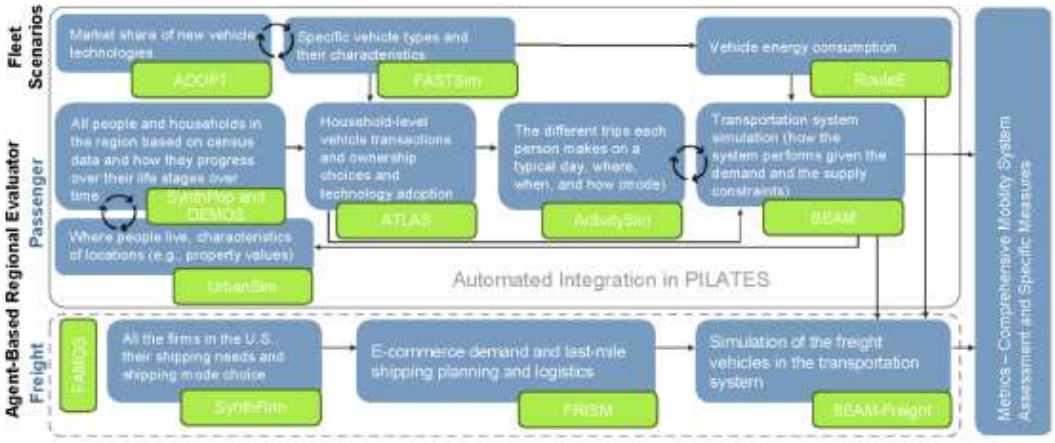


*Figura 1.3: Struttura BEAM CORE*

### 1.4.3. Microscopica

Il concetto di microsimulazione si applica quando si desidera studiare l'evoluzione del traffico concentrandosi sull'analisi delle singole unità; infatti, questi modelli (noti anche come modelli disaggregati) offrono una rappresentazione puntuale, precisa e specifica del traffico e della sua dinamica istante per istante, consentendo di simulare il movimento di ciascun veicolo della rete e analizzare aspetti come velocità, accelerazione e interazioni con gli altri veicoli (come sorpassi e cambi di corsia). Per fare ciò, considerano in dettaglio gli aspetti geometrici dell'infrastruttura stradale, oltre al comportamento reale del conducente, il quale è influenzato dalle caratteristiche del veicolo e dalle abitudini di guida, raggiungendo in questo modo un livello di dettaglio molto elevato, nettamente superiore a quello presente negli altri due tipi di simulazione del traffico.

Nello specifico, la microsimulazione prende in considerazione i seguenti elementi del sistema di trasporto:

- Strade → composte da corsie con caratteristiche specifiche come larghezza, velocità massima e diritti di accesso.
- Incroci → con informazioni sulle svolte consentite e la segnaletica (ad esempio, segnali di stop o precedenza). Gli incroci semaforizzati includono i programmi delle fasi semaforiche.
- Movimento di ogni singolo veicolo lungo un itinerario → posizione, velocità, accelerazione.
- Comportamento dei veicoli di diverse tipologie → mezzi pesanti, autobus, motocicli, biciclette, pedoni, ecc.
- Interazione tra i veicoli → distanziamento, sorpasso, incroci, pedoni, ecc. La dinamica del veicolo dipende non solo dalle sue caratteristiche tecniche e dalla posizione degli altri veicoli, ma anche dal comportamento del conducente.
- Interazione tra veicoli e infrastruttura → semafori, parcheggi, caselli, ecc.
- Analisi delle prestazioni e dell'impatto ambientale → di ogni singolo veicolo, arco o dell'intera rete.
- Analisi di utilità del percorso → multimodale per ogni singolo utente.

Allo stesso tempo, però, tali modelli presentano una complessità notevole: solo di recente, grazie all'aumento delle capacità di calcolo, sono diventati strumenti di analisi molto interessanti, in grado di operare con una risoluzione temporale e spaziale elevata. A causa della grande quantità di dati necessari e prodotti da tali modelli (per esempio si pensi a simulazioni che coinvolgono le attività giornaliere di più di un milione di persone), generalmente sono adatti a reti di dimensioni ridotte, dove non sono richieste risorse computazionali e di memoria eccessive.

Il movimento dei veicoli è determinato dalle scelte individuali e dalle interazioni con gli altri mezzi presenti sulla rete stradale. Le decisioni riguardanti il percorso da seguire, l'accelerazione, i cambi di corsia, distanza di sicurezza e il comportamento alle

intersezioni di ciascun veicolo sono modellate in modo esplicito e dettagliato sulla base di algoritmi comportamentali e non sono separate dal controllo del veicolo poiché vengono visti come un'unica entità. Inoltre, ogni veicolo possiede caratteristiche specifiche relative alle prestazioni del veicolo stesso, come l'accelerazione massima o la velocità massima, e alle caratteristiche del conducente, come il tempo di reazione o la velocità desiderata.

Nonostante i modelli microscopici riproducano il movimento dei singoli veicoli in modo dettagliato, vengono spesso impiegati anche per analisi aggregate, come la valutazione dei flussi di traffico, la lunghezza delle code e i tempi di percorrenza complessivi. Questo approccio consente di ottenere una visione completa e dettagliata delle dinamiche del traffico, fornendo informazioni utili per la progettazione e l'ottimizzazione delle infrastrutture stradali. Per queste ragioni, i modelli microscopici rappresentano lo strumento più avanzato per valutare gli effetti di scelte progettuali e regolamentari sulla rete stradale, sia a livello locale che su scala più ampia, rendendo possibile la progettazione di nuove infrastrutture stradali, sistemi di controllo semaforico, corsie riservate, zone a traffico limitato, fino a simulare eventi eccezionali come incidenti e cantieri stradali, che possono temporaneamente limitare la capacità delle sezioni stradali e influenzare significativamente le condizioni del traffico veicolare.

Graficamente, i software di microsimulazione consentono di visualizzare animazioni tridimensionali e bidimensionali delle condizioni istantanee della rete stradale, comprese la velocità di percorrenza e la presenza di code. Dal punto di vista statistico, invece, è possibile ottenere informazioni su flussi di traffico, velocità medie, tempi di viaggio, frequenza di fermata dei veicoli, tempo trascorso in coda, lunghezza delle code, consumo di carburante ed emissioni inquinanti.

La principale differenza tra i modelli macroscopici e gli attuali modelli microscopici in via di sviluppo risiede nel risultato finale e nella difficoltà nel reperire i dati necessari allo studio. In particolare, l'uso di un approccio macroscopico richiede una conoscenza approssimativa della domanda di trasporto, descritta da una matrice OD (origine-destinazione) tra le diverse zone dell'area di studio. Al contrario, per studiare un sistema di trasporto con un approccio microscopico, è necessario conoscere dettagliatamente la

domanda di trasporto, analizzando i viaggi di ogni singola persona piuttosto che considerare solo il numero di persone che si muovono da una zona all'altra.

Anche la rete di trasporto gioca un ruolo cruciale nella differenziazione tra i due approcci: uno studio microscopico richiede una rete di trasporto con un elevato livello di dettaglio (che aumenta esponenzialmente all'aumentare della dimensione della rete), mentre uno studio macroscopico si basa su un grafo semplificato delle arterie principali. Inoltre, anche i modelli di assegnazione utilizzati differiscono: gli approcci macroscopici possono utilizzare modelli statici per reti congestionate e non, mentre gli approcci microscopici includono la variabile temporale, valutando il livello di traffico nella rete in ogni istante.

Un altro progresso riguarda la disponibilità di dati per caratterizzare più dettagliatamente la domanda e l'offerta di trasporto: sempre più città rendono disponibili online notevoli quantità di dati (come conteggi di traffico, reti di trasporto pubblico e tabelle degli orari aggiornate), database geografici (come OpenStreetMap) e applicazioni sui cellulari che permettono di tracciare i viaggi con la tecnologia GPS. Per queste ragioni, la calibrazione di modelli di traffico microscopici utilizzando i grandi volumi di dati attualmente disponibili, chiamati *big data*, è di notevole interesse. Nel campo della pianificazione dei trasporti, i modelli che permettono la microsimulazione sono l'evoluzione naturale dei modelli convenzionali, e consentono di modellare meglio i processi trasportistici e di predire con maggiore precisione i flussi di traffico e le relative emissioni/inquinanti nell'atmosfera. La necessità di utilizzare modelli più dettagliati è nata non solo dalla volontà di ottenere risultati migliori, ma anche dal fatto che il comportamento delle persone e i servizi di trasporto sono sempre più diversificati e complessi da modellare a causa dei seguenti motivi:

- Lavori part-time sempre più diffusi rispetto ai contratti regolari, che comportano una variazione più rapida della domanda di trasporto.
- Sviluppo di servizi di *bike/car-sharing* e percorsi multimodali.
- Cicli semaforici variabili nel tempo e spesso correlati a intersezioni vicine (per esempio l'onda verde).
- Applicazioni che permettono di identificare in tempo reale il percorso da seguire.

### 1.4.4. Domanda nella simulazione microscopica

Una microsimulazione richiede una domanda di trasporto molto dettagliata, in grado di descrivere lo spostamento del singolo utente in un preciso istante, anziché il flusso medio di persone stimato in un certo intervallo temporale. Generare la domanda di trasporto significa specificare i viaggi (*trips*), i percorsi (*routes*) per i veicoli oppure i piani (*plans*) per le persone.

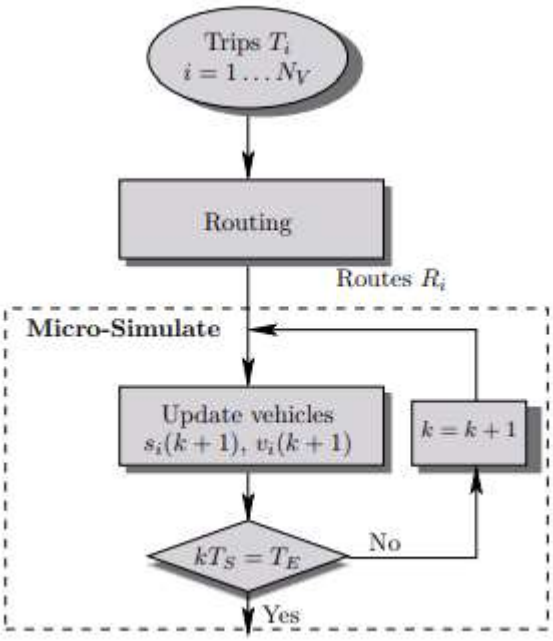


*Figura 1.4: Modellazione della domanda con microsimulazione*

Il viaggio, nell'ambiente della microsimulazione, è definito come lo spostamento da una posizione ad un'altra della rete, ed inizia in un determinato tempo utilizzando un certo modo di trasporto. La posizione, invece, è definita come la posizione x sull'arco a, dove x è la distanza dal nodo di ingresso dell'arco a. Quindi, si definisce il viaggio $T_i$ del veicolo i del tipo $m_i$ dalla posizione di origine $x_{oi}$ sull'arco di origine $a_{oi}$ all'arco di destinazione $a_{di}$ e posizione $x_{di}$ che prende inizio nell'istante $t_{oi}$ come:

$T_i = (a_{oi}, x_{oi}, a_{di}, x_{di}, m_i, t_{oi})$.

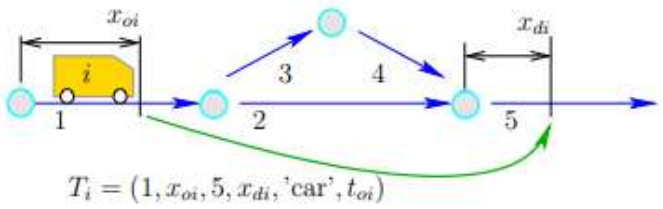


*Figura 1.5: Esempio di un viaggio con relativi attributi*

Il viaggio, però, non descrive il percorso preciso effettuato dal veicolo, infatti per quello c'è l'itinerario, il quale contiene la sequenza degli archi che porta il veicolo dall'arco di origine $a_o$ all'arco di destinazione $a_d$, oltre agli attributi del viaggio. Si definisce perciò l'itinerario $R_i$ del veicolo i del tipo $m_i$ dalla posizione di origine $x_{oi}$ sull'arco di origine $a_{oi}$ all'arco di destinazione $a_{di}$ e posizione $x_{di}$ che prende inizio nell'istante $t_{oi}$ come:

$R_i = (x_{oi}, x_{di}, [a_{1i}, a_{2i}, \dots, a_{ni,i}], m_i, t_{oi})$.

dove $[a_{0i}, a_{1i}, \dots, a_{ni,i}]$ è la sequenza di $n_i$ archi attraversati per andare dall'arco di origine $a_{1i} = a_{oi}$ all'arco di destinazione $a_{ni,i} = a_{di}$. Al contrario della definizione del viaggio, l'itinerario non lascia ambiguità sul percorso tra origine e destinazione.

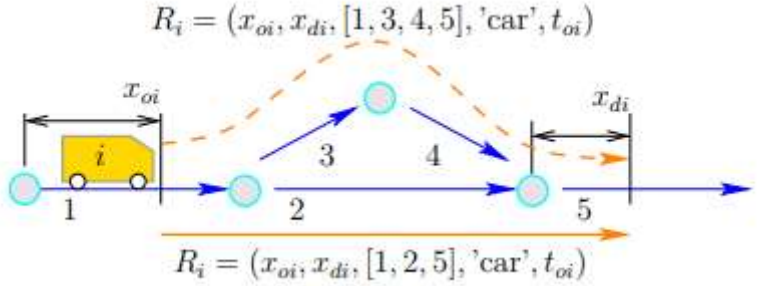


*Figura 1.6: Esempio di un itinerario con relativi attributi*

La microsimulazione, a differenza della teoria classica dei trasporti che richiede una descrizione aggregata dei dati, necessita di una domanda di trasporto in forma disaggregata, la quale descrive il viaggio di ogni singola persona. Chiaramente, è molto più impegnativo descrivere la domanda di trasporto in questa modalità, anche se con una buona stima di essa si arriva sicuramente a risultati più precisi. Le modalità e le

approssimazioni per la generazione di tale domanda dipendono fortemente dalla possibilità di ottenere determinate informazioni sul traffico, dallo scopo dello studio, dalla possibilità di utilizzare programmi che trattano enormi quantità di dati e dalla dimensione dell'area di studio. I metodi per la generazione della domanda di trasporto in forma disaggregata, supportati dall'ambiente di simulazione SUMO, sono tre:

- Metodo del conteggio dei flussi.
- Metodo dei flussi OD.
- Metodo della domanda basata sulle attività.

#### 1.4.4.1 Metodo del conteggio dei flussi

La generazione dei viaggi con il metodo del conteggio dei flussi consente di modellare molto precisamente i flussi di traffico in un determinato intervallo temporale su piccole reti di trasporto senza loop (itinerari circolari), e con poca generazione di traffico al suo interno.

L'idea alla base del metodo è quella di poter generare viaggi ed itinerari per i singoli veicoli a partire da alcuni conteggi del traffico, i quali devono essere effettuati in ogni arco dell'area di studio e, se possibile, simultaneamente. Proprio per questo motivo, non è l'approccio ideale per lo studio di grandi reti di trasporto, ma alcuni flussi possono essere ottenuti indirettamente facendo l'ipotesi di conservazione di essi nei nodi.

In particolare, servono due differenti tipologie di flussi per ricostruire gli itinerari per ogni modo di trasporto m e per ogni intervallo temporale h:

- I flussi generatori in entrata $FG_{a,m,h}$ per tutti gli archi a che entrano nell'area di studio.
- I flussi di svolta $FT_{a1,a2,m,h}$ per tutti i connettori che collegano l'arco $a_1$ con l'arco $a_2$, tranne che per i nodi che hanno un arco di uscita.

Nell'ambiente di simulazione SUMO esiste un apposito strumento chiamato JTROUTER che crea gli itinerari $R_i$ a partire dai dati forniti, i quali iniziano tutti dagli archi che entrano nell'area di studio analizzata e la attraversano. Successivamente, è possibile riprodurre lo

scenario in termini di microsimulazione fornendo al microsimulatore direttamente gli itinerari generati $R_i$.

| Mode | 7:00-8:00<br>Flussi generatori | Flussi di svolta | 8:00-9:00<br>Flussi gener. | Flussi di svolta | ... |
|---|---|---|---|---|---|
| Car | $FG_{3,\text{car},7} = 30$<br>$FG_{12,\text{car},7} = 20$ | $FT_{8,9,\text{car},7} = 10$<br>$FT_{8,10,\text{car},7} = 40$<br>$FT_{1,2,\text{car},7} = 0$<br>$FT_{1,7,\text{car},7} = 0$<br>$FT_{1,11,\text{car},7} = 10$<br>$FT_{10,7,\text{car},7} = 0$<br>$FT_{10,11,\text{car},7} = 40$ | $FG_{3,\text{car},8} = 15$<br>$FG_{12,\text{car},8} = 5$ | $FT_{8,9,\text{car},8} = 4$<br>$FT_{8,10,\text{car},8} = 16$<br>$FT_{1,2,\text{car},8} = 0$<br>$FT_{1,7,\text{car},8} = 0$<br>$FT_{1,11,\text{car},8} = 4$<br>$FT_{10,7,\text{car},8} = 0$<br>$FT_{10,11,\text{car},8} = 16$ | |
| Bus | $FG_{3,\text{bus},7} = 5$<br>$FG_{12,\text{bus},7} = 2$ | $FT_{8,9,\text{bus},7} = 2$<br>$FT_{8,10,\text{bus},7} = 5$<br>$FT_{1,2,\text{bus},7} = 0$<br>$FT_{1,7,\text{bus},7} = 0$<br>$FT_{1,11,\text{bus},7} = 2$<br>$FT_{10,7,\text{car},7} = 0$<br>$FT_{10,11,\text{car},7} = 5$ | $FG_{3,\text{bus},8} = 1$<br>$FG_{12,\text{bus},8} = 5$ | $FT_{8,9,\text{bus},8} = 4$<br>$FT_{8,10,\text{bus},8} = 2$<br>$FT_{1,2,\text{bus},8} = 0$<br>$FT_{1,7,\text{bus},8} = 0$<br>$FT_{1,11,\text{bus},8} = 4$<br>$FT_{10,7,\text{car},8} = 0$<br>$FT_{10,11,\text{car},8} = 2$ | |
| ⋮ | | | | | |

*Figura 1.7: Esempio di domanda descritta da flussi generatori e di svolta*

### 1.4.4.2 Metodo dei flussi OD

L'idea di questo approccio è quella di generare viaggi e successivamente itinerari a partire da dati riferiti a spostamenti fra diverse zone dell'area di studio. Tali spostamenti sono rappresentati da una matrice origine-destinazione ODM (*origin-to-destination matrix*), contenente il numero di spostamenti da una zona all'altra in un determinato arco temporale. La ODM è una rappresentazione aggregata della domanda di trasporto e descrive i flussi medi della domanda tra le varie zone (numero di spostamenti in un arco temporale); al fine di effettuare una microsimulazione è quindi necessaria la sua disaggregazione. Nell'ambito della microsimulazione, la domanda ODM è tipicamente descritta da una serie di matrici OD, definita per ogni modo di trasporto e per ogni intervallo temporale.

In generale, $D_{o,d,m,h}$ rappresenta la domanda di trasporto OD riferita a viaggi effettuati in un intervallo temporale h dalla zona di origine o alla zona di destinazione d con la modalità di trasporto m con i dati che possono essere disponibili da un censimento, da un'indagine o da una combinazione di un modello di distribuzione ed un modello di scelta modale.

| Mode | 7h-8h | 8h-9h | ⋯ |
|---|---|---|---|
| car | $D_{1,2,car,7} = 100$ | $D_{1,2,car,8} = 80$ | ⋯ |
| | $D_{3,2,car,7} = 243$ | $D_{3,2,car,8} = 155$ | ⋯ |
| | $D_{2,3,car,7} = 135$ | $D_{2,3,car,8} = 77$ | ⋯ |
| | ⋮ | ⋮ | ⋮ |
| bus | $D_{2,5,bus,7} = 56$ | $D_{2,5,bus,8} = 43$ | ⋯ |
| | $D_{3,4,bus,7} = 24$ | $D_{3,4,bus,8} = 14$ | ⋯ |
| | ⋮ | ⋮ | ⋮ |
| bike | $D_{1,5,bike,7} = 12$ | $D_{1,5,bike,8} = 10$ | ⋯ |
| | $D_{1,4,bike,7} = 24$ | $D_{1,4,bike,8} = 23$ | ⋯ |
| | ⋮ | ⋮ | ⋮ |

*Figura 1.8: Esempio di domanda per i modi di trasporto car, bus, bike*

L'obiettivo della disaggregazione è quello di generare i parametri dei viaggi $T_i = (a_{oi}, x_{oi}, a_{di}, x_{di}, m_i, t_{oi})i = 1 \dots N_T$ dalle informazioni sui flussi $D_{o,d,m,h}$. Quindi, ogni flusso $D_{o,d,m,h}$ produce esattamente $D_{o,d,m,h}$ viaggi che partono tutti da un arco della zona o, terminano su un arco della zona d, usano il mezzo m e partono nell'intervallo temporale [h, h + 1]. Il numero totale dei viaggi è dunque $N_T = \Sigma_{o,d} D_{o,d,m,h}$.

Per prepararsi alla disaggregazione è necessario identificare gli archi interni alle zone e calcolare la probabilità che ogni arco sia quello di partenza o di arrivo dei viaggi. In generale, si trovano $n_j$ archi all'interno la zona j, i quali rappresentano gli archi di origine e destinazione dei viaggi. All'interno di una zona, alcuni archi ricevono più partenze/arrivi rispetto ad altri archi della zona: in questo senso la probabilità $p_{jk}$ definisce la quota percentuale dei viaggi che partono/arrivano sull'arco $a_{jk}$ nella zona j (la somma delle probabilità $p_{jk}$ su tutti gli archi nella zona è uguale a 1). Le probabilità che i viaggi partano o arrivino in un determinato arco riflettono il numero di potenziali luoghi di origine o destinazione nelle vicinanze ad esso (per esempio case, uffici, industrie, scuole), con questi ultimi che possono essere stimati da alcuni attributi degli archi:

- La probabilità $p_{jk}$ aumenta linearmente con la lunghezza $l_{kj}$ dell'arco $a_{jk}$ perchè una strada più lunga ha più probabilità di essere un'origine o una destinazione di un viaggio.

- La probabilità $p_{jk}$ dipende dal tipo di arco, infatti le strade con priorità 3-6 hanno una maggiore probabilità di essere scelte perchè hanno tipicamente la più elevata densità di abitazione in una zona urbana, le strade con una più bassa priorità (<3) sono prevalentemente sentieri con poche abitazioni nei pressi, mentre strade con un'alta priorità (>6) sono stradoni o autostrade e quindi non sono idonee ad essere origine o destinazione di un viaggio.

La procedura di disaggregazione di ogni flusso $D_{o,d,m,h}$ in $T_i$ viaggi è la seguente:

1. Determinazione degli archi di partenza $a_{oi}$ e di destinazione $a_{di}$ :
    - $N_{ok} = p_{ok}\ D_{o,d,m,h}$ dei $D_{o,d,m,h}$ viaggi che partono dall'arco $a_{ok}$.
    - $M_{dk} = p_{dk}\ D_{o,d,m,h}$ dei $D_{o,d,m,h}$ viaggi che arrivano sull'arco $a_{dk}$.

Le associazioni degli archi di partenza e di arrivo con i viaggi sono indipendenti l'una dall'altra.

2. Generazione della posizione di partenza e quella di arrivo:
    - La posizione di partenza $x_{oi}$ sull'arco $a_{oi}$ viene determinata da un generatore aleatorio che restituisce una posizione fra 0 e $L_{oi}$.
    - La posizione di arrivo $x_{di}$ sull'arco adi viene determinata da un generatore aleatorio che restituisce una posizione fra 0 e $L_{di}$.
3. Generazione del tempo di partenza: Il tempo $t_{oi}$ viene determinato da un generatore aleatorio che restituisce un tempo fra h e h + 1.

Una volta applicata questa procedura per ogni flusso $D_{o,d,m,h}$, vengono determinati i parametri di tutti i viaggi.

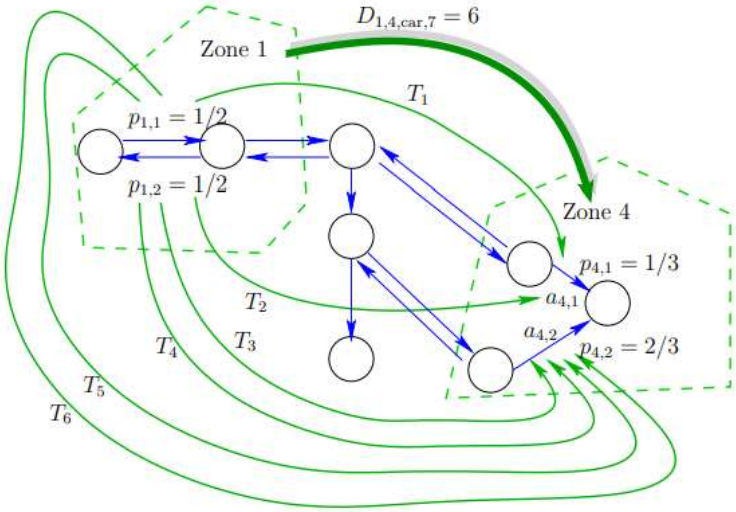


*Figura 1.9: Schema di disaggregazione dei flussi*

### 1.4.4.3 Metodo della domanda basata sulle attività

L'ideologia del metodo è quella di analizzare la fonte che genera il traffico nelle strade: ogni veicolo presente nell'infrastruttura, infatti, deriva da persone che necessitano di spostarsi da un'attività ad un'altra, in differenti intervalli temporali, percorsi e motivi di spostamento.

Stimare la domanda basata sulle attività significa provare a riprodurre i viaggi delle persone e dei veicoli nella rete stradale, partendo dall'analisi del diario di ogni persona, il quale riporta le attività svolte durante la giornata in differenti orari, luoghi, modi di trasporto scelti per muoversi tra le diverse attività ed i percorsi utilizzati.

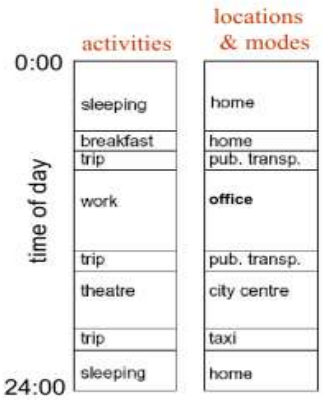


*Figura 1.10: Esempio di diario giornaliero di una persona*

Si tratta di un metodo molto performante, ma è necessaria un'enorme quantità di dati per costruire un diario per ogni elemento della popolazione. Tutti i metodi che vengono utilizzati si basano sulla creazione di una popolazione virtuale (ad ogni persona virtuale è associato il proprio diario personale) che corrisponda il più possibile alla realtà, e una volta definita completamente, è possibile creare gli itinerari, i piani ed effettuare la microsimulazione. Come il modello a 4 stadi, anche i processi principali dei modelli di domanda basata sulle attività possono essere suddivisi in 4 parti:

1. *Population synthesizer* → consiste nello stimare la popolazione nell'area di studio. Molti attributi come età media, veicoli posseduti, mezzi preferiti ecc. possono essere d'aiuto per le fasi successive e dato il livello di dettaglio dello studio e dei modelli disponibili è necessario avere dei dati di input molto dettagliati.

2. *Activity generator* → consiste nel definire le attività che ogni persona svolge durante la giornata, in determinati luoghi ed orari.

3. *Planner* → consiste nella generazione dei piani, con un piano di trasporto che indica il modo di trasporto utilizzato (eventualmente anche l'ID del veicolo da usare) e l'itinerario per spostarsi da un'attività all'altra. La generazione del piano è quindi un processo che unisce la fase di scelta modale e quella dell'assegnazione del percorso. Per ogni persona possono essere creati diversi piani.

4. Microsimulazione → consiste nell'esecuzione dei piani di tutte le persone della popolazione virtuale, tenendo conto delle interazioni che questi possano avere con gli altri utenti. Dato che per ogni persona vengono creati più piani, è necessario avere un criterio per capire quale sceglierebbe l'utente.

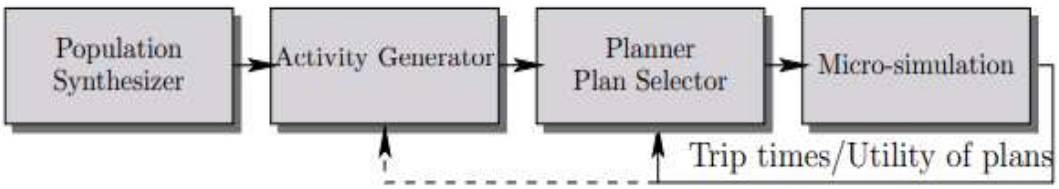


*Figura 1.11: Struttura dei modelli di domanda basati sulle attività*

I risultati ottenibili dalla fase della microsimulazione possono far emergere il fatto che alcune persone possano non essere soddisfatte del piano o addirittura delle attività scelte. La variazione di alcuni piani ed attività porta a dover risolvere un processo iterativo composto da una serie di microsimulazioni che dovrebbe convergere ad una situazione di equilibrio in cui ogni utente non ha più modo di cambiare il piano di trasporto, ottenendone un beneficio in termini di utilità. La situazione di equilibrio si suppone sia quella che riproduca le scelte che le persone effettuerebbero nella realtà. L'accortezza principale sta quindi nel calibrare al meglio la funzione che determina per ogni utente l'utilità che associa ai vari piani che potrebbe eseguire: è quindi necessario capire quali attributi (tempo, distanza, ecc.) siano i più "percepiti" dalle persone e che quindi pesano di più nella scelta del proprio piano.

In questi modelli, si rappresentano le sequenze delle attività di ogni persona durante l'arco di una giornata attraverso uno schema chiamato *activity pattern*. Gli esempi di schema di attività tipici sono: casa-lavoro-casa, casa-studio-casa, casa-scuola-lavoro, casa-shopping-casa, ecc. Ogni attività è rappresentata da:

- Una durata (minima, massima).
- Un tempo di inizio (minimo, massimo).
- Una posizione riferita all'edificio in cui viene svolta.

La popolazione virtuale che viene simulata deve corrispondere il più possibile alla realtà, utilizzando metodi che cercano di far confluire diverse tipologie di dati. Una volta compiuti i primi due passi del metodo di domanda basata sulle attività, la popolazione virtuale possiede i seguenti attributi:

- Edificio di residenza dove trascorre la notte.
- Edifici in cui svolge le attività (lavoro, studio).
- Dati socioeconomici (occupazione, numero di figli, membri della famiglia ecc.).
- Disponibilità dei veicoli (auto, bici, moto).
- Modo di trasporto preferito.
- Schema delle attività.

Attualmente, l'ambiente hybridPY può generare una popolazione virtuale e le relative attività in due diverse modalità: la generazione basata sul budget temporale e la generazione da matrici OD. Entrambi assumono che la distribuzione dei diversi schemi delle attività sulla popolazione sia conosciuta e forniscono attualmente solo una prima coppia di attività tra le quali viene compiuto uno spostamento.

Un piano, invece, nel contesto della microsimulazione, indica per ogni persona gli spostamenti (incluso il luogo e tempo di partenza, luogo di arrivo e modo di trasporto) per muoversi da un'attività all'altra e consiste in una sequenza di fasi, chiamate *stages*. Le informazioni contenute in uno stage assomigliano a quelle contenute in un viaggio oppure in un itinerario, ma è qualcosa di più generale e ha i seguenti attributi:

- Il tipo di *stage* (un'attività come guidare l'auto, andare in bici, andare a piedi, andare in autobus).
- Un tempo d'inizio stimato $t_{o,j}$.
- Un tempo finale stimato $t_{o,j}$.
- Una località nel caso in cui il tipo di stage sia un'attività, oppure arco e posizione di inizio ($a_{o,j}$, $x_{o,j}$) e arco e posizione finale ($a_{d,j}$, $x_{d,j}$) nel caso in cui il tipo di *stage* sia uno spostamento.
- Informazioni sul veicolo utilizzato nel caso in cui un veicolo sia coinvolto.

Per essere più precisi, nel caso di un viaggio in auto/bici, si indicano anche l'ID del parcheggio di partenza/arrivo e nel caso di un bus la linea e le fermate di arrivo e partenza. Essendo un piano visto come serie di *stages*, si riesce quindi a descrivere anche gli spostamenti multimodali più complessi.

Un altro concetto molto importante nella domanda è la strategia, la quale si differenzia dal modo di trasporto poiché essa è una combinazione di modi di trasporto che consentono l'utilizzo di uno di essi. Quando gli utenti identificano il piano per spostarsi da casa a lavoro o altri luoghi, non prendono in considerazione tutte le possibilità che la rete stradale ed i servizi di trasporto offrono: le persone stabiliscono certe strategie per redigere uno o più piani, per poi scegliere quello a loro più vantaggioso. Le strategie praticabili per un individuo sono prevalentemente determinate dalle sue caratteristiche

socioeconomiche, dalla proprietà del veicolo e dalla posizione delle attività. Tali considerazioni consentono di restringere la scelta strategica prima di eseguire simulazioni.

Le strategie attualmente disponibili per la fase di costruzione del piano sono:

- *Pedestrian Strategy* → l'utente cammina tra le strutture legate alle attività che deve svolgere o più precisamente tra i punti degli archi più vicini alle rispettive strutture.

- *Car strategy* → l'utente cammina verso il parcheggio della sua auto, poi guida fino al parcheggio vicino alla struttura di destinazione e infine cammina dal parcheggio alla struttura di destinazione.

- *Bike strategy* → l'utente pedala tra l'origine e la destinazione. Nel caso in cui l'arco vicino ad una struttura non abbia accesso alla bicicletta, cammina per la distanza tra la struttura e l'accesso alla bicicletta più vicino.

- *Public transport strategy* → l'utente cammina verso la fermata del trasporto pubblico più vicina. Alla fermata, aspetta la linea di trasporto pubblico che lo porta più velocemente alla sua destinazione e, una volta sceso dal bus, cammina verso la struttura di destinazione. Ci possono essere anche trasferimenti intermedi, in cui l'utente scende ad una fermata, cammina verso un'altra fermata (se necessario) e prende un'altra linea di trasporto pubblico.

- *Bike + public transport strategy* → si tratta di una combinazione tra la strategia *bike* e la strategia di trasporto pubblico. L'utente utilizza, se possibile, la bici per raggiungere la fermata del trasporto pubblico più vicina per poi utilizzare i mezzi pubblici. L'ultima tappa dalla fermata del trasporto pubblico alla struttura di arrivo viene di nuovo eseguita in bicicletta, quando possibile.

In realtà, ogni persona ha la possibilità di utilizzare diversi modi di trasporto in base a quali tipologie di mezzi possiede e può sempre andare a piedi fino a destinazione, andare in auto, moto o bici nel caso in cui le possiede e con il bus se è presente una rete di trasporto pubblico. Per ogni persona vengono generati tanti piani quante sono le strategie di cui può usufruire. Una volta individuato il mezzo di trasporto, l'arco di origine $a_{o,j}$ e

destinazione $a_{d,j}$, la durata dello *stage* j viene determinata attraverso il routing (assegnazione del percorso), dato che, una volta determinata la sequenza degli archi $r_k$ fra origine e destinazione, è possibile individuare anche il tempo di percorrenza $g_k$.

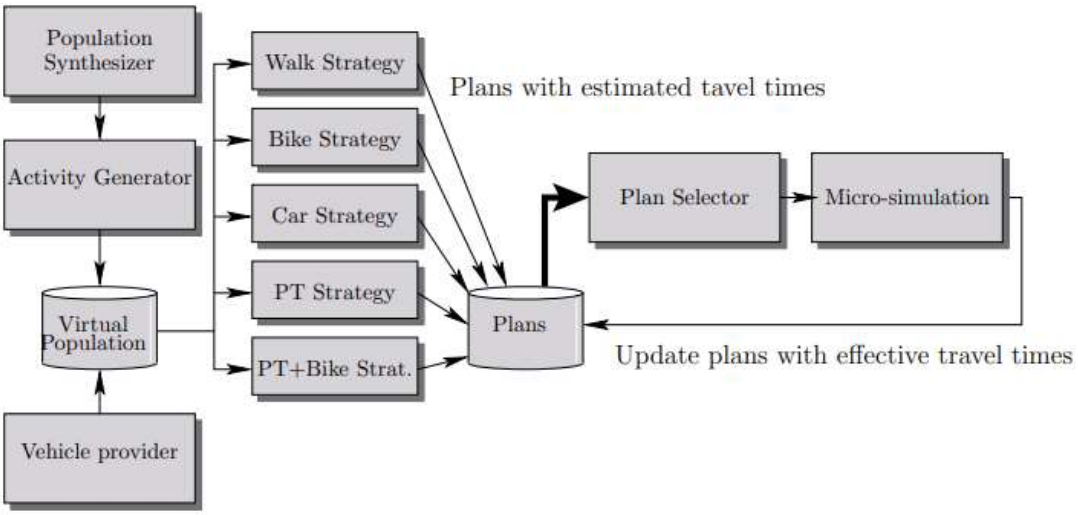


*Figura 1.12: Schema per la generazione della popolazione virtuale*

Nel programma hybridPY è possibile costruire un metodo di assegnazione della miglior strategia alle persone in rete congestionata stocastico e dinamico, una volta individuati i piani che possono utilizzare (in base alle strategie per loro fattibili). La metodologia dinamica cerca di riprodurre al meglio la modalità di scelta e quindi la linea di pensiero adottata dagli utenti: andando ad analizzare il generico utente abitudinario che in una generica città deve decidere quale modo di trasporto utilizzare per andare al lavoro, prova inizialmente ad utilizzare una strategia (tra quelle fattibili) utilizzando i mezzi che potrebbero convenirgli di più. Una volta provate diverse strategie per andare al lavoro (e dopo aver misurato i tempi effettivi dal microsimulatore), esso sceglie quello che, in base ad alcuni parametri (tempo di viaggio, ma anche ritardi, comfort, ecc.), gli risulta più conveniente. In particolare, dopo aver creato la popolazione virtuale ed i piani fattibili per ogni persona (con i rispettivi tempi stimati $t_i$), il programma segue i seguenti passi:

1. Assegna a tutte le persone il piano con il tempo stimato minore, dopo aver modificato aleatoriamente i tempi sommandoli ad una variabile aleatoria a media nulla e con deviazione standard pari a $\alpha t_i$.

2. Esegue la microsimulazione dei piani registrando i tempi effettivi di percorrenza per ognuno di essi.
3. Confronta per una percentuale β del numero di persone totali i tempi effettivi (dei piani già eseguiti) e stimati (per i piani non ancora eseguiti) dei rispettivi piani fattibili, selezionando quello con il tempo minore dopo averli sommati ad una variabile aleatoria a media nulla e con deviazione standard pari a $\alpha t_i$.
4. Esegue di nuovo la microsimulazione dei piani registrando i tempi effettivi di percorrenza dei piani eseguiti.
5. Riproduce gli ultimi due punti fino a quando ogni persona ha consolidato la scelta del proprio piano, che si suppone essere quello che utilizzerebbe nella realtà.

Il coefficiente β serve a condurre l'algoritmo all'equilibrio: se tutte le persone valutassero l'opzione di cambiare piano di trasporto dopo un'esperienza su rete congestionata, la congestione si sposterebbe semplicemente da un'altra parte. Il coefficiente α, invece, serve a considerare l'aleatorietà dei tempi stimati e di percorrenza nella rete dovuta ad eventuali interazioni e a spingere le persone a provare piani con tempi simili a quello più veloce.

Basare la scelta della strategia in base ai tempi effettivi dalla microsimulazione risulta molto preciso poiché grazie ad essa è possibile analizzare tutto ciò che può influire sul viaggio: interazione con gli altri utenti, attese ai semafori, code, attesa dell'autobus, tempo per arrivare al parcheggio o alla fermata. Chiaramente, potrebbero essere utilizzate altre variabili oltre al solo tempo di percorrenza (come ritardi o indicatori di convenienza), anche se esso rappresenta l'attributo per eccellenza nel condizionare la scelta delle persone.

## 1.5 SCELTA DEL TIPO DI SIMULAZIONE DA UTILIZZARE

Un aspetto importante da considerare è il modo in cui i flussi e le capacità di trasporto vengono gestiti nei modelli mesoscopici e microscopici. Con i modelli mesoscopici utilizzati (ad esempio MATSim), la capacità del collegamento viene pre-calcolata in base

agli attributi del collegamento, come lunghezza, velocità consentita e numero di corsie e durante la simulazione l'algoritmo consente ai flussi di collegamento di aumentare fino a questo limite di capacità. Dopo aver raggiunto il limite, il flusso di collegamento si satura e rimane a questa capacità di trasporto, indipendentemente dai veicoli presenti nei collegamenti a monte. Ciò consente di simulare uno scenario con una domanda ridotta: ad esempio, uno scenario può essere simulato con solo il 10% dei veicoli, il che risulta esattamente nel 10% dei flussi reali del collegamento (viene fatto proprio questo con la domanda di trasporto), ma questo è possibile solo se i limiti di capacità pre-calcolati sono ridotti anche essi di un fattore di 10. Tutto questo si traduce in un guadagno di un ordine di grandezza in termini di velocità computazionale.

Al contrario, il modello microscopico mostra un comportamento completamente diverso poiché la capacità non è definita esplicitamente, ma è una funzione implicita dei molti attributi dei collegamenti, delle corsie e dei veicoli. In forte contrasto con la simulazione mesoscopica, dove il flusso rimane al limite di capacità, la simulazione microscopica mostra che il flusso del collegamento scende quasi a zero con un conseguente collasso del traffico, dopo aver raggiunto il limite di capacità implicita. Sebbene questo possa essere un effetto desiderato, poiché assomiglia alla realtà, la modellazione e la calibrazione di tutti gli attributi e i parametri è generalmente difficile, specialmente con reti di grandi dimensioni.

In sostanza, la simulazione mesoscopica quindi non sperimenta collassi di traffico quando i flussi raggiungono i limiti di capacità (però possono verificarsi rigurgiti a monte), ma i limiti di capacità impostati sono probabilmente diversi da quelli della realtà. Con la microsimulazione, invece, i limiti di capacità sono potenzialmente più vicini alla realtà, ma se gli attributi e i parametri non sono modellati correttamente, il collasso del traffico si verifica in momenti e luoghi irrealistici. Inoltre, con una microsimulazione, generalmente non è possibile ridurre la domanda per abbassare il carico computazionale, come nel caso delle simulazioni mesoscopiche.

Sebbene non esista una soluzione definitiva a questo problema, è chiaro come i due modelli abbiano proprietà complementari che possono essere esplorate in un approccio *ibrido* (mesoscopico-microscopico): a questo proposito, viene utilizzato per lo studio in oggetto il software chiamato hybridPY, il quale consente la microsimulazione di un'area

all'interno di un modello di domanda mesoscopica più ampio, e può eseguire simulazioni mesoscopiche e microscopiche dello stesso scenario dalla stessa piattaforma scambiando risultati tra i due modelli.

## 1.6 INDIVIDUAZIONE DEGLI ITINERARI E *ROUTING*

Il problema dell'individuazione di un unico itinerario tra origine e destinazione, chiamato *routing*, rappresenta un aspetto molto importante della simulazione poiché determina i flussi veicolari nella rete ed i risultanti impatti ambientali. La determinazione degli itinerari può essere effettuata prima o durante la microsimulazione, oppure fra due simulazioni all'interno di un processo iterativo.

Tutti metodi hanno in comune il fatto che determinano una sequenza di archi $r_k$ ben precisa fra un insieme possibili collegamenti $R_{ao,ad}$ tra l'arco di origine $a_o$ e l'arco di destinazione $a_d$ di un qualsiasi viaggio. Quindi, ogni sequenza $r_k$ con $k \in R_{ao,ad}$ inizia con l'arco $a_o$ e finisce con l'arco $a_d$ ovvero $r_k = [a_o, \ldots, a_d]$. In generale, l'algoritmo cerca di minimizzare il tempo di viaggio fra $a_o$ e $a_d$, ma per questo scopo occorre stimare i tempi di percorrenza $c_a$ per ogni arco a (il tempo di viaggio $g_k$ della sequenza di archi presenti in $r_k$ corrisponde alla somma dei tempi di percorrenza degli archi stessi). Quindi, la funzione del costo di un itinerario diventa $g_k = \Sigma\, c_a$. I costi $g_k$ di tutti i percorsi, con $k \in R_{ao,ad}$, possono essere raggruppati in un vettore di costi $g_{ao,ad} = [\ldots, gk, \ldots]$.

### 1.6.1. Percorso di minimo costo (algoritmo di Dijkstra)

Il *routing* più diffuso nelle simulazioni del traffico è quello che considera il percorso con il minimo tempo di percorrenza tra l'arco di origine e arco di destinazione, ignorando gli altri possibili percorsi, anche se il loro tempo di viaggio è leggermente superiore a quello che rappresenta il minimo tempo di percorrenza in assoluto. L'algoritmo che permette questa operazione è conosciuto come algoritmo di Dijkstra. Formalmente, il percorso con il minimo tempo di percorrenza fra l'arco di origine $a_o$ e arco di destinazione $a_d$ è la

sequenza di archi $r_j$ se il suo rispettivo costo del viaggio $g_j$ costituisce il minimo fra tutte le possibili sequenze $k \in R_{ao,ad}$, ovvero $g_j = \min(g_{ao,ad})$ oppure $j = \text{argmin}(g_{ao,ad})$.

### 1.6.2. Scelta aleatoria del percorso

La scelta aleatoria del percorso rappresenta un metodo dove la probabilità di scegliere un determinato percorso aumenta con il diminuire del rispettivo tempo di percorrenza, così facendo anche gli itinerari con un tempo di viaggio maggiore di quello del percorso con il minimo tempo di percorrenza hanno una probabilità non nulla di essere scelti. Questo metodo di scelta riflette più la realtà, in quanto itinerari con simili tempi di viaggio ottengono una simile probabilità di essere scelto.

In pratica, la scelta del percorso si svolge in due passi:

1. Prima viene associata ad ogni sequenza di archi $r_j$ una probabilità $p_j$ che dipende dal proprio tempo di percorso $g_j$, ma anche dai tempi di viaggio di tutte le altre alternative in $g_{ao,ad}$. Ci sono varie possibilità per calcolare la probabilità ed una di queste prende il nome di *C-Logit*, il quale è preferibile al *Logit multinomiale* dato che può essere usato anche con itinerari fortemente sovrapposti. Inoltre, il *C-Logit* mantiene il vantaggio di fornire una soluzione esplicita per le probabilità della scelta di un itinerario.
2. Una volta che le probabilità $p_k$ sono determinate, il percorso $r_j$ effettivo viene scelto dall'insieme $R_{ao,ad}$ attraverso un generatore aleatorio che permette di identificare la sequenza $r_j$ rispettando le probabilità $p_k$.

Esistono poi tre approcci diversi per determinare gli itinerari dei veicoli: il *pre-routing*, il *Dynamic User Equilibrium* ed il *routing dinamico*.

### 1.6.3. *Pre-routing*

Con il *pre-routing*, il tempo di percorrenza degli archi $c_a$ viene stimato prima di effettuare la microsimulazione, facendo l'ipotesi di rete non congestionata ed assumendo che tutti i veicoli nella rete non debbano rallentare a causa del traffico. In tal caso, si assume che la

velocità del veicolo sia vincolata solamente alla velocità massima consentita sullo specifico arco, quindi il tempo di percorrenza dell'arco a vale: $c_a = l_a / V_a$, dove $l_a$ e $V_a$ rappresentano rispettivamente la lunghezza e la velocità massima consentita sull'arco a. Dopo che i tempi di percorrenza $c_a$ sono stati stimati, è possibile determinare gli itinerari $R_i$ con il minor tempo di percorrenza per tutti veicoli e iniziare la microsimulazione.

Ovviamente, l'ipotesi di rete non congestionata è ragionevole solo per scenari con un basso livello di traffico: nel caso in cui ci fossero congestioni, l'ipotesi non sarebbe più corretta e bisognerebbe considerare altri approcci. L'approccio del *pre-routing* ha delle caratteristiche simili all'assegnazione *all or nothing* (tutto o niente) classica.

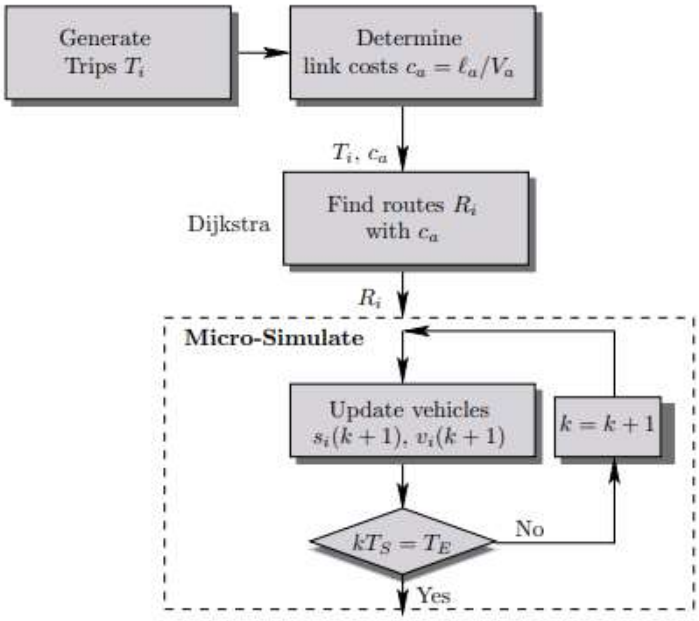


*Figura 1.13: Schema dell'algoritmo pre-routing*

### 1.6.4. *Dynamic User Equilibrium (DUE)*

Il *Dynamic User Equilibrium* è un approccio iterativo che effettua una successione di microsimulazioni in cui i tempi di percorrenza degli archi vengono aggiornati dopo ogni iterazione, con l'obiettivo che essi convergano verso un valore di equilibrio (appunto *l'User Equilibrium*). Tiene conto, quindi, dell'effetto di congestione e della possibilità per gli utenti di cambiare l'itinerario.

L'idea base dell'approccio *DUE* è quella che ogni utente cerchi sempre di minimizzare il suo tempo di viaggio, confrontando i tempi di percorrenza delle varie alternative, ma anche quelli derivanti dalla sua esperienza precedente (di un giorno prima o di una settimana prima), ipotizzando che i tempi di percorrenza di ogni tratto stradale che contribuiscono alla scelta dell'itinerario corrispondano a quelli registrati dalla sua ultima esperienza, se sono stati percorsi. Si suppone comunque che l'utente conosca i tempi di percorrenza di tutte le strade che non ha percorso.

Dato che tutti i veicoli adattano il proprio itinerario per minimizzare i tempi di viaggio, le congestioni possono spostarsi su altri tratti, variandone i rispettivi tempi di percorrenza. Quindi, il giorno successivo può accadere che gli utenti si imbattano in una nuova condizione di traffico e cerchino di aggiustare nuovamente il loro itinerario. La teoria del *DUE* dice che esiste una situazione, chiamata di equilibrio, nella quale gli utenti non riescano più in nessun modo a ridurre il tempo di viaggio anche cambiando l'itinerario, e la sfida dell'algoritmo è perciò quella di raggiungere o convergere verso questa situazione il più presto possibile.

Nell'ambito della microsimulazione, l'algoritmo iterativo del *DUE* inizia prima della prima iterazione (k=0) con i tempi di percorrenza $c_{a,0}$ che vengono stimati utilizzando il metodo del *pre-routing*. Dopodiché, inizia la prima iterazione con gli *step* principali di ogni iterazione k>0 che sono:

1. *Routing* → determinazione degli itinerari $R_{i,k}$, i = 1…$N_V$ dell'iterazione k basandosi sui tempi $c_{a,k-1}$ dell'iterazione precedente (per gli archi utilizzati) e su quelli stimati.
2. Microsimulazione → fornendo in input gli itinerari $R_{i,k}$, si ottengono come output i nuovi tempi (medi) di percorrenza sugli archi simulati $c_{a,k}$.
3. Test di arresto/convergenza → esistono varie possibilità per verificare se i tempi di percorrenza hanno raggiunto un equilibrio, una di queste sta nel verificare se i tempi $c_{a,k}$ non sono variati in modo significativo rispetto all'ultima iterazione. Di conseguenza, l'algoritmo termina quando si verifica che la sommatoria della differenza tra $c_{a,k}$ e $c_{a,k-1}$ in valore assoluto divisa per il numero di archi $N_a$ è inferiore a una soglia $\gamma$ sufficientemente bassa.

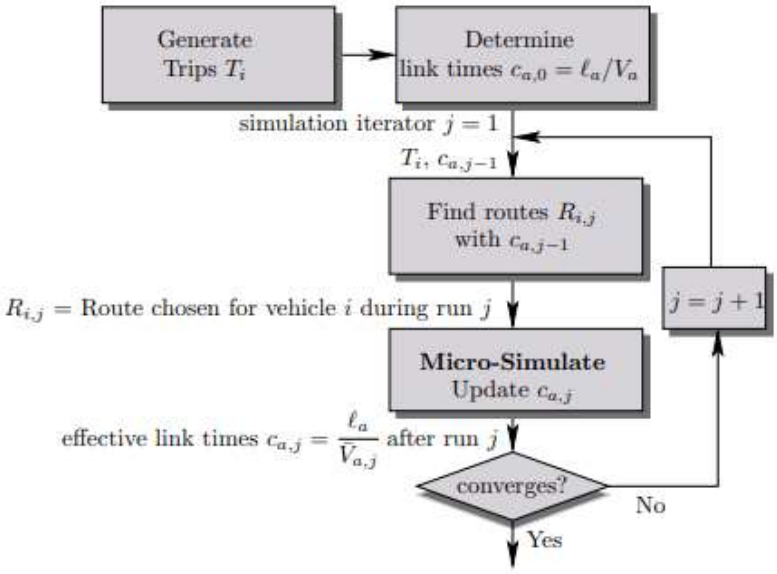


*Figura 1.14: Schema dell'algoritmo DUE*

Al fine di garantire una convergenza dell'algoritmo è necessario che non troppi veicoli cambino l'itinerario dopo ogni iterazione. Per tale motivo, non è possibile usare l'algoritmo di Dijkstra per il *routing* (nel primo step) perchè tutti veicoli si sposterebbero sul percorso di minimo tempo dopo ogni iterazione e ci sarebbe una oscillazione permanente. È quindi necessario utilizzare un algoritmo che scelga un itinerario con una probabilità proporzionale al suo tempo di viaggio stimato, come per esempio il *C-Logit* o *Gawron*, disponibile in SUMO.

L'approccio del *DUE* è paragonabile all'*User Equilibrium* della teoria classica basata sui flussi sugli archi. Purtroppo, questo metodo richiede per ogni iterazione lo svolgimento di una microsimulazione e dunque un tempo di calcolo prolungato (dipendente dalla grandezza dello scenario).

### 1.6.5. *Routing dinamico*

Con il *routing dinamico*, chiamato nella terminologia di SUMO *one shot assignment*, il *routing* viene effettuato durante l'esecuzione della microsimulazione e non è quindi necessario determinare gli itinerari $R_i$ prima poiché basta specificare solamente i viaggi $T_i$ come input. L'idea alla base di questo metodo è stata dedotta da un processo

decisionale reale: l'autista di una macchina sceglie il suo percorso nell'istante in cui inizia a guidare considerando la situazione del traffico in quel momento (attraverso la radio o internet oppure direttamente dalla sua esperienza).

Tecnicamente, il procedimento del *routing dinamico* è implementato in questo modo: nel momento in cui il veicolo i viene inserito nella simulazione al tempo $t_{oi}$, il routing del percorso di minimo tempo viene effettuato con l'algoritmo di Dijkstra dall'arco di origine $a_{oi}$ all'arco di destinazione $a_{di}$, come per l'approccio *pre-routing*. I tempi di percorso degli archi $c_{a,k}$, invece, vengono aggiornati in ogni istante t, considerando la velocità media attuale dei veicoli $v_a(t)$ su ogni arco a:

- $c_{a,k} = l_a / V_a$ per k=0;
- $c_{a,k} = c_{a,k-1}*\alpha + [l_a / V_{a,k}]*(1 - \alpha)$ per k>0,

dove $c_{a,k-1}$ è il tempo medio di percorrenza sull'arco a nello step precedente e $0<\alpha<1$ è un parametro che indica con quale peso tener conto dei costi ottenuti all'iterazione precedente ed è necessario per evitare troppi sbalzi dei valori di $c_{a,k}$. Il *routing dinamico* non dà risultati identici al *Dynamic User Equilibrium*, ma tiene comunque conto di una rete congestionata e non necessita di iterazioni, risparmiando molto tempo avendo un costo computazionale molto inferiore.

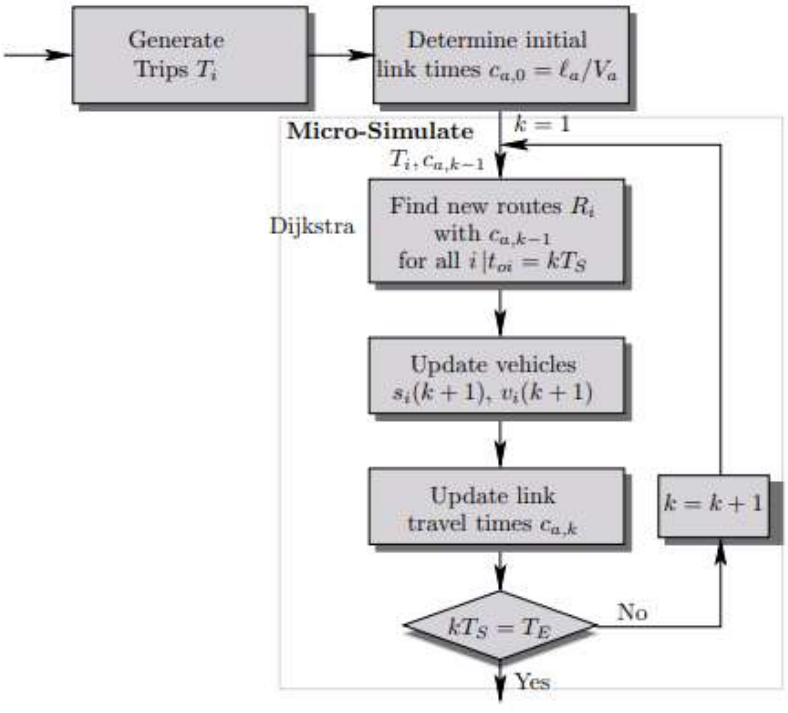


*Figura 1.15: Schema dell'algoritmo routing dinamico*

## 1.7. VISUALIZZAZIONE CON UNREAL ENGINE

Dopo aver sviluppato una microsimulazione del traffico, si può arricchire ulteriormente lo scenario con una visualizzazione urbana della città con tanto di veicoli circolanti annessi.

In questo senso, è stato utilizzato il software Unreal Engine, un potente motore grafico sviluppato da Epic Games, utilizzato principalmente per la creazione di videogiochi. Originariamente rilasciato nel 1998, Unreal Engine ha subito numerose evoluzioni e aggiornamenti, con la versione più recente, Unreal Engine 5, lanciata nel 2021. Questo motore è noto per le sue capacità di *rendering* in tempo reale, che permettono la creazione di ambienti altamente dettagliati con illuminazione e ombreggiature avanzate. Le tecnologie introdotte in Unreal Engine 5, come Lumen e Nanite, offrono rispettivamente un'illuminazione globale dinamica e una gestione efficiente di geometrie estremamente dettagliate.

Oltre ai videogiochi, Unreal Engine trova applicazione in molti altri settori: è utilizzato nella visualizzazione architettonica per creare modelli interattivi di edifici e spazi, nella produzione cinematografica per creare effetti visivi e ambienti virtuali, nelle simulazioni di addestramento per creare ambienti simulati per la formazione e l'educazione e nello sviluppo di esperienze immersive in realtà virtuale e aumentata.

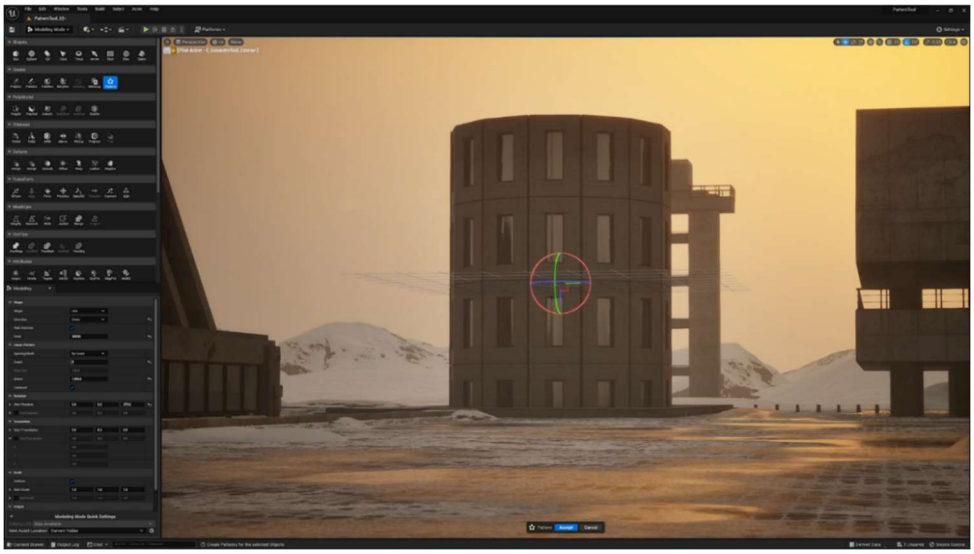

*Figura 1.16: Interfaccia Unreal Engine*

Per quanto riguarda la visualizzazione urbana, vi sono numerosi possibili impieghi:

- Creazione di modelli 3D dettagliati → consente di importare modelli 3D dettagliati di edifici, strade, infrastrutture e spazi pubblici creati in software CAD o programmi di modellazione 3D come AutoCAD, SketchUp, Revit, Blender e 3ds Max. Questi modelli possono essere arricchiti con *texture* realistiche, materiali e illuminazione avanzata per ottenere una rappresentazione visivamente accurata dell'ambiente urbano.
- *Rendering* in tempo reale → permette agli utenti di esplorare l'ambiente urbano in modo interattivo. Con tecnologie come Lumen, offre un'illuminazione globale dinamica che migliora ulteriormente il realismo delle visualizzazioni e gli utenti possono vedere come l'illuminazione naturale e artificiale cambia nel corso della giornata o in base alle condizioni meteorologiche.
- Interazione e navigazione → permette di creare esperienze interattive dove gli utenti possono navigare attraverso la città virtuale, camminare o volare sopra di essa. È possibile aggiungere punti di interesse con informazioni dettagliate, simulazioni di traffico e movimenti pedonali, e persino integrare interazioni come aprire porte o attivare scenari specifici.
- Simulazione di scenari urbanistici → può essere utilizzato per simulare diversi scenari urbanistici, come l'impatto di nuove infrastrutture sul traffico, l'analisi del flusso pedonale, la gestione delle emergenze e la pianificazione delle risorse. Queste simulazioni aiutano a prendere decisioni informate basate su visualizzazioni realistiche e dati dinamici.
- Integrazione con dati GIS → può integrare dati GIS (Geographic Information System) per visualizzare informazioni geospaziali in 3D, il che lo rende utile per visualizzare dati topografici, confini di proprietà, zone di rischio, densità di popolazione e altre informazioni rilevanti per la pianificazione urbana. I dati GIS possono essere importati direttamente in Unreal Engine per creare mappe interattive e visualizzazioni immersive.


*Figura 1.17: Visualizzazione 3D di una città dall'alto sul software*

# 2. SOFTWARE UTILIZZATO: HYBRIDPY

## 2.1. SOFTWARE DI MICROSIMULAZIONE

Per la microsimulazione sono disponibili vari pacchetti commerciali e *open-source*. Tra i più utilizzati sono presenti:

- Vissim.
- Aimsun.
- Paramics.
- MATSim.
- Cube Dynasim.
- SUMO.

Le caratteristiche di questi software sono abbastanza simili essendo utilizzati per i medesimi scopi, ma presentano alcune differenze.

### 2.1.1. Vissim

Vissim, sviluppato da PTV Group, è un sofisticato strumento di simulazione del traffico, progettato per rappresentare in dettaglio le dinamiche complesse del traffico veicolare e pedonale nelle aree urbane. Grazie alla sua capacità di modellare il comportamento individuale di ogni veicolo e pedone, Vissim tiene conto di diversi fattori come velocità, accelerazione, decelerazione e manovre di sorpasso, oltre alle interazioni con gli altri utenti della strada. La sua interfaccia utente intuitiva semplifica la creazione, visualizzazione e analisi delle simulazioni di traffico, rendendo il software accessibile anche a chi non è esperto.

Una delle sue caratteristiche principali è la capacità di modellare complessi schemi semaforici e configurazioni delle intersezioni, supportando la simulazione multimodale

che include veicoli privati, trasporto pubblico, biciclette e pedoni. Inoltre, è altamente personalizzabile tramite l'uso di script e API, consentendo agli utenti di adattare il software alle loro specifiche esigenze di simulazione e offre funzionalità avanzate di visualizzazione in 3D, facilitando la comprensione delle dinamiche del traffico.

Questo software trova ampio impiego nella pianificazione urbana, nella gestione del traffico, nella progettazione dei sistemi di trasporto pubblico e nella valutazione dell'impatto ambientale. È anche utilizzato in progetti di ricerca e sperimentazione per studiare il comportamento del traffico e testare nuove tecnologie di trasporto intelligente.

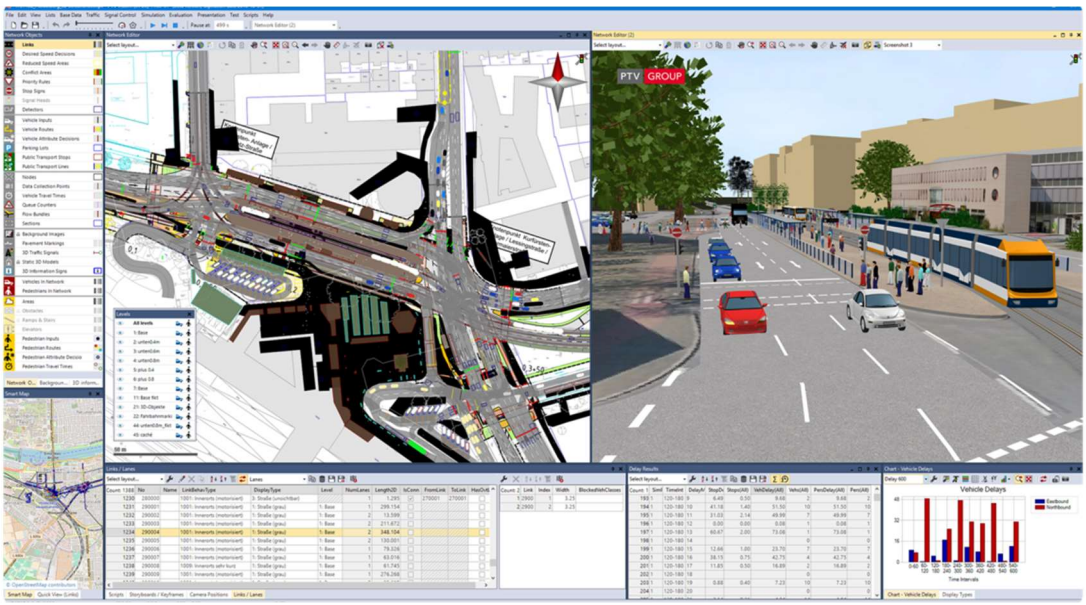


*Figura 2.1: Interfaccia grafica PTV Vissim*

### 2.1.2. Aimsun

Aimsun, sviluppato da Siemens Mobility, è un software di simulazione del traffico ampiamente impiegato in contesti urbani e stradali complessi che consente di modellare dettagliatamente il comportamento dei veicoli e dei pedoni, considerando una serie di variabili come la velocità, l'accelerazione e le interazioni tra gli utenti della strada. La sua interfaccia utente intuitiva semplifica notevolmente la creazione, la visualizzazione e l'analisi delle simulazioni di traffico, rendendo il software accessibile anche a utenti meno esperti.

Aimsun supporta la simulazione di diverse modalità di trasporto, tra cui veicoli privati, trasporto pubblico, biciclette e pedoni, consentendo agli utenti di analizzare le interazioni tra questi diversi modi di trasporto. Inoltre, è altamente personalizzabile e può essere adattato alle specifiche esigenze di simulazione degli utenti.

Questo software trova impiego in molteplici settori, dalla pianificazione urbana e infrastrutturale alla gestione del traffico in tempo reale ed è utilizzato anche nella progettazione di sistemi di trasporto pubblico e nella valutazione dell'impatto ambientale del traffico.

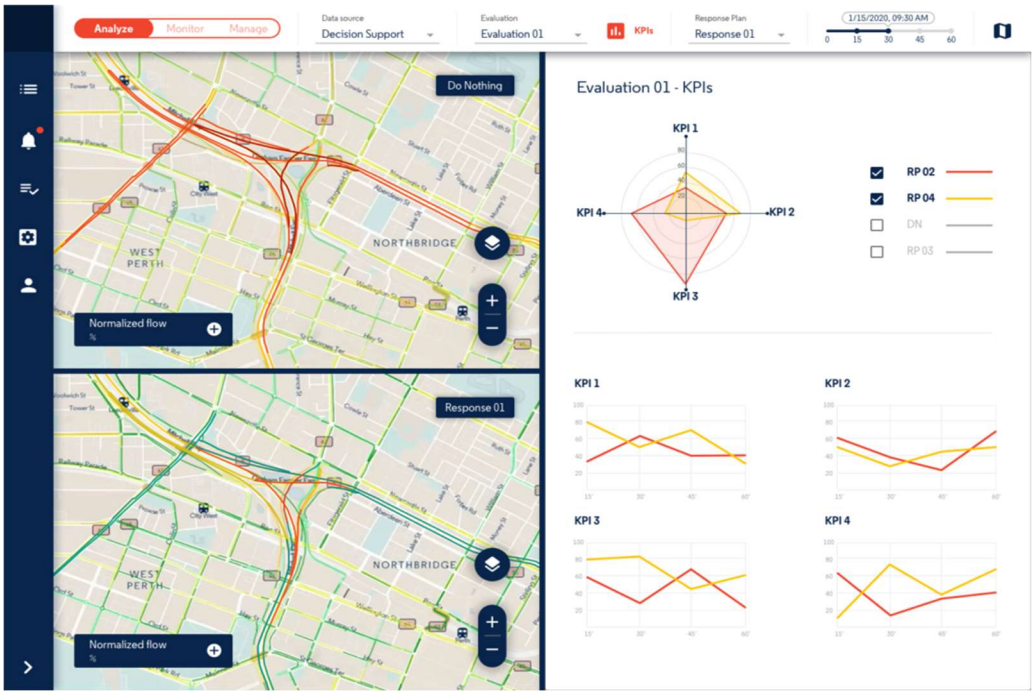


*Figura 2.2: Interfaccia grafica Aimsun*

### 2.1.3. Paramics

Paramics, sviluppato da SYSTRA, è un software di simulazione del traffico progettato per modellare il flusso veicolare in ambienti urbani e stradali complessi che grazie alla sua capacità di considerare variabili come velocità, accelerazione e interazioni tra gli utenti della strada, offre una simulazione dettagliata del comportamento individuale di ogni veicolo.

Paramics eccelle nella modellazione di schemi semaforici, configurazioni di intersezioni e flussi di traffico complessi, consentendo una valutazione accurata delle prestazioni e

della sicurezza stradale. La flessibilità del software consente anche di adattarlo alle esigenze specifiche di simulazione degli utenti e di integrarlo con altri strumenti di analisi del traffico.

Predisponendo la descrizione della rete stradale è possibile sviluppare un modello molto raffinato, in grado di considerare tutte le variabili (geometriche, dinamiche e cinematiche) che determinano i fenomeni di congestione. Oltre alle rappresentazioni grafiche, Paramics Discovery permette di ottenere report sia statistici che numerici dal livello più generale di "area complessiva" fino al livello più specifico del "singolo veicolo per singolo istante di marcia".

Le sue applicazioni spaziano dalla pianificazione urbana e infrastrutturale alla gestione del traffico in tempo reale, dalla progettazione di sistemi di trasporto pubblico alla valutazione dell'impatto ambientale del traffico e viene utilizzato anche in progetti di ricerca e sperimentazione.


*Figura 2.3: Simulazione tridimensionale Paramics*

### 2.1.4. MATSim

MATSim, l'acronimo di Multi-Agent Transport Simulation, è un framework open-source per la simulazione del trasporto basato su agenti. Questo approccio innovativo consente

di modellare il comportamento dei singoli attori nel sistema di trasporto, come veicoli, pedoni e ciclisti, consentendo una rappresentazione accurata delle dinamiche di mobilità. Ogni entità è rappresentata come un agente autonomo che prende decisioni in base al suo ambiente circostante e agli obiettivi individuali consentendo di modellare in dettaglio le preferenze personali, gli orari di lavoro e i percorsi di viaggio dei singoli utenti.

Integra dati reali, come dati GPS e di conteggio del traffico, per calibrare e validare le simulazioni, garantendo così una maggiore precisione nei risultati. Inoltre, permette la modellazione della rete di trasporto stradale e pubblico, consentendo agli agenti di pianificare e ottimizzare i propri percorsi in base alle condizioni della rete e ai vincoli di viaggio.

Il modello stradale di MATSim è relativamente semplice e, per questo, ha una velocità di esecuzione elevata, rendendolo capace di condurre analisi per scenari molto grandi, coinvolgendo anche intere nazioni. Comunque sia, bici, pedoni e trasporto pubblico sono considerati in parallelo rispetto alla strada e non interagiscono con i veicoli nelle code in quanto gli sviluppatori hanno sostenuto che ciò non fosse necessario per il fatto che pedoni e biciclette non hanno problemi di capacità; ciò comporta però una notevole approssimazione della realtà.

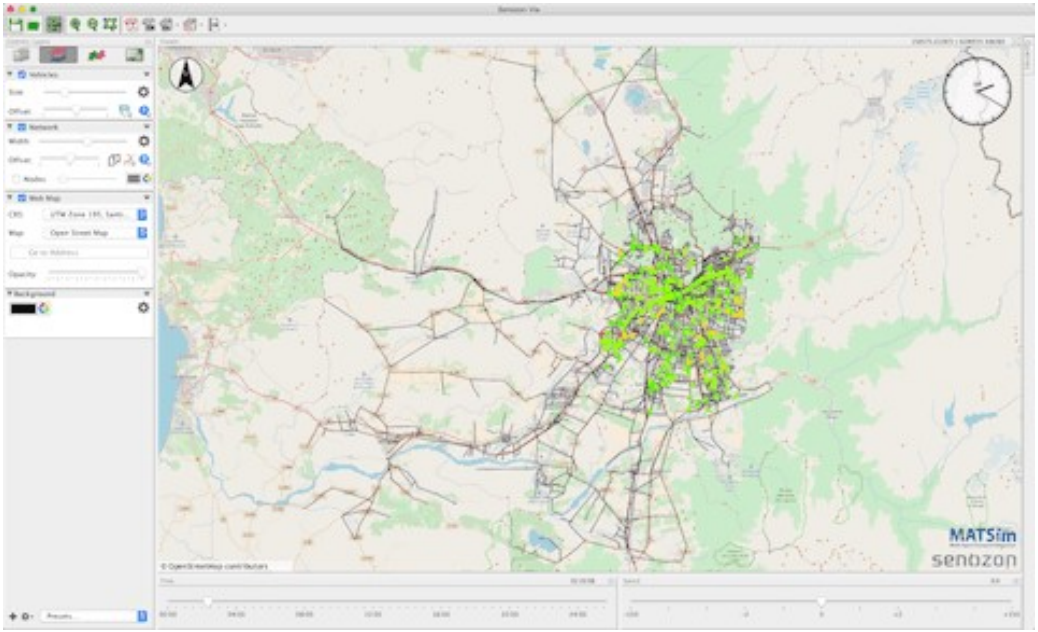


*Figura 2.4: Interfaccia grafica MATsim*

### 2.1.5. Cube Dynasim

Cube Dynasim è un software di simulazione del traffico sviluppato da Citilabs, progettato per modellare con precisione le dinamiche del traffico veicolare in contesti urbani e stradali complessi che offre una rappresentazione accurata delle interazioni tra veicoli, pedoni e infrastrutture stradali, permettendo agli utenti di valutare e ottimizzare le performance del sistema di trasporto.

Una delle caratteristiche distintive di Cube Dynasim è la sua capacità di simulare il comportamento individuale di ogni veicolo e pedone, tenendo conto di vari fattori come velocità, accelerazione, decelerazione e interazioni con gli altri utenti della strada. Questo consente una modellazione dettagliata delle condizioni del traffico e una valutazione approfondita delle prestazioni della rete stradale.

Il software offre un'interfaccia utente intuitiva che semplifica la creazione, la visualizzazione e l'analisi delle simulazioni di traffico, consentendo agli utenti di esplorare scenari alternativi e valutare l'impatto di interventi infrastrutturali e politiche di mobilità. Inoltre, supporta la simulazione di diverse modalità di trasporto, tra cui veicoli privati, trasporto pubblico, biciclette e pedoni, consentendo agli utenti di esaminare le interazioni tra questi diversi modi di trasporto.

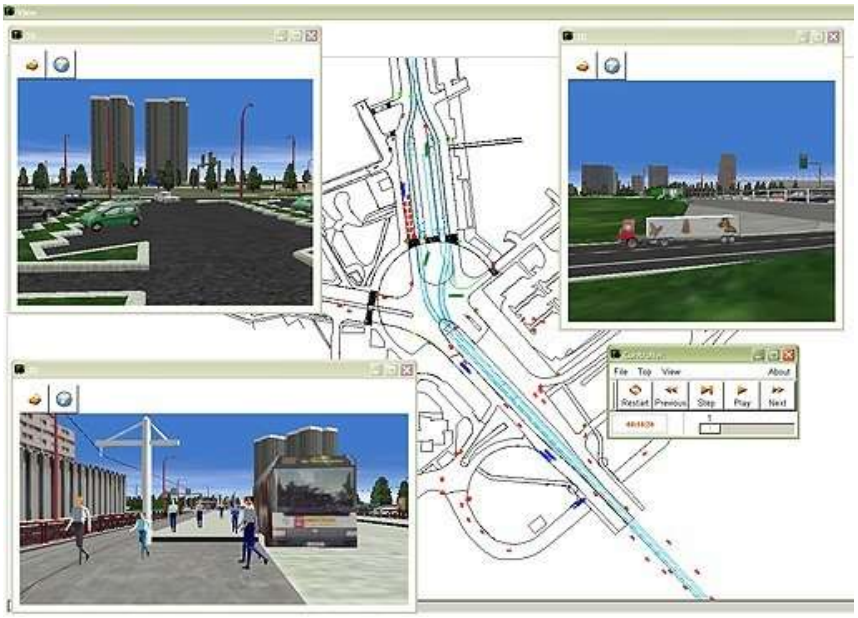

*Figura 2.5: Interfaccia 2D e 3D di Cube Dynasim*

### 2.1.6. SUMO

SUMO (*Simulation of Urban MObility*) è un software *open-source* progettato per la simulazione del traffico e della mobilità urbana sviluppato dal DLR, il Centro Aerospaziale Tedesco. Questo strumento è estremamente utile per la ricerca, la pianificazione del traffico e la progettazione di infrastrutture di trasporto poiché consente di modellare dettagliatamente reti stradali e comportamenti veicolari.

SUMO offre una simulazione microscopica, simulando il movimento di veicoli e pedoni individualmente e il suo approccio dettagliato permette di analizzare una vasta gamma di scenari di traffico, dalle strade cittadine congestionate alle autostrade a scorrimento veloce. Inoltre, può simulare l'interazione tra veicoli convenzionali, veicoli autonomi, trasporto pubblico e pedoni, rendendolo uno strumento estremamente versatile per varie applicazioni. Una peculiarità di SUMO è proprio quella di considerare con grande dettaglio le interazioni tra gli utenti e tra quest'ultimi e la rete di trasporto con un'alta velocità di esecuzione: infatti, il simulatore è in grado di rappresentare l'evoluzione del flusso di traffico, in termini di movimento di ogni singolo veicolo, ad intervalli di tempo regolari e in maniera continua nello spazio.

L'utilizzo di SUMO coinvolge diverse fasi fondamentali. In primo luogo, si crea una rete stradale dettagliata, che può essere costruita manualmente, importata da dati GIS, o convertita da altre fonti di dati tramite strumenti come NETCONVERT. Successivamente, si definiscono gli scenari di traffico specificando i flussi di veicoli e pedoni, inclusi rotte, orari di partenza e destinazioni. Con la rete e la domanda di traffico pronte, si esegue poi la simulazione utilizzando SUMO-Simulation, che può essere monitorata e visualizzata in tempo reale tramite SUMO-GUI, un'interfaccia grafica che mostra il movimento dei veicoli e l'interazione con l'infrastruttura stradale, permettendo di osservare il comportamento del traffico in dettaglio. Dopo la simulazione, i dati raccolti possono essere analizzati per valutare vari aspetti del traffico, come i tempi di viaggio, le velocità medie, i livelli di congestione e l'efficacia delle strategie di gestione del traffico.

Oltre al microsimulatore vero e proprio, il pacchetto software di SUMO comprende tutte le applicazioni necessarie per espletare le operazioni propedeutiche alla simulazione stessa, come la ricerca del percorso (DUAROUTER e JTRROUTER), la generazione della domanda basata sulle attività (ACTIVITYGEN), la visualizzazione e la modifica

della rete (NETEDIT) ed il calcolo delle emissioni (EMISSIONSMAP e EMISSIONSDRIVINGCYCLE). Tali componenti del software SUMO possono essere utilizzati attraverso l'interfaccia grafica di SUMOpy.

SUMO trova applicazione in diversi campi. Si tratta anche uno strumento fondamentale per testare e ottimizzare le tecnologie dei Sistemi di Trasporto Intelligente (ITS), come i sistemi di controllo del traffico in tempo reale, e per simulare e testare il comportamento dei veicoli autonomi in diversi scenari di traffico.

## 2.2. HYBRIDPY

HybridPY è un software *open-source* utilizzato per eseguire scenari di simulazione basato su SUMOpy e scritto in Python 3.9, il che rende il suo utilizzo molto più intuitivo e *user-friendly* rispetto a SUMO. La forza di hybridPY è che consente di eseguire scenari SUMO e MATSim indipendentemente e può trasformare i dati tra i due modelli di simulazione. Di conseguenza, è in grado di eseguire e analizzare scenari di simulazione microscopica, mesoscopica ma anche ibrida, in cui gli scenari MATSim e SUMO vengono eseguiti sequenzialmente.

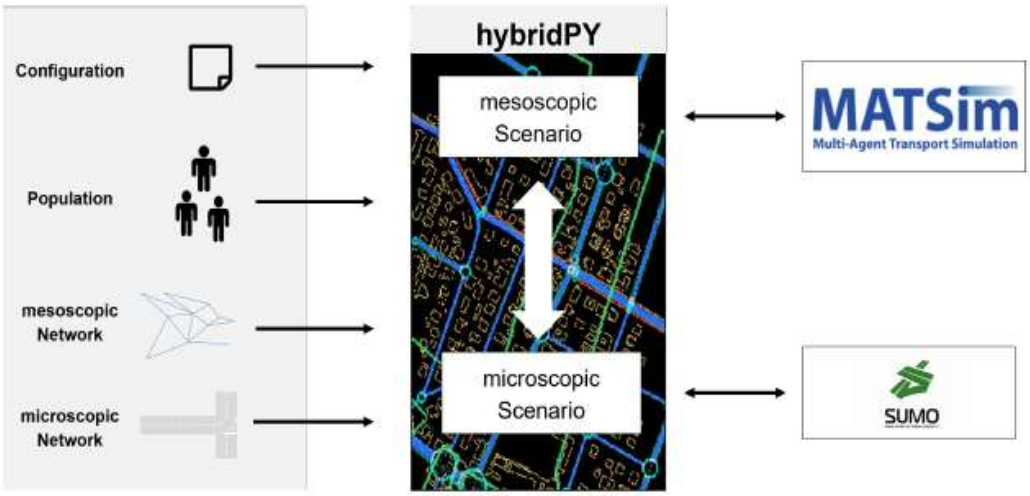


*Figura 2.6: Schema concettuale hybridPY*

Il software è in grado di leggere, scrivere e anche creare file di configurazione, file di popolazione e file di rete sia per la simulazione del traffico mesoscopica che per quella microscopica. Tutti i processi e le funzioni possono essere avviati dalla sua GUI (interfaccia grafica), o alternativamente tramite *scripting* su Python, con il browser dei dati che offre all'utente una visione strutturata dei dati di simulazione necessari. L'*Editor* della rete, invece, è mostrato sul lato destro, consentendo di visualizzare le reti microscopiche e mesoscopiche a diversi livelli, permettendo un esame dettagliato e mirato del modello di rete ibrido.

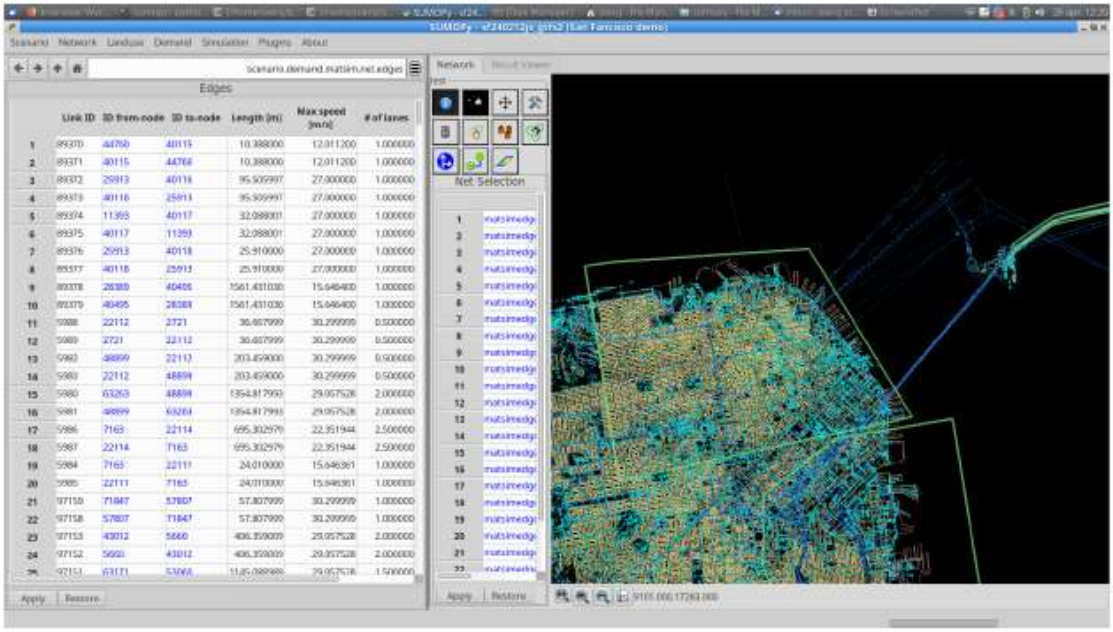

*Figura 2.7: Interfaccia grafica hybridPY*

Gli step principali per effettuare una simulazione di un sistema di trasporto sono:

1. Preparazione della rete → consiste nell'importare i dati della rete stradale da OpenStreetMap o altre sorgenti di dati e nel successivo *editing* per correggere eventuali errori.
2. Importazione dei dati della rete mesoscopica → si importa la rete mesoscopica proiettandola nel sistema di coordinate definito dalla simulazione microscopica.
3. Caricamento delle impostazioni di configurazione necessarie per eseguire la simulazione MATSim nello scenario → include parametri relativi alla durata della simulazione, al comportamento degli agenti, alle funzioni di punteggio, ai modelli

di scelta del mezzo oppure dati della popolazione come demografia degli agenti o dei viaggiatori, programmi delle attività, coppie origine-destinazione e preferenze di mobilità.

4. Esecuzione dello scenario → la simulazione MATSim utilizza la rete importata, la configurazione e i dati della popolazione e simula il comportamento di viaggio degli agenti individuali basato sui parametri definiti, generando file di output contenenti i risultati della simulazione.
5. Importazione nello scenario hybridPY → dopo che la simulazione MATSim è completa, i file di output generati da MATSim vengono importati nello scenario hybridPY. Per ragioni di efficienza, vengono importati solo i dati della popolazione dall'output MATSim all'interno dell'area SUMO.
6. Importazione della domanda → può essere eseguita tramite l'approccio basato sui viaggi o l'approccio basato sulla popolazione.
7. Simulazione su SUMO → viene utilizzato per simulare le dinamiche del traffico e i movimenti dei veicoli basati sui dati importati da MATSim.
8. Analisi dei risultati → i risultati possono essere aggregati (valori medi) o disaggregati (riferiti al singolo utente). Possono essere analizzati consumi di carburante, emissioni di inquinanti, tempi di viaggio (distinti in tempi di percorrenza e tempi d'attesa), o flussi sugli archi. I risultati di una microsimulazione, come i tempi di percorrenza sugli archi, possono essere riutilizzati per migliorare i risultati in uno step di simulazione successivo.

Per poter comprendere il principio base di funzionamento della simulazione, occorre partire dalla definizione della rete, la quale viene trattata con un approccio microscopico.

## 2.3. ELEMENTI DELLA RETE

Nella microsimulazione, vista l'attenzione posta sui singoli utenti e sulle interazioni tra loro, è necessario modellizzare la rete nel dettaglio, andando a classificare ogni singolo elemento in base alla sua funzione. Si vanno quindi a delineare alcuni elementi fondamentali che compongono il sistema dell'offerta.

### 2.3.1. Archi e nodi

Analogamente al modello di una rete di trasporto macroscopico, una rete di trasporto definita all'interno di un software di microsimulazione, come SUMO, è costituita da un grafo bidimensionale composto da archi orientati (chiamati *edges* o *links*), che possono avere una o più corsie, e nodi (*nodes*), che rappresentano le intersezioni o i punti terminali degli archi. Nella microsimulazione, sono fondamentali i numerosi attributi associati a ciascun elemento della rete.

Gli attributi più significativi di ogni elemento sono:

- Le coordinate (x, y) del centro del nodo, che possono essere georeferenziate (Latitudine LAN, Longitudine LON) o proiettate su un sistema di coordinate locali.
- Il tipo di nodo, ad esempio incrocio con diritto di precedenza, incrocio semaforizzato, incrocio con obbligo di stop, ecc.

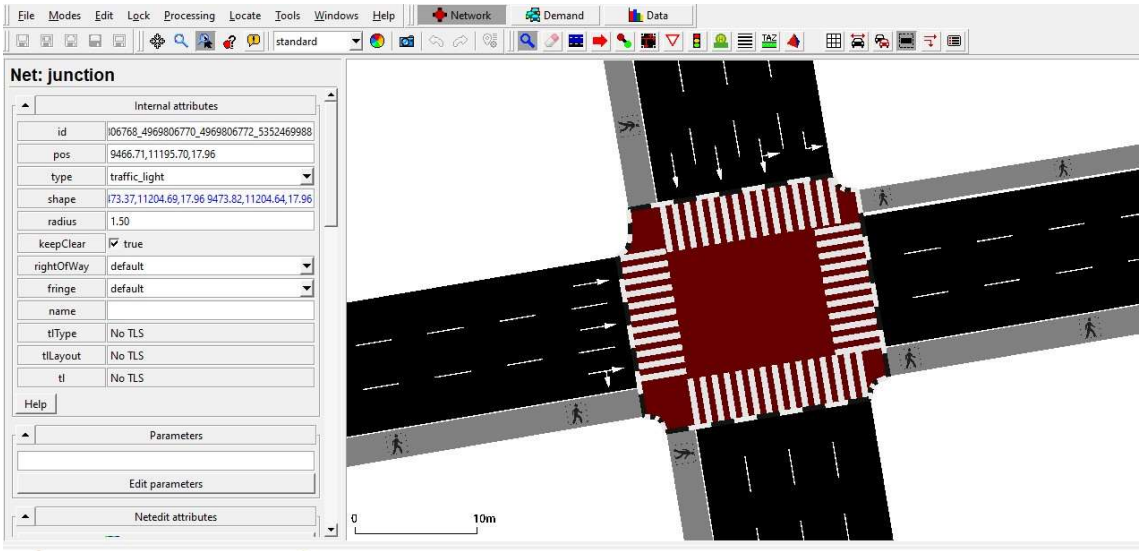


*Figura 2.8: Esempio di nodo con caratteristiche su Netedit*

Un nodo è anche composto da connettori (*connections*). Se il nodo fa parte di un sistema semaforico, contiene anche informazioni sulle fasi semaforiche.

Gli attributi principali di un arco, oltre al nodo di origine e destinazione, sono:

- La lunghezza.

- La priorità della strada, che in SUMO varia da 1 a 14, dove il valore 0 rappresenta la tipologia di arco con la priorità più bassa (es. sentieri in un parco) e la priorità 14 significa la priorità più alta (es. autostrada).

Un arco contiene inoltre una o più corsie (*lanes*), ognuna delle quali ha i suoi attributi. Le caratteristiche di un arco sono quindi determinate essenzialmente dalle caratteristiche delle corsie che lo compongono.

### 2.3.2. Corsie e connettori

Gli archi, come già anticipato, sono caratterizzati dalle corsie che li compongono, le quali sono numerate con un indice che parte da zero per la corsia più a destra. Ogni corsia ha i seguenti attributi:

- Larghezza.
- Velocità massima consentita.
- Eventuali limiti di accesso per una certa categoria di veicoli (es. riservato a biciclette o autobus).

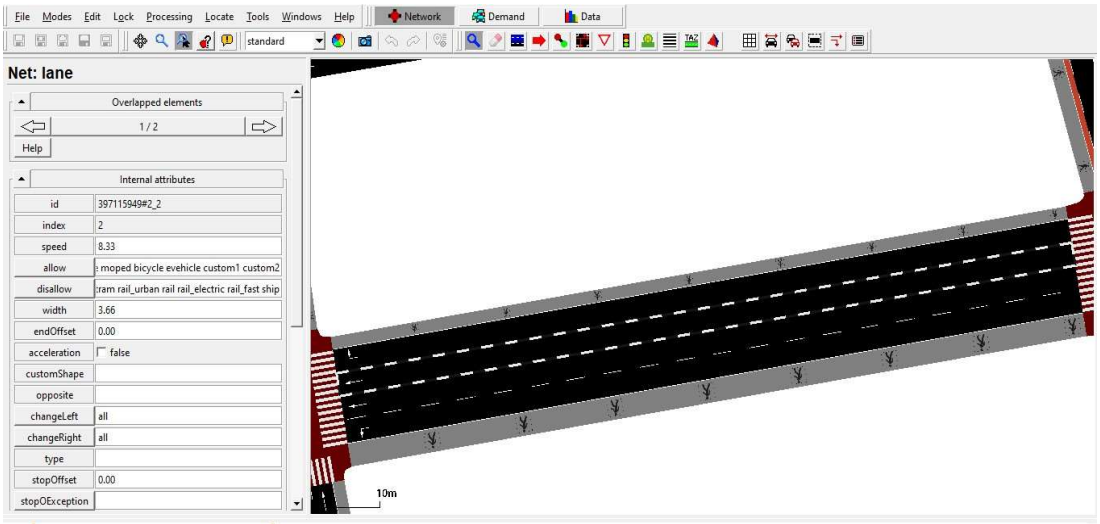


*Figura 2.9: Esempio di corsia con caratteristiche su Netedit*

Dato che ogni corsia può avere i propri diritti di accesso e limiti di velocità, è possibile creare strade con marciapiedi, corsie ciclabili o definire diversi limiti di velocità per ciascuna corsia di un arco.

I punti terminali degli archi sono invece i nodi (*junctions*), rappresentati in maniera dettagliata mettendo in evidenza ogni possibile manovra. Una svolta ammessa è rappresentata da un connettore che collega le rispettive corsie in entrata e in uscita. Le corsie per pedoni non necessitano di connettori poiché i pedoni possono muoversi anche contro mano.

Un connettore collega sempre e solo due corsie, quindi è descritto da una corsia in ingresso e una in uscita. Gli attributi più importanti di un connettore corrispondono a quelli già citati in precedenza per le corsie, ovvero larghezza, velocità massima consentita ed eventuali limiti di accesso per una certa categoria di veicoli. Quando viene creata una nuova giunzione, o se ne modificano i bordi, viene creata automaticamente una serie di nuove connessioni. Se viene selezionata una corsia di origine in modalità connessione, tutte le corsie in uscita all'incrocio vengono colorate in base alle categorie seguenti:

- Azzurro → corsia di origine selezionata.
- Verde chiaro → la corsia è già connessa alla corsia di origine.
- Verde scuro → la corsia è un possibile obiettivo ma non è ancora connessa.
- Fucsia → la corsia è già connessa e questa connessione è impostata forzatamente come prioritaria.
- Giallo → la corsia non è collegata e una connessione sarebbe insolita per motivi di conflitto.

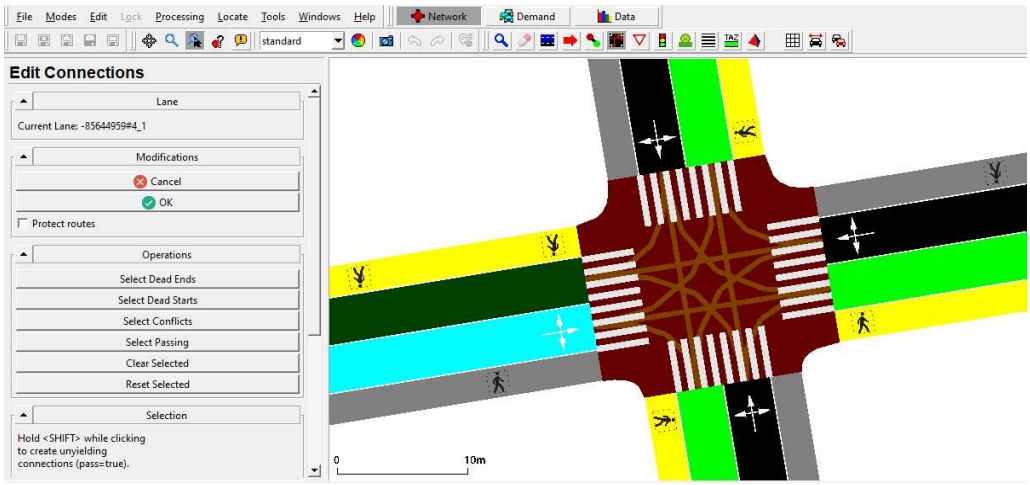


*Figura 2.10: Esempio di connettore su Netedit*

Nel caso di nodo privo di intersezione semaforica, si può incontrare il caso in cui due connettori convergano nella stessa corsia di destinazione. In questo caso, è necessario fornire delle regole in merito alla precedenza:

- Blu → connettore selezionato.
- Rosso → ha il diritto di passare.
- Verde → dà la precedenza.
- Grigio → nessun conflitto.
- Arancione → conflitto non regolato.
- Azzurro → mutuo conflitto.

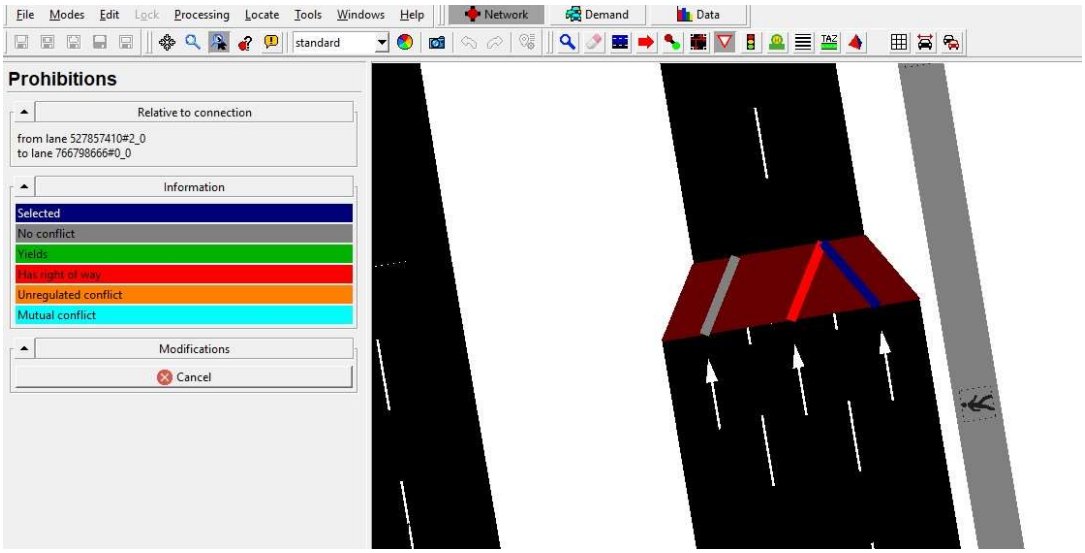


*Figura 2.11: Esempio di precedenza su Netedit*

### 2.3.3. Semafori

Un sistema semaforico può riferirsi a un singolo nodo o essere sincronizzato per un gruppo di nodi. Il programma semaforico è definito attraverso i connettori dei nodi che fanno parte del sistema semaforico, indicando i momenti in cui è possibile utilizzarli. Un programma semaforico è rappresentato da una sequenza di fasi in cui ogni fase ha una durata e definisce il colore di ogni connettore che fa parte del sistema semaforico. Ogni colore ha un significato specifico:

- Verde → attraversamento consentito.

- Verde scuro → attraversamento consentito ma senza diritto di precedenza sul verde.
- Giallo → il veicolo cerca di fermarsi prima del semaforo, ma attraversa se fermarsi non è più possibile.
- Giallo scuro → giallo senza diritto di precedenza sul giallo.
- Marrone (off-blinking) → rappresenta il caso stradale di giallo lampeggiante.
- Rosso → attraversamento impedito.
- Ciano (off) → il connettore non ha nessun semaforo associato e quindi vige il diritto di precedenza.

Dopo aver associato un colore ad ogni connettore del sistema semaforico per ciascuna fase di un ciclo semaforico, il programma può essere rappresentato da una tabella.

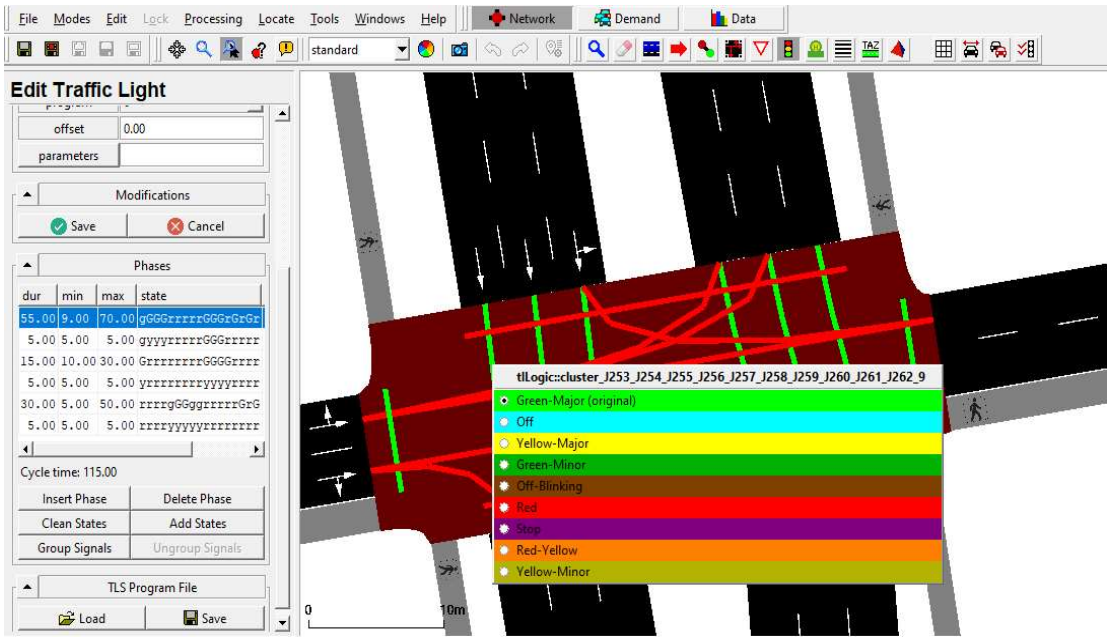


*Figura 2.12: Esempio connettori semaforici su Netedit*

# 3. CASO STUDIO: MODELLO DI SAN FRANCISCO

San Francisco, denominata ufficialmente City and County of San Francisco, è un centro commerciale, finanziario e culturale nella California settentrionale. Con una popolazione di 808.437 residenti nel 2022, è la quarta città più popolosa dello stato della California dopo Los Angeles, San Diego e San Jose. La città copre una superficie di 46,9 miglia quadrate (corrispondenti a 121 chilometri quadrati) all'estremità della penisola di San Francisco, rendendola la seconda città di grandi dimensioni più densamente popolata degli Stati Uniti dopo New York City, e la quinta contea più densamente popolata degli Stati Uniti, dietro solo quattro distretti di New York City. Tra le 92 città statunitensi con oltre 250.000 residenti, San Francisco è al primo posto per reddito pro capite e al sesto posto per reddito complessivo nel 2022.

*Figura 3.1: San Francisco skyline*

## 3.1. SISTEMI DI TRASPORTO

Il traffico urbano a San Francisco è noto per essere impegnativo e spesso congestionato, specialmente durante le ore di punta del mattino e del pomeriggio. La città è caratterizzata da strade strette, colline ripide e un'infrastruttura stradale che risale ad un periodo in cui il numero di veicoli in circolazione era molto inferiore rispetto ad oggi, il che rende la

circolazione in certe zone della città più difficile, con una rete stradale che spesso appare labirintica ai visitatori.

Le strade principali come Market Street, Lombard Street e Van Ness Avenue, così come i ponti che collegano la città con le aree circostanti, possono diventare particolarmente congestionate durante le ore di punta. Anche le strade che attraversano i quartieri residenziali possono essere strette e trafficate, con spazi di parcheggio limitati che contribuiscono ad aumentare la congestione.

Per di più, il traffico a San Francisco è influenzato dalla presenza di eventi speciali, manifestazioni, lavori stradali e condizioni meteorologiche, i quali possono aggiungere ulteriori sfide alla circolazione stradale. In generale, la città è impegnata a migliorare la mobilità urbana attraverso investimenti in trasporto pubblico, infrastrutture ciclabili e piani di gestione del traffico.

### 3.1.1. Trasporto pubblico

Il trasporto pubblico è il mezzo più utilizzato quotidianamente a San Francisco. Durante i giorni lavorativi, oltre 560.000 persone utilizzano le 69 linee di autobus del Muni, mentre più di 140.000 viaggiano con il sistema di metropolitana leggera Muni Metro per un totale del 32% della popolazione residente, rendendo la città la quarta negli Stati Uniti per utilizzo di trasporto pubblico e la prima sulla Costa Occidentale.

La San Francisco Municipal Railway, nota principalmente come Muni, è il principale sistema di trasporto pubblico della città e, dal 2023, è l'ottavo più grande negli Stati Uniti. Questo sistema include la metropolitana leggera e sotterranea Muni Metro, una vasta rete di autobus e filobus, una storica linea di tram che attraversa Market Street da Castro Street a Fisherman's Wharf e i celebri cable car, considerati National Historic Landmark nonché una popolare attrazione turistica.


*Figura 3.2: Linee Muni*

Affianco al Muni, si può citare anche il Bart (Bay Area Rapid Transit), ovvero un sistema ferroviario ad alta velocità che serve l'intera area della Bay Area, collegando San Francisco a città come Oakland, Berkeley e l'aeroporto internazionale di San Francisco, la quale rappresenta un'opzione popolare per i pendolari che lavorano in diverse città della regione.

Per accogliere i numerosi cittadini che si recano quotidianamente a Silicon Valley, inoltre, aziende come Genentech, Google e Apple offrono trasporti privati in autobus per i loro dipendenti, partendo da San Francisco.

### 3.1.2. Strade e autostrade

San Francisco ha 1.088 miglia (1.741 km) di strade, di cui 946 miglia sono di superficie e 59 miglia sono autostrade. La sua geografia unica e le rivolte contro le autostrade alla fine degli anni '50 hanno influenzato la configurazione delle vie di trasporto. Tra le

principali vie di comunicazione all'interno della città si annoverano l'Interstate 80 che inizia all'approccio dell'Oakland Bay Bridge ed è l'unico collegamento automobilistico diretto con l'East Bay; la U.S. Route 101 si collega all'Interstate 80 e offre accesso al Sud della città e alla Silicon Valley mentre a Nord attraversa il Golden Gate Bridge verso la contea di Marin e il North Bay; la State Route 1 entra a San Francisco da Nord attraverso il Golden Gate Bridge e prosegue come arteria principale della 19th Avenue, unendosi all'Interstate 280 al confine meridionale della città; l'Interstate 280 continua a Sud da San Francisco e verso Est lungo il bordo meridionale della città, terminando poco a sud dell'Oakland Bay Bridge nel quartiere South of Market; la State Route 35 entra in città da Sud come Skyline Boulevard e termina alla sua intersezione con la Highway 1; la State Route 82 entra a San Francisco da Sud come Mission Street e termina poco dopo la sua giunzione con la 280, il capolinea occidentale della storica Lincoln Highway, la prima strada transcontinentale attraverso l'America. Dopo il terremoto di Loma Prieta del 1989, sono state demolite l'Embarcadero Freeway e una parte della Central Freeway, convertendole in boulevard a livello stradale.

### 3.1.3. Ciclisti e pedoni

La bicicletta è un mezzo di trasporto molto popolare a San Francisco, con 75.000 residenti che si spostano con essa ogni giorno. Negli ultimi anni, la città ha migliorato l'infrastruttura ciclabile con piste ciclabili protette che ad ora contano 28 miglia (45 chilometri) di lunghezza per una percentuale che si attesta al 2,6% delle strade cittadine. Il progetto Bay Wheels, inizialmente chiamato Bay Area Bike Share, è stato lanciato nell'agosto 2013 con 700 biciclette nel centro di San Francisco e stazioni di ancoraggio sparse per tutta la città, rendendole notevolmente accessibili. Per quanto riguardano gli spostamenti giornalieri, la Municipal Transportation Agency (MTA) nel 2008 stimava che circa 128.000 viaggi giornalieri in città fossero effettuati in bicicletta, pari al 6% dei viaggi totali.

Anche il traffico pedonale è molto diffuso. Nel 2015, Walk Score ha classificato San Francisco come la seconda città più percorribile a piedi degli Stati Uniti.

## 3.2. IMPORTAZIONE DELLA RETE

Il primo elemento che bisogna considerare al fine della microsimulazione è il modello dell'offerta di trasporto, che in questo caso è rappresentato dall'area urbana della città di San Francisco, scaricabile dal database di OpenStreetMap. Esso crea e fornisce gratuitamente dati geografici come mappe stradali ed è stato introdotto perché la maggior parte delle mappe considerate gratuite in realtà hanno restrizioni legali o tecniche sul loro utilizzo, impedendo alle persone di utilizzarle in modo produttivo.

Un file OpenStreetMap contiene una mappa con le seguenti funzionalità relative alla simulazione del traffico:

- I nodi e le loro connessioni definiscono la posizione e la forma di tutte le strade e gli incroci.
- Il tipo di strada ne specifica le dimensioni e l'importanza.
- Il limite di velocità di una strada è solitamente determinato implicitamente dalla legge. Nel caso in cui il limite di velocità differisca dal valore legale implicito, viene fornito esplicitamente tramite la massima velocità di una strada.
- Il numero totale di corsie (per entrambe le direzioni) ha un valore implicito che dipende dalla proprietà della strada. Se il valore reale differisce dal valore predefinito, può essere fornito tramite le corsie chiave.
- La posizione di ogni semaforo è descritta come un nodo con la coppia chiave-valore *segnale di traffico*.
- Le strade a senso unico sono contrassegnate con la coppia chiave-valore *una direzione*.

Rispetto al formato file di rete SUMO, il formato file OpenStreetMap è carente della logica dei semafori e del significato delle corsie e collegamenti tra esse in corrispondenza di un incrocio.

Dopo aver individuato l'area d'interesse nel browser, è possibile delimitarla con un'area rettangolare e, esportandola, viene generato un file con estensione *.osm* o *.osm.xml*.


*Figura 3.3: Mappa OSM di San Francisco*

Si possono anche utilizzare strumenti come Overpass API o JOSM (Java OpenStreetMap Editor). Il file che viene generato, però, non è utilizzabile direttamente su hybridPY, infatti esiste il comando NETCONVERT per convertire i dati OSM in una rete SUMO.

## 3.3. *EDITING* DELLA RETE

Grazie al file OSM, quindi, è possibile caricare la rete nel software per avere lo scenario di partenza. La rete a disposizione contiene numerose informazioni ed è molto dettagliata, però siccome si tratta di un'area molto vasta (circa quattro volte più grande di Bologna), una sua eccessiva pesantezza può rendere le successive operazioni molto lente. Proprio per questo motivo, prima di importare la domanda di trasporto e svolgere le simulazioni, è necessario alleggerire la rete con una serie di interventi di *editing*.

Lo strumento utilizzato per questo scopo è Netedit, un *editor* di rete grafico incluso in SUMO che può essere utilizzato per creare reti da zero o per visualizzare e modificare tutti gli aspetti delle reti esistenti. Essendo dotato di una potente interfaccia di selezione ed evidenziazione, può essere utilizzato anche per eseguire il debug degli attributi di rete.

Netedit è costruito sopra Netconvert, quindi, come regola generale, tutto ciò che Netconvert può fare, anche Netedit può farlo. Inoltre, Netedit ha funzionalità di annullamento/ripetizione illimitate e quindi consente di correggere rapidamente gli errori di modifica. L'apertura di Netedit può essere lanciata da Python o direttamente da hybridPY e, dopo aver terminato di modificare la rete, il file viene salvato in formato .xml.

Per svolgere l'*editing* in modo preciso, puntuale e coerente con la rete vera e propria, è stata necessaria una continua consultazione di Google Street View. In questo modo, ogni qualvolta erano presenti sulla rete virtuale uno o più elementi di difficile comprensione, il confronto immediato con l'immagine satellitare o stradale rilevata da Google ha permesso di modificare la rete in modo da renderla più vicina possibile alla situazione reale.

Gli obiettivi principali dell'attività di *editing* sono stati:

- Minimizzare il numero di archi e nodi in modo da sfoltire e semplificare la rete per renderla il più leggera possibile. Dovendo anche aggiungere la domanda di trasporto in seguito, infatti, il file completo si aggira su 1 GB che lo rende molto dispendioso a livello computazionale.
- Aggiungere elementi presenti nella realtà che non sono stati trasferiti sul modello virtuale come semafori, linee di trasporto ecc.
- Semplificare i punti in cui si sovrappongono numerosi nodi e archi per poi correggere eventualmente l'elevazione.

L'*editing* della rete non è stato svolto unicamente prima dell'importazione della domanda di trasporto, ma è stato un processo portato avanti di pari passo con le simulazioni svolte. Grazie ad esse, infatti, è stato possibile identificare punti che erano sfuggiti alla fase preliminare, ma soprattutto elementi legati direttamente al traffico urbano:

- Cicli semaforici.

- Connettori.
- Attraversamenti pedonali.
- Velocità ammesse per ogni arco.

### 3.3.1. Percorsi pedonali

Il marciapiede è un arco particolare nel quale possono transitare solo pedoni in entrambi i sensi di marcia. Nel trattare i marciapiedi, spesso si è presentato il caso in cui erano presenti due archi pedonali per corsia: come spesso accade nelle grandi città, le strade sono infatti costeggiate dai portici, i quali spesso sono riconosciuti come un ulteriore percorso pedonale. Oltre ai portici, esistono altri vari casi per il quale il programma riconosce più di un percorso pedonale nella stessa direzione per corsia. Ai fini dell'*editing*, non è necessario mantenere entrambi gli archi pedonali ma ne basta soltanto uno.

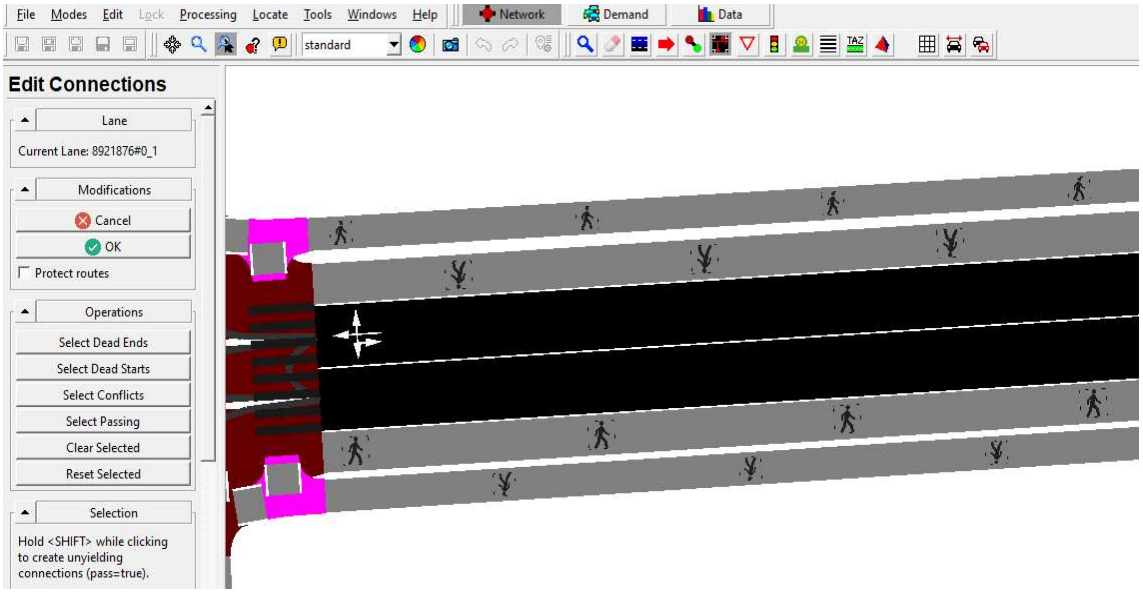


*Figura 3.4: Doppio arco pedonale*

In altri casi, può essere invece utile aggiungere un marciapiede ove non presente: se si vuole aggiungere un arco pedonale, il programma permette di farlo con un apposito comando, ma solamente nella direzione del senso di marcia. Nel caso fosse necessario inserire un marciapiede nella direzione opposta o, comunque, in entrambe, il metodo

migliore è duplicare la corsia senza marciapiede e modificare le proprietà della nuova corsia per farla diventare pedonale. Questo ultimo metodo è stato utilizzato per aggiungere lo spazio necessario per inserire le fermate del trasporto pubblico, le quali possono essere collocate solamente sui marciapiedi. Siccome si trovano al centro della carreggiata prima di un incrocio, è stato necessario dividere l'arco in due parti per crearne uno piccolo in prossimità dell'intersezione, e poi aggiungere un arco stradale a sinistra per poi modificarlo in pedonale.

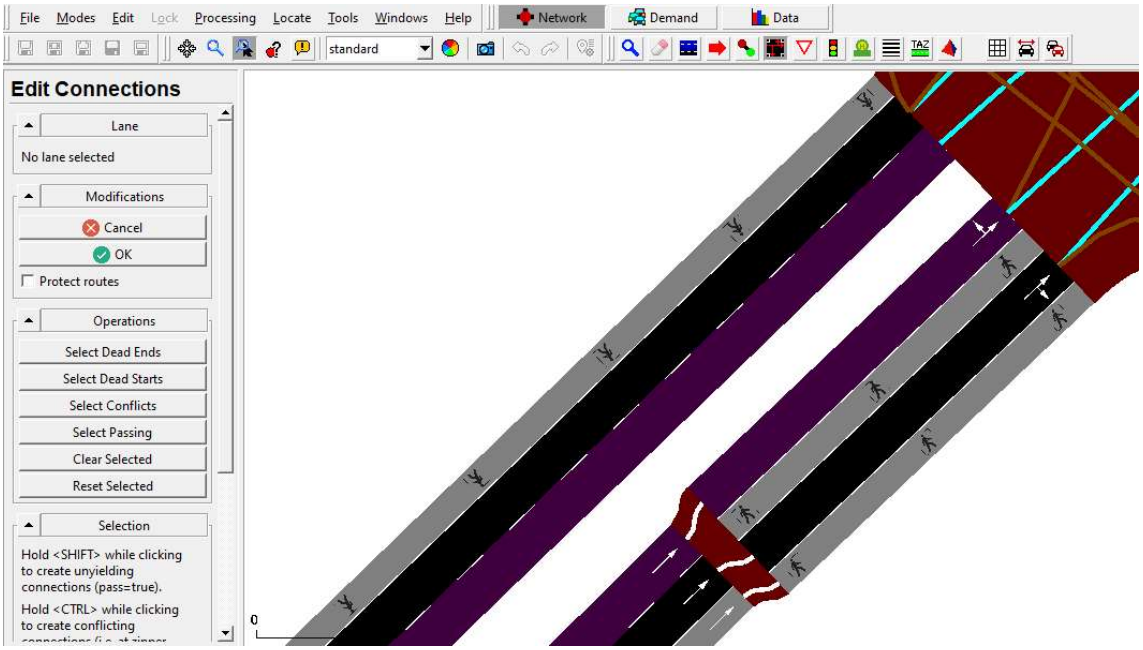


*Figura 3.5: Arco pedonale creato per la fermata*

### 3.3.2. Linee tramviarie

Il trasporto pubblico in superficie prevede, oltre a numerosi autobus e filobus che servono tutta la città, anche una storica linea tramviaria che percorre le principali vie del centro città. Mentre in The Embarcadero Street il tram viaggia su sede riservata al centro della carreggiata, nella quasi totalità delle altre vie la rotaia è annegata nella pavimentazione stradale e vi possono transitare anche gli altri veicoli. Perciò, è stato necessario cambiare quasi interamente gli attributi di tali archi in modo da renderli transitabili anche dal trasporto privato. Tale modifica si può notare perché la rappresentazione dell'arco è passata da essere a trattini a viola.

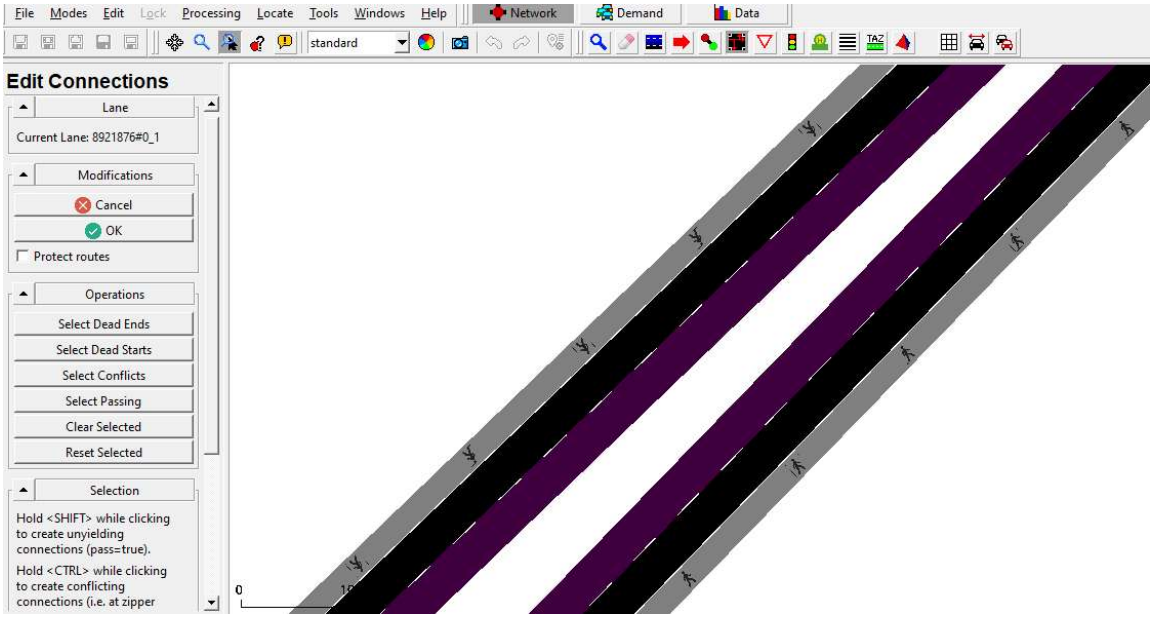


*Figura 3.6: Linea tramviaria su sede promiscua*

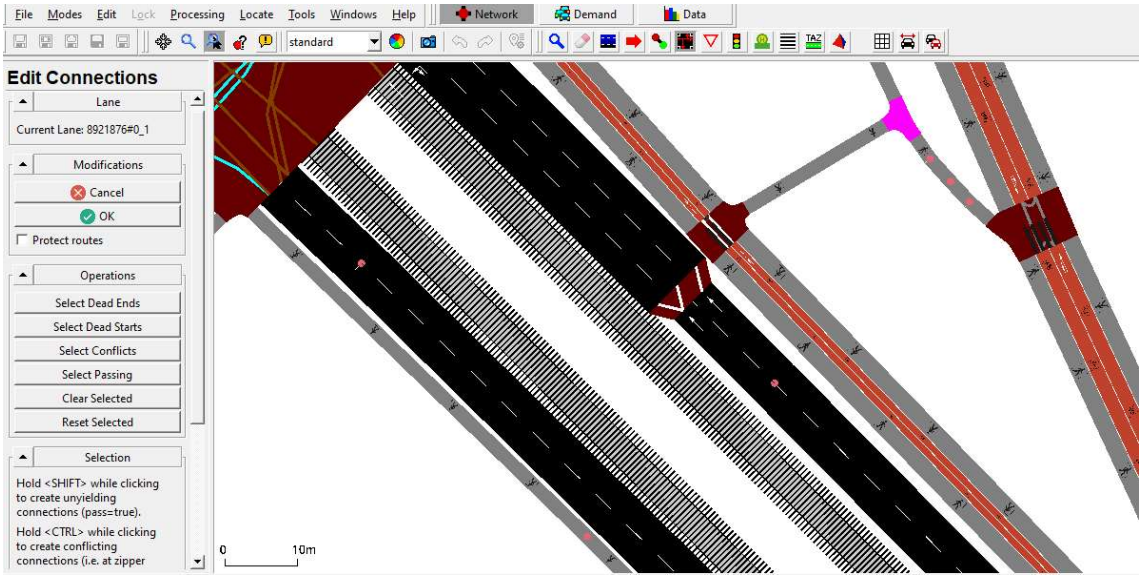


*Figura 3.7: Linea tramviaria su sede riservata*

### 3.3.3. Nodi e connessioni

Un incrocio stradale consiste in uno o più nodi ravvicinati e collegati fra di loro. Nella rappresentazione, l'obiettivo principale è semplificare il più possibile l'incrocio, ovvero minimizzare il numero dei nodi conservando però le sue funzionalità (svolte e collegamenti interni). Molti incroci, in particolare quelli più semplici, sono formati da un singolo nodo. In questo caso i passaggi da fare sono i seguenti:

1. Configurare la geometria migliore per il nodo, l'operazione è facilitata se si inseriscono i *geometry point* sul finire dell'arco.
2. Verificare o attribuire la corretta tipologia di precedenza stradale al nodo.
3. Controllare ed eventualmente modificare i connettori delle corsie.
4. Inserire gli attraversamenti pedonali.

Esistono però situazioni dell'infrastruttura viaria che a causa della loro complessità non possono essere rappresentate con un singolo nodo. Questo tipo di incrocio si presenta confusionario per via di molti nodi colleganti strade ordinarie e viabilità ciclabile concentrati in poco spazio. Per editare questo tipo di situazioni occorre seguire in ordine i seguenti passaggi:

1. Pulire inizialmente in maniera generale perchè spesso la confusione non permette neppure di comprendere come si articola l'incrocio. In genere è difficile poter muovere i nodi per esplorare gli archi coinvolti nell'intersezione poiché essi sono legati ad una posizione fissa. Occorre sbloccare queste posizioni fisse ed eliminare gli elementi importati male.
2. Ricostruire gli archi costituenti l'incrocio, schematizzando in modo preciso tutti gli archi corti di collegamento tra i vari archi principali. In generale, la ricostruzione dell'incrocio è una operazione da effettuare tendendo a ridurre al minimo la quantità di nodi e archi all'interno dell'incrocio ma senza perdere le caratteristiche funzionali dello stresso.
3. Controllare e modificare i connettori tra le corsie dei vari archi. Talvolta, negli incroci complessi è necessario un notevole cambiamento rispetto ai collegamenti di default che crea il programma. La scelta dei connettori corretti è fatta in funzione dell'usuale comportamento dei veicoli, della segnaletica orizzontale e verticale posta sull'incrocio, di eventuali manovre non permesse ma molto ricorrenti.
4. Inserire gli attraversamenti pedonali ove non presenti.
5. Implementare il sistema semaforico negli incroci che lo prevedono.

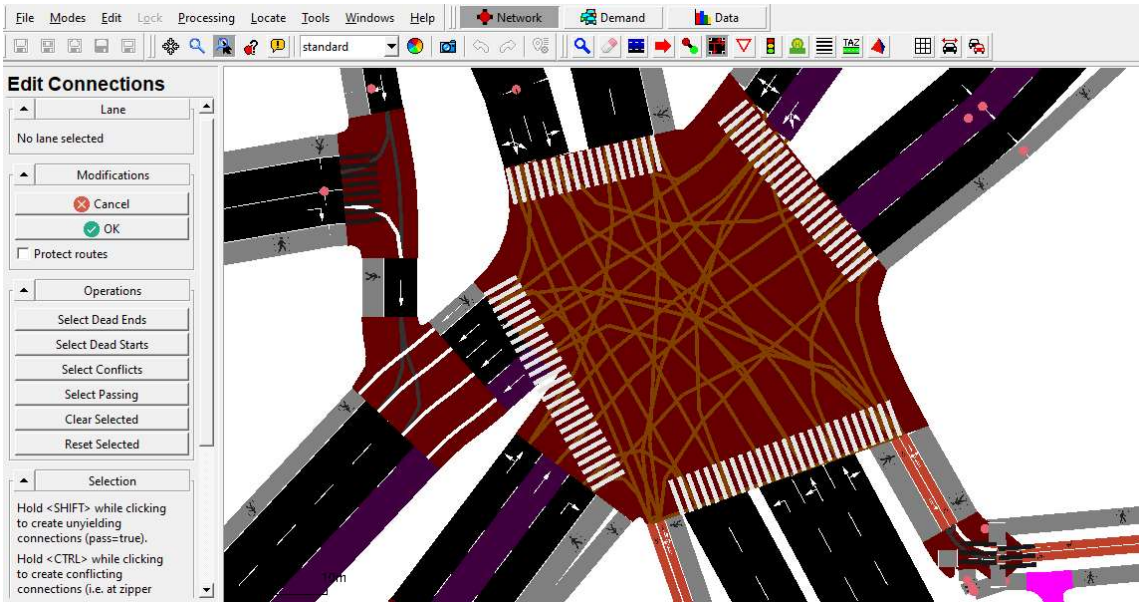


*Figura 3.8: Nodo complesso ristrutturato in base agli interventi illustrati*

Siccome la rete di San Francisco è già notevolmente grande, se gli incroci fossero composti da tanti piccoli archi e nodi, la simulazione diventerebbe estremamente lenta e le interazioni tra i veicoli si complicherebbero molto. In questo senso, si è deciso di semplificare anche gli incroci più grandi e complessi (come quelli su Market Street) utilizzando un unico nodo nel quale convergono tutti gli archi in ingresso. In tal modo, il ciclo semaforico è più semplice da determinare essendo unico.

### 3.3.4. Semafori

Qualora il nodo che si sta trattando sia un incrocio semaforizzato, occorre inserire il semaforo o modificarne il ciclo qualora fosse già stato creato di default. Con il comando che crea il semaforo, viene generato il sistema semaforico sul nodo con un ipotetico ciclo programmato in automatico: in termini di tempo in base a quanto impostato nella fase di importazione della rete (in questo caso 90 secondi) e in termini di fasi in base agli archi d'ingresso presenti.

Il ciclo semaforico è schematizzato all'interno della tabella *Phases*, in cui ogni riga corrisponde ad una fase del ciclo con il rispettivo tempo e le lettere indicano lo stato di colore di tutti i connettori durante quella fase. Lo scorrimento del tempo e quindi la

successione delle fasi avviene dall'alto verso il basso e selezionando una riga di fase dalla tabella se ne visualizza lo stato di colore dei connettori sulla rete. Dopo aver creato il semaforo occorre attribuirgli il ciclo e le fasi corrette. Per cambiare lo stato di colore di un connettore occorre cliccare con il tasto destro sul connettore, quindi scegliere il colore dalla finestra che si apre. La gamma di scelta di stato di colore è maggiore rispetto ai tre classici verde, giallo e rosso; questo perché il comando è stato sviluppato per ottenere una simulazione più reale ed efficace del traffico in tali incroci. In particolare, la distinzione tra major e minor permette di differenziare il caso di precedenza assoluta nella stessa fase del ciclo (da attribuire ad esempio a connettori con la destra libera o agli attraversamenti pedonali) dal caso di permesso di attraversamento dovendo eventualmente dare la precedenza.

Una ulteriore specificazione del ciclo semaforico riguarda il sistema di controllo: può essere a ciclo fisso o a ciclo attuato. Il ciclo fisso prevede delle fasi di durata stabilita e queste non possono cambiare; il ciclo attuato, invece, prevede una durata indicativa, la quale però può variare da una durata minima ad una durata massima in base a dati di traffico attuali che vengono comunicati da sensori che monitorano continuamente il flusso negli incroci. Nel caso del software e della simulazione, se la fase di verde sta per terminare ma continuano a sopraggiungere veicoli, essa viene prolungata di un tempo $\Delta t$ fino a che non viene rilevata una diminuzione di veicoli in arrivo (ovviamente il limite superiore è dato dalla durata massima indicata in tabella). Viceversa, se in prossimità dell'intersezione arrivano pochi veicoli durante la fase di verde, essa potrebbe anche essere accorciata per favorire un'altra corrente in entrata (il limite inferiore in questo caso è la durata minima indicata). L'adozione di un ciclo semaforico attuato comporta diversi benefici:

- Riduce i tempi di attesa per i veicoli poiché le luci del semaforo si adattano alle condizioni attuali del traffico. Ciò migliora il flusso del traffico, riducendo ingorghi e tempi di viaggio.
- La riduzione dei tempi di attesa contribuisce ad una maggiore efficienza energetica, riducendo il consumo di carburante e le emissioni di gas serra.
- Si registra anche un aumento della sicurezza, poiché riducendo i tempi di attesa e migliorando il flusso del traffico, diminuiscono le possibilità di incidenti stradali.

In certi casi, dove sono presenti numerosi semafori in successione, è stato necessario strutturarli in maniera simile in merito alle fasi semaforiche, in modo tale che si creasse una sorta di onda verde per sfoltire al massimo il traffico (è il caso, per esempio, di Van Ness Ave ovvero la Route 101).

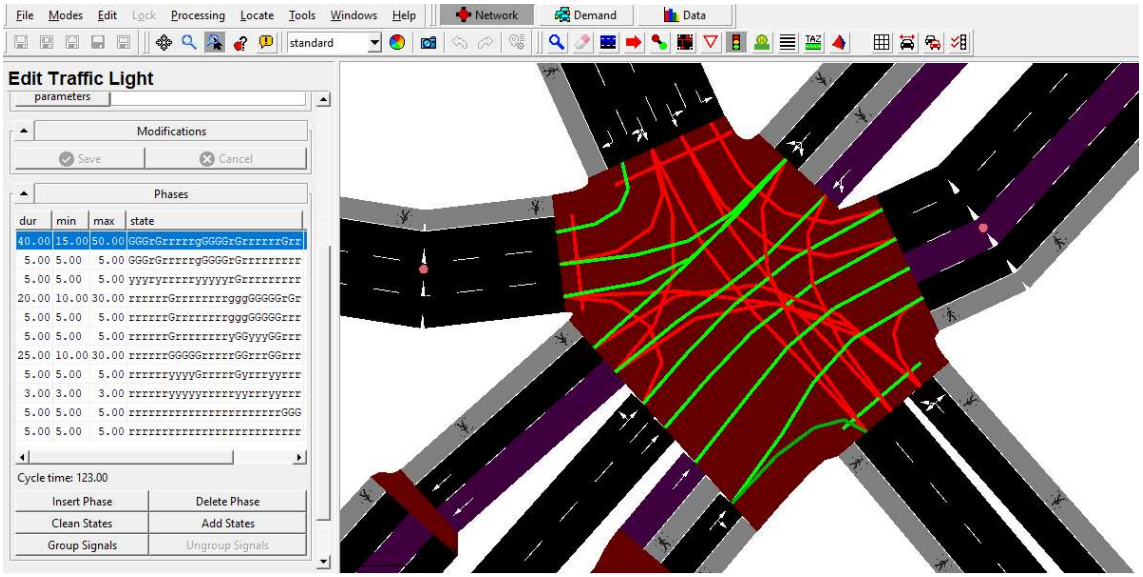


*Figura 3.9: Ciclo semaforico sistemato*

### 3.3.5. Punti di elevazione

I punti di elevazione (*elevation points*) sono punti geografici su una mappa che hanno un'associazione con un'altitudine o un'altezza rispetto al livello del mare. Ciò significa che ogni punto sulla Terra ha un valore di elevazione, che può essere positivo (se il punto è sopra il livello del mare) o negativo (se il punto è sotto il livello del mare).

Importando la rete da OpenStreetMap, vengono anche inclusi i dati riguardo all'altitudine di ogni *junction* e *geometry point*. Siccome San Francisco è caratterizzata da un paesaggio ricco di strade ripide e saliscendi a causa dell'orografia irregolare e ci sono numerosi punti in cui le strade si sovrappongono, questi dati sono spesso inesatti. In tal senso, è stato necessario modificare manualmente i dati di elevazione di gran parte dei punti della rete consultando di pari passo Google Street View e dando una stima approssimativa.

Un altro modo per ottenere i dati di elevazione è il Google Maps Elevation API, che consente di ottenere dati di altitudine per punti specifici sulla Terra utilizzando le coordinate geografiche.

Tutto ciò è stato fatto al fine della visualizzazione sul software Unreal Engine, e perciò la cosa più importante è stata quella di verificare che nel caso di sovrapposizioni di strade ci fosse abbastanza luce tra un piano e l'altro (circa 5/6 m), soprattutto nel caso dell'Oakland Bay Bridge, il quale prevede un livello per ogni senso di marcia.

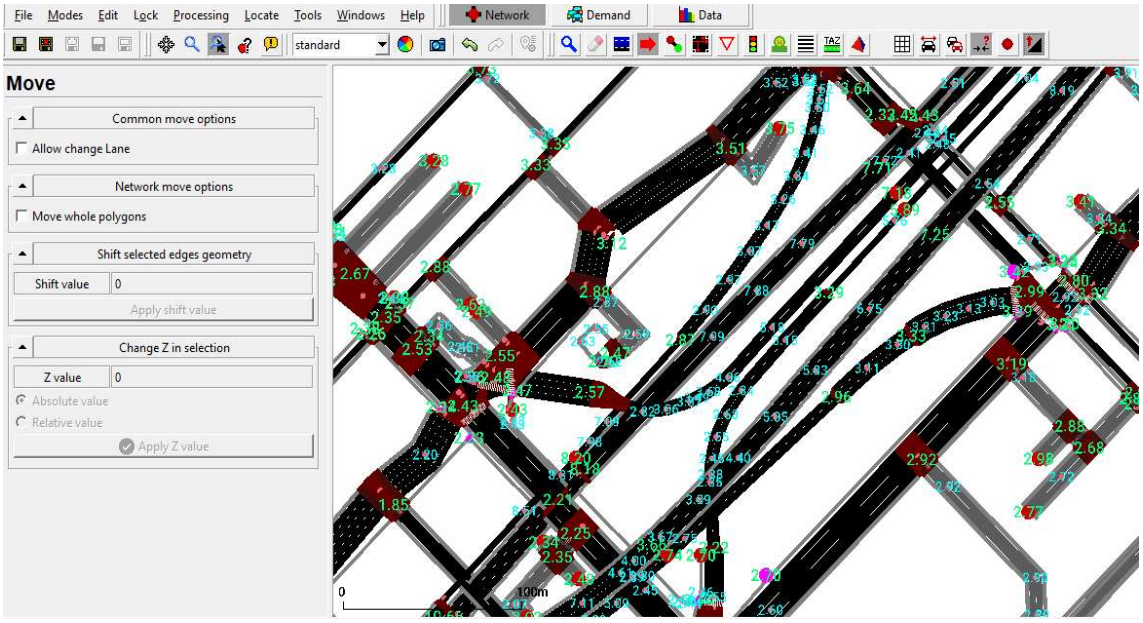


*Figura 3.10: Punti di elevazione*

## 3.4. DOMANDA DI TRASPORTO

Dopo aver modificato preliminarmente la rete con gli interventi illustrati nel paragrafo precedente, è necessario inserire la domanda di trasporto. Come spiegato in precedenza, l'inserimento della domanda di trasporto fa proprio parte di uno dei passi chiave dell'approccio ibrido mesoscopico-microscopico. Infatti, essendo presente già una simulazione mesoscopica MATSim della San Francisco Bay Area (si tratta di un'area molto più vasta rispetto alla zona urbana di San Francisco), la domanda è stata presa da essa e adattata allo scenario in oggetto.

La domanda di trasporto in MATSim è rappresentata in un file di testo *plan.xml*, il quale contiene i piani di viaggio dettagliati di ogni singola persona nella popolazione. La domanda di BEAM CORE, invece, preferisce un file di tipo *csv* ma può anche essere convertita nel formato compatibile con MATSim.

```
<person id="4">
  <plan score="242.173886086" selected="yes">
    <activity type="Home" x="567919.6149779453" y="4189907.8363222484"
              end_time="15:17:06"> </activity>
    <leg mode="car" dep_time="15:17:06" trav_time="00:06:44">
      <route type="links" start_link="62009" end_link="45050"
             distance="9327" trav_time="00:06:44">31025 108348 ... 73295</route>
    </leg>
    <activity type="othdiscr" x="566908.2465720219" y="4183217.0123480107"
              start_time="15:23:50" end_time="16:10:04"> </activity>
    <leg mode="car" dep_time="16:10:04" trav_time="00:15:04">
      <route type="links" start_link="45050" end_link="120216"
             distance="5023" trav_time="00:15:04">45051 73294 ... 89906</route>
    </leg>
    <activity type="social" x="565895.0902456312" y="4184989.135202894"
              start_time="16:25:08" end_time="16:30:57"> </activity>
    <leg mode="car" dep_time="16:30:57" trav_time="00:08:59">
      <route type="links" start_link="112191" end_link="62008"
             distance="7082" trav_time="00:08:59">112043 78048 ... 31024</route>
    </leg>
    <activity type="Home" x="568031.3048479672" y="4189448.878315132"
              start_time="16:39:56"> </activity>
  </plan>
</person>
```

*Figura 3.11: File di domanda MATSim*

Un piano di una persona è composto da due elementi: *activities* e *legs* di un viaggio multimodale. Le prime contengono informazioni sulla posizione, il tipo e la tempistica, mentre le altre contengono informazioni su percorso (come una sequenza di ID di collegamento), modo di trasporto e tempistica (in MATSim sono quasi identiche alle *fasi* di un piano in SUMO).

### 3.4.1. Importazione della rete MATSim/BEAM CORE

A questo punto, le reti dei due simulatori mesoscopici MATSim e BEAM CORE possono essere importate. Entrambi i simulatori utilizzano attributi di rete quasi identici, con l'unica differenza che riguarda il tipo di file preferiti. L'importatore si occupa anche della proiezione delle coordinate tra MATSim/BEAM CORE e SUMO. Oltre agli *ID* dei nodi, agli *ID* dei collegamenti e alla posizione, vengono letti i seguenti attributi dei collegamenti: lunghezza, velocità libera, capacità, numero di corsie e modalità di trasporto che hanno accesso. La capacità è quella predefinita per ciascun collegamento, la quale deve essere ridotta nel caso in cui la domanda venga ridotta per la velocità di calcolo.

### 3.4.2. Import della popolazione (piani)

L'*import* della popolazione genera una popolazione sintetica in hybridPY basata sulle informazioni contenute nel file *plan.xml* di MATSim, ma viene creata solo nell'area coperta dalla rete SUMO, ovvero l'area urbana di San Francisco importata con OpenStreetMap.

Ogni attività di una persona deve essere associata a una struttura (edificio, area, ecc.), le quali sono tipicamente importate come poligoni nell'area della rete stradale SUMO. Durante l'analisi del file *plan.xml* di MATSim, vengono considerati solo i piani con una catena di attività di almeno due attività situate all'interno della rete, ed è inoltre possibile specificare l'ora di inizio e di fine delle attività importate per filtrarle. Una volta filtrata una catena di attività, vengono importati e aggiunti alla popolazione sintetica in hybridPY l'*ID* della persona rispettiva, il modo di trasporto delle *legs* e la posizione e i tempi delle attività. Dalla posizione di ciascuna attività vengono identificati la struttura più vicina e l'arco della rete SUMO più vicino.

Dalla popolazione sintetica, hybridPY genera di conseguenza i piani, i quali consistono fondamentalmente in un instradamento multimodale tra gli archi delle strutture, dove la scelta delle modalità è in base agli attributi dei piani importati. In questo modo, i piani contengono archi SUMO anziché archi MATSim. In questa fase di sviluppo non esiste una corrispondenza perfetta tra i collegamenti MATSim e gli archi SUMO perché vengono utilizzate solo le posizioni delle attività da BEAM CORE.

### 3.4.3. Importazione dei viaggi

I viaggi, nell'ambiente SUMO, sono definiti come veicoli che viaggiano in un dato momento da un arco di origine a un arco di destinazione. In contrasto con i piani, i viaggi in SUMO non contengono persone.

I viaggi vengono estratti dai piani MATSim per generare i viaggi esterni quando si utilizza l'*import* della popolazione oppure per generare scenari di simulazione microscopica basati sui viaggi. Ciò significa che le *legs* MATSim provenienti dall'esterno della rete SUMO all'interno e viceversa sono rappresentate da viaggi SUMO, stessa cosa vale per quelle di

MATSim che attraversano l'area SUMO. Infine, è possibile definire una finestra temporale per filtrare le *legs* da importare. Per ognuna, vengono identificati i bordi SUMO dove la rispettiva rotta MATSim entra o esce dall'area. A tal fine, vengono individuati tutti gli *archi frangia* della rete SUMO, i quali sono tutti gli archi che si trovano al confine della rete, in cui tipicamente c'è una inversione di marcia o un nodo cieco alla fine. Dopo aver identificato gli *archi frangia* di ingresso o uscita per ciascuna *leg*, viene trovato il rispettivo itinerario SUMO con uno dei metodi di *routing*. MATSim, quindi, viene utilizzato solo per determinare gli *archi frangia* di ingresso o uscita della rete SUMO (quindi l'origine e la destinazione) poiché non esiste corrispondenza tra gli archi SUMO e quelli MATSim.

### 3.4.4. Applicazione a San Francisco

La domanda di trasporto analizza uno scenario giornaliero del sistema di trasporto BEAM CORE calibrato e validato per l'intera area della Baia di San Francisco, in modo da simulare correttamente i viaggi a San Francisco generati e/o diretti all'esterno e quelli che attraversano la città. Lo scenario della Baia di BEAM CORE consiste nel simulare solo il 10% della popolazione attiva, il che si traduce in circa 650.000 agenti che compiono 4 viaggi al giorno con la seguente suddivisione delle modalità di trasporto:

- 51,1% auto non condivisa.
- 30,9% auto condivisa con altre persone.
- 9,3% a piedi.
- 1,6% bicicletta.
- 6,0% trasporti pubblici.
- 1,0% servizi di *ride-hailing*.

La rete importata da OSM e successivamente semplificata contiene 182.000 collegamenti e 73.000 nodi, per un totale di 46.000 chilometri. Per quanto riguarda il modello di microsimulazione, la rete SUMO è stata convertita da OSM e migliorata manualmente come descritto nella sezione di *editing* prima e dopo le varie simulazioni, invece, il trasporto pubblico è stato creato dai GTFS della San Francisco Municipal Transportation Authority.

La rete mesoscopica di BEAM CORE è stata importata in hybridPY con la popolazione sintetica e i viaggi esterni in base ai processi descritti in precedenza.

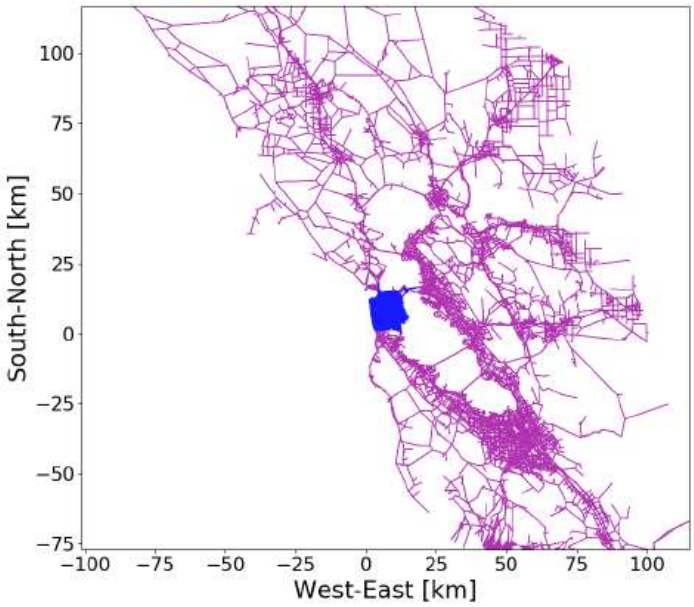


*Figura 3.12: Rete BEAM CORE della San Francisco Bay Area*

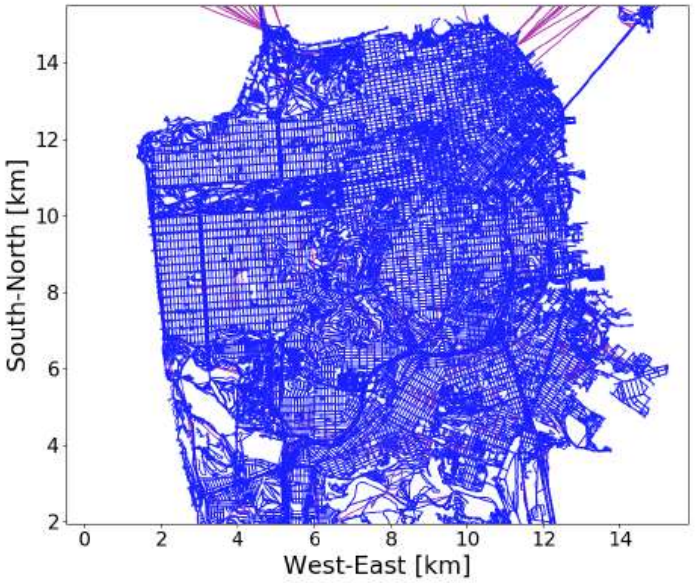


*Figura 3.13: Rete SUMO della città di San Francisco*

## 3.5. *ROUTING*

Dopo aver importato la domanda di trasporto comprensiva dei viaggi esterni e della popolazione virtuale, bisogna trasformare i viaggi in itinerari con uno dei metodi illustrati nel capitolo introduttivo. A tale scopo, il *routing* scelto è quello stocastico (più realistico), il quale su hybridPY contiene le seguenti voci o parametri principali:

- Tempo di inizio e fine in secondi.
- Modello matematico per la scelta dell'itinerario.
- Parametri del metodo *Gawron*.
- Parametri del metodo *Logit*.
- Algoritmo di *routing*.
- Massimo numero di alternative considerate.
- Numero di fili di esecuzione paralleli di *routing*.
- Tempo di precarico in secondi.

Ad essi sono stati assegnati i seguenti valori:

| | |
|---|---|
| time_begin/time_end | -1(dall'inizio alla fine) |
| method_routechoice | logit |
| beta_logit | 0,15 |
| gamma_logit | 1,0 |
| theta_logit | 0,001 |
| algorythm_routing | dijkstra |
| n_alternatives_max | 20 |
| n_threads | 0 |
| time_preload | 200 |

*Tabella 3.1: Valori stochastic routing*

Oltre alle voci indicate, ne sono presenti altre meno importanti oppure che sono state lasciate invariate. All'interno dell'elenco sono inoltre presenti delle voci con un

quadratino affianco da spuntare a seconda che si voglia abilitare o meno l'opzione (*True* o *False* nell'ambiente *scripting*).

Per il metodo di scelta del percorso viene utilizzato il metodo *Gawron* o il metodo *Logit*. L'input per ciascuno dei due metodi è una funzione di peso o costo sugli archi della rete proveniente dalla simulazione o dai costi predefiniti (nel primo passaggio o per i bordi che non sono stati ancora percorsi), oltre ad un *set* di percorsi in cui ognuno ha un vecchio costo e una vecchia probabilità (dall'ultima iterazione) e necessita di un nuovo costo e di una nuova probabilità.

L'algoritmo *Gawron* calcola le probabilità di scelta da un insieme di percorsi alternativi per ogni conducente. Vengono considerati i seguenti dati per calcolare tali probabilità:

- Il tempo di percorrenza lungo il percorso utilizzato nella fase di simulazione precedente.
- La somma dei tempi di percorrenza dei bordi per un insieme di percorsi alternativi.
- La probabilità precedente di scegliere un percorso.

Il meccanismo *Logit*, invece, applica una formula fissa a ciascun percorso per calcolare la nuova probabilità, ignorando i vecchi costi e le vecchie probabilità e considerando il costo del percorso direttamente come la somma dei costi degli archi dell'ultima simulazione.

Per quanto riguarda l'algoritmo di *routing*, quello di Dijkstra è utilizzato per trovare il percorso più breve da un nodo di partenza a tutti gli altri nodi in un grafo pesato, nel quale i pesi rappresentano le distanze tra i nodi. Un problema frequente con questo tipo di assegnazione, però, è quello che tutti i veicoli utilizzino il percorso più veloce partendo dal presupposto che siano soli nella rete e di conseguenza si bloccano nei *colli di bottiglia* a causa dell'enorme quantità di traffico. Per ovviare a questo problema, SUMO fornisce diversi strumenti al fine di determinare percorsi adatti che tengano conto dei tempi di percorrenza in una rete carica di traffico.

L'altro strumento utilizzato per il *routing* è il *Marouter*, il quale opera un'assegnazione macroscopica classica e impiega funzioni matematiche (resistive) che aumentano approssimativamente il tempo di percorrenza all'aumentare del flusso, non rendendo

necessarie lunghe simulazioni microscopiche. Viene utilizzato un metodo basato sulle penalità per trovare più percorsi che aggiunge una penalità data ad ogni arco del percorso precedentemente più breve e poi ricalcola. Se la penalità è troppo piccola potrebbe non trovare un nuovo percorso con questo metodo, quindi il numero di percorsi risultante potrebbe essere inferiore al numero desiderato. La funzione di vincolo di capacità codificata si basa su limite di velocità, numero di corsia e priorità di bordo per calcolare tempi di percorrenza e flussi in base alla densità.

All'interno di questo strumento sono presenti i seguenti parametri:

- Tempo di inizio e fine in secondi.
- Intervallo di aggregazione in secondi.
- Numero massimo di alternative.
- Algoritmo di *routing*.
- Numero di fili di esecuzione paralleli di *routing*.
- Tolleranza.
- Penalità di svolta a sinistra o di percorso.
- Estremo superiore e inferiore di costo.
- Massimo numero di iterazioni.
- Metodo di scelta del percorso.
- Parametri *Gawron* e *Logit*.

Ad essi sono stati assegnati i seguenti valori:

| | |
|---|---|
| simtime_start/simtime_end | hour_begin*3600/hour_end*3600 |
| aggregation_interval | 3600 |
| n_max_alternatives | 10 |
| routingalgorithm | dijkstra |
| n_routing_threads | 1 |
| tolerance | 0,001 |
| penalty_leftturn/penalty_paths | 0,0/1,0 |
| c_upperbound/c_lowebound | 0,5/0,15 |
| n_iter_max/n_iter_inner_max | 10/100 |
| routechoicemethod | logit |
| beta_logit | 0,15 |
| gamma_logit | 1,0 |
| theta_logit | 0,01 |

*Tabella 3.2: Valori macroscopic router*

# 4. MICROSIMULAZIONE

Dopo aver creato uno scenario con il *routing* stocastico e uno con il *macroscopic router*, si procede con la microsimulazione del traffico.

## 4.1. PARAMETRI

A tal proposito, nella sezione *Simulation* di hybridPY, se si seleziona SUMO appaiono le opzioni di simulazione seguenti:

- Modalità di interfaccia grafica.
- Tempo di inizio e fine in secondi.
- Tempo di teletrasporto in secondi che avviene in seguito ad uno stallo.
- Passo temporale in secondi.
- Tempo di campionamento dei dati in uscita in secondi.
- Tempo di riscaldamento in secondi.
- Intervallo in secondi oltre il quale i risultati sono campionati se si sceglie l'opzione di evoluzione di essi.
- *Rerouting* abilitato e relative caratteristiche (periodo, periodo prima della partenza e intervallo di aggiornamento del peso degli archi in secondi).
- Numero di passi di adattamento per il *rerouting*.

Il *rerouting* permette agli utenti di cambiare il loro percorso se in possesso di un dispositivo di navigazione in grado di visualizzare la situazione di congestione in tempo reale.

Ad essi sono stati assegnati i seguenti valori:

| | |
|---|---|
| guimode | native |
| simtime_start/simtime_end | 21600(6:00 am)/28860(8:01 am) |
| time_to_teleport | 60 |
| time_step | 0,5 |
| time_sample | 3600 |
| time_warmup | 3600 |
| time_interval_evolution | 3600 |
| is_rerouting | True |
| probability_rerouting | 0,4 |
| period_rerouting | 180 |
| preperiod_rerouting | 180 |
| adaptationinterval_rerouting | 180 |
| adaptationsteps_rerouting | 3 |

*Tabella 4.1: Parametri della simulazione*

Questo tipo di simulazione prende il nome di *En Route Simulation* perché si aggiorna durante la simulazione proprio grazie al *rerouting*.

Dopo aver utilizzato le precedenti opzioni per le due simulazioni, ne è stata prodotta un'altra a partire dallo scenario con il *macroscopic router* aumentando la probabilità di *rerouting* a 0,8.

## 4.2. RISULTATI

A seconda del tipo di *routing*, si ottengono ovviamente risultati diversi. Essi possono essere visualizzati al termine della simulazione nella sezione *Results* di hybridPY e producono risultati in termini di tempi, velocità, emissioni, rumore sia per quanto i veicoli che per gli archi. Possono anche essere esportati su Excel per essere elaborati e ricavare indicatori complessivi di rete.

Edge Results

| | ID edge | Entered | re | Left | Av. times [s] | Av. Densities [veh/km] | Av. waits [s] | Av. speeds [m/s] | Abs. Fuel [ml] | Abs. CO [mg] | Abs. CO2 [mg] | Abs. NOx [mg] | Abs. PMx [mg] | Specific fuel [l/km/h] | Specific CO [g/km/h] | Specific CO2 [g/km/h] | Specific PMx [g/km/h] | Fuel per veh. [ml/veh] | CO per veh. [mg/veh] | CO2 per veh. [mg/veh] | PMx per veh [mg/veh] | Noise [dB] |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 122 | 122 | 31 | 0 | 40 | 13.050000 | 0.680000 | 0.000000 | 10.230000 | 64.123982 | 9465.130402 | 2.174510 | 0.000000 | 78.913833 | 2974.699367 | 297.535663 | 9154.042329 | 1.348868 | 6396.493999 | 1640.011685 | 0456.930807 | 7.434937 | 56.350000 |
| 123 | 123 | 45 | 0 | 44 | 8.150000 | 1.390000 | 0.500000 | 9.340000 | 01.205384 | 0610.655252 | 2.468156 | 0.000000 | 39.889763 | 3824.367125 | 389.052558 | 1768.460449 | 1.777958 | 6796.114061 | 691.368133 | 0913.211763 | 3.159531 | 56.310000 |
| 124 | 124 | 32 | 0 | 34 | 15.610000 | 0.740000 | 0.000000 | 11.870000 | 96.120555 | 0770.726790 | 2.288130 | 0.000000 | 93.841993 | 2417.167660 | 218.176951 | 7439.408616 | 1.037309 | 4690.359294 | 1325.972480 | 5213.076170 | 6.304257 | 57.290000 |
| 125 | 125 | 34 | 0 | 34 | 5.020000 | 0.940000 | 0.000000 | 10.840000 | 06.904969 | 8473.029480 | 6.585996 | 0.000000 | 40.599533 | 1707.353346 | 151.629017 | 5254.770687 | 0.726549 | 2806.085440 | 249.206749 | 8636.370176 | 1.194104 | 52.150000 |
| 126 | 126 | 36 | 0 | 36 | 8.130000 | 0.890000 | 0.000000 | 11.580000 | 88.582318 | 8631.498138 | 5.594010 | 0.000000 | 34.066823 | 3206.602642 | 304.622812 | 9868.130588 | 1.426395 | 8533.308688 | 810.652514 | 6260.754413 | 3.795876 | 55.480000 |
| 127 | 127 | 43 | 0 | 40 | 9.340000 | 1.170000 | 0.000000 | 10.610000 | 29.583628 | 4918.882647 | 5.799377 | 0.000000 | 07.583330 | 4531.985395 | 450.450087 | 3946.207374 | 2.081662 | 0509.990317 | 1044.625178 | 2342.227892 | 4.827519 | 57.510000 |
| 128 | 128 | 191 | 0 | 189 | 9.370000 | 42.690000 | 6.000000 | 1.650000 | 78.136571 | 0846.010441 | 8.389519 | 0.000000 | 08.991109 | 9597.337781 | 645.126607 | 5610.722447 | 32.153389 | 5995.892693 | 846.217355 | 8443.989049 | 2.151719 | 57.690000 |
| 129 | 129 | 190 | 0 | 191 | 12.390000 | 6.940000 | 4.500000 | 7.940000 | 53.115771 | 8014.929312 | 2.954789 | 0.000000 | 05.017037 | 2247.114315 | 197.513235 | 7693.586553 | 5.124475 | 6345.192533 | 620.427951 | 9528.932103 | 2.654975 | 59.800000 |
| 130 | 130 | 110 | 0 | 108 | 17.430000 | 5.770000 | 4.500000 | 5.480000 | 75.000725 | 1091.242543 | 9.330931 | 0.000000 | 04.475703 | 0847.126803 | 260.579248 | 3377.465448 | 4.210657 | 9542.293866 | 1108.940445 | 9362.391497 | 3.704145 | 58.620000 |
| 131 | 131 | 94 | 0 | 94 | 6.030000 | 3.500000 | 0.000000 | 8.110000 | 71.878435 | 7534.931441 | 3.189379 | 0.000000 | 18.516862 | 5821.204699 | 561.479026 | 7916.663731 | 2.416739 | 3036.934877 | 292.924803 | 9347.161589 | 1.260818 | 54.060000 |
| 132 | 132 | 99 | 0 | 99 | 5.440000 | 3.730000 | 0.000000 | 8.080000 | 84.679885 | 6311.984312 | 7.413178 | 0.000000 | 13.212482 | 6166.432172 | 597.863765 | 8979.036882 | 2.572426 | 2741.259393 | 265.777619 | 8437.044578 | 1.143560 | 54.120000 |
| 133 | 133 | 99 | 0 | 99 | 3.080000 | 3.890000 | 0.000000 | 8.240000 | 13.019576 | 4896.908643 | 9.255153 | 0.000000 | 64.310717 | 6099.803766 | 586.954635 | 8774.202331 | 2.533913 | 1563.767875 | 150.473825 | 4813.022779 | 0.649603 | 51.700000 |
| 134 | 134 | 99 | 0 | 99 | 2.980000 | 3.980000 | 0.000000 | 8.120000 | 20.408520 | 7540.364214 | 2.305455 | 0.000000 | 75.940399 | 7331.689851 | 728.118066 | 2564.230197 | 3.152362 | 1784.044531 | 177.175396 | 5490.629348 | 0.767075 | 52.440000 |
| 135 | 135 | 63 | 0 | 53 | 27.460000 | 3.510000 | 2.500000 | 5.020000 | 55.291230 | 2811.749524 | 6.209645 | 0.000000 | 88.580594 | 7116.036018 | 817.357988 | 1896.002099 | 2.815393 | 5890.126787 | 1825.162496 | 8893.829174 | 6.286781 | 58.520000 |
| 136 | 136 | 72 | 0 | 79 | 15.240000 | 2.360000 | 5.500000 | 7.940000 | 95.377063 | 5614.801754 | 5.724712 | 0.000000 | 69.548689 | 6384.674966 | 663.783546 | 9646.346137 | 2.865163 | 2594.376341 | 1309.375940 | 8754.279319 | 5.651805 | 59.760000 |
| 137 | 137 | 54 | 0 | 57 | 10.250000 | 1.730000 | 7.000000 | 8.770000 | 87.138998 | 9983.137554 | 0.885045 | 0.000000 | 30.486017 | 3191.049685 | 325.337864 | 9819.562555 | 1.415864 | 5643.369216 | 575.359793 | 7365.889753 | 2.503954 | 56.630000 |
| 138 | 138 | 48 | 0 | 48 | 3.710000 | 1.480000 | 0.500000 | 10.030000 | 20.115809 | 5078.209024 | 6.018752 | 0.000000 | 16.243384 | 6534.715922 | 682.771822 | 0106.888613 | 3.164808 | 5000.419079 | 522.462688 | 5385.958724 | 2.421737 | 55.230000 |
| 139 | 139 | 96 | 0 | 93 | 8.750000 | 5.080000 | 1.500000 | 5.540000 | 29.649781 | 4170.189928 | 8.563736 | 0.000000 | 58.930543 | 6225.694964 | 723.982789 | 9155.360822 | 2.604992 | 3992.001139 | 464.227710 | 2282.680513 | 1.670356 | 57.360000 |
| 140 | 140 | 99 | 0 | 92 | 6.700000 | 2.810000 | 0.500000 | 10.360000 | 62.753089 | 8540.956885 | 8.207084 | 0.000000 | 14.408748 | 9992.199116 | 995.222258 | 0748.340454 | 4.565250 | 6959.917763 | 693.207270 | 1417.299479 | 3.179857 | 59.060000 |
| 141 | 141 | 357 | 0 | 352 | 10.580000 | 26.710000 | 8.500000 | 4.050000 | 06.407544 | 2437.641696 | 3.782695 | 0.000000 | 02.511232 | 4816.469646 | 783.340689 | 7880.338328 | 16.758379 | 5283.520675 | 681.811850 | 6255.042486 | 1.975685 | 59.850000 |
| 142 | 142 | 359 | 0 | 357 | 10.130000 | 13.420000 | 1.000000 | 7.780000 | 89.810905 | 9441.487501 | 2.060794 | 0.000000 | 52.281762 | 7298.160999 | 779.499525 | 4005.472588 | 12.061834 | 6029.035260 | 613.876541 | 8553.336111 | 2.663960 | 63.390000 |
| 143 | 143 | 360 | 0 | 361 | 7.760000 | 29.560000 | 4.000000 | 3.860000 | 92.242088 | 4113.594556 | 0.605698 | 0.000000 | 68.772393 | 2142.586832 | 369.811657 | 9654.724937 | 16.333533 | 3359.375205 | 428.051847 | 0335.361468 | 1.302019 | 60.220000 |
| 144 | 144 | 363 | 0 | 362 | 5.600000 | 12.100000 | 3.500000 | 8.940000 | 30.375368 | 5952.834096 | 1.915271 | 0.000000 | 00.159341 | 7794.487200 | 706.767953 | 4761.555109 | 7.945976 | 2485.273981 | 238.376410 | 7648.293910 | 1.109778 | 60.880000 |
| 145 | 145 | 95 | 0 | 95 | 17.900000 | 6.370000 | 5.500000 | 4.280000 | 45.458692 | 3034.733437 | 3.987647 | 0.000000 | 63.766590 | 2418.829091 | 616.113667 | 8205.490446 | 4.778229 | 0139.928683 | 1319.550918 | 1194.643682 | 3.901407 | 58.440000 |
| 146 | 146 | 91 | 0 | 91 | 3.680000 | 2.380000 | 0.000000 | 11.680000 | 42.277265 | 3076.503994 | 1.433160 | 0.000000 | 59.517551 | 7944.719279 | 775.533505 | 4447.395854 | 3.740154 | 3723.541508 | 363.478066 | 1458.037727 | 1.752940 | 57.030000 |
| 147 | 147 | 91 | 0 | 90 | 1.260000 | 4.340000 | 0.000000 | 8.260000 | 28.302273 | 1371.621006 | 6.676773 | 0.000000 | 96.085698 | 9860.315914 | 2176.336151 | 1106.586229 | 9.784694 | 2153.530707 | 235.988529 | 6626.022990 | 1.060992 | 55.810000 |
| 148 | 148 | 88 | 0 | 85 | 24.640000 | 4.640000 | 12.000000 | 5.290000 | 78.134802 | 0848.485440 | 2.982107 | 0.000000 | 02.309944 | 2138.628975 | 448.809676 | 7346.411182 | 5.066075 | 9599.686341 | 2339.326400 | 0301.533775 | 8.179958 | 61.460000 |
| 149 | 149 | 72 | 0 | 63 | 15.570000 | 2.220000 | 2.500000 | 9.280000 | 56.264241 | 6097.133852 | 0.850079 | 0.000000 | 74.856397 | 6938.491620 | 722.388056 | 1350.111323 | 3.233175 | 4153.559226 | 1473.571303 | 3551.261807 | 6.595228 | 61.030000 |
| 150 | 150 | 47 | 0 | 40 | 17.940000 | 1.560000 | 0.000000 | 7.550000 | 98.358441 | 9441.137832 | 8.050366 | 0.000000 | 30.749867 | 5199.166301 | 582.797578 | 5996.611036 | 2.426453 | 7263.826001 | 1935.178718 | 3116.729408 | 8.057036 | 59.270000 |

*Figura 4.1: Risultati della simulazione su hybridPY*

Siccome non è stato possibile ottenere i risultati in funzione dei veicoli ma solamente in funzione degli archi, sono stati scelti i seguenti output da inserire nel *plot* (mappa in cui vengono mostrati graficamente i dati relativi ad una grandezza) per fare un confronto tra i vari scenari:

- Veicoli entrati nell'arco.
- Densità media.
- Velocità media.

Per ogni tipo di simulazione, viene prodotto una stampa grafica relativa a queste tre grandezze. Unitamente ad esse è stata calcolata anche la velocità media della rete, ottenuta moltiplicando la velocità media di ogni arco per la lunghezza di esso (solo per quelli in cui entrano veicoli) e dividendo la somma totale per la somma della lunghezza degli archi attraversati.

La simulazione è stata condotta dalle ore 6:00 alle 8:01, però i risultati che vengono prodotti fanno riferimento solo all'ultima ora, ovvero dalle 7:00 alle 8:01 (a causa del parametro del tempo di *warm up time* di un'ora). Ciò è dovuto al fatto che all'istante zero della simulazione la rete è priva di veicoli e utenti e quindi nei primi minuti ognuno può circolare liberamente senza alcun tipo di iterazione, andando a compromettere i risultati

ottenuti. Utilizzando invece un tempo di riscaldamento di un'ora, la rete ha il tempo necessario per caricarsi e restituire risultati più adeguati.

### 4.2.1. *Stochastic routing*

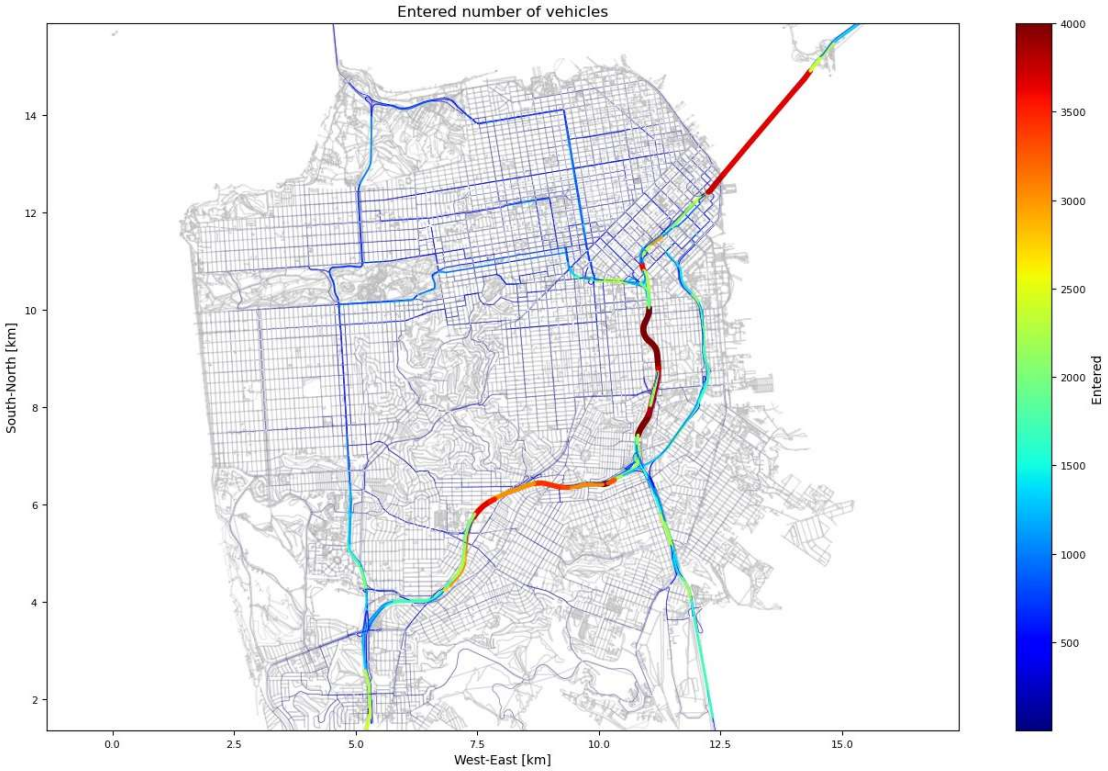


*Figura 4.2: Plot dei veicoli entrati nell'arco nello stochastic routing*

Il primo dato fa riferimento alla quantità di veicoli che accedono ad ogni arco. Come si può notare dalla scala cromatica e di spessore, gli archi utilizzati non sono numerosi e quelli che lo sono hanno un numero di veicoli in entrata considerevole.


*Figura 4.3: Plot della densità media nello stochastic routing*

Tale visualizzazione indica la densità dei veicoli per chilometro in ogni arco e dà quindi indicazioni sia sulla quantità di veicoli circolanti sia sulla possibile congestione che si può creare proprio nell'infrastruttura, se confrontato con il dato precedente e anche con il numero di corsie. Anche in questo caso la distribuzione non è ottimale poiché i pochi archi visibili hanno spesso colori caldi, i quali indicano un valore molto elevato.

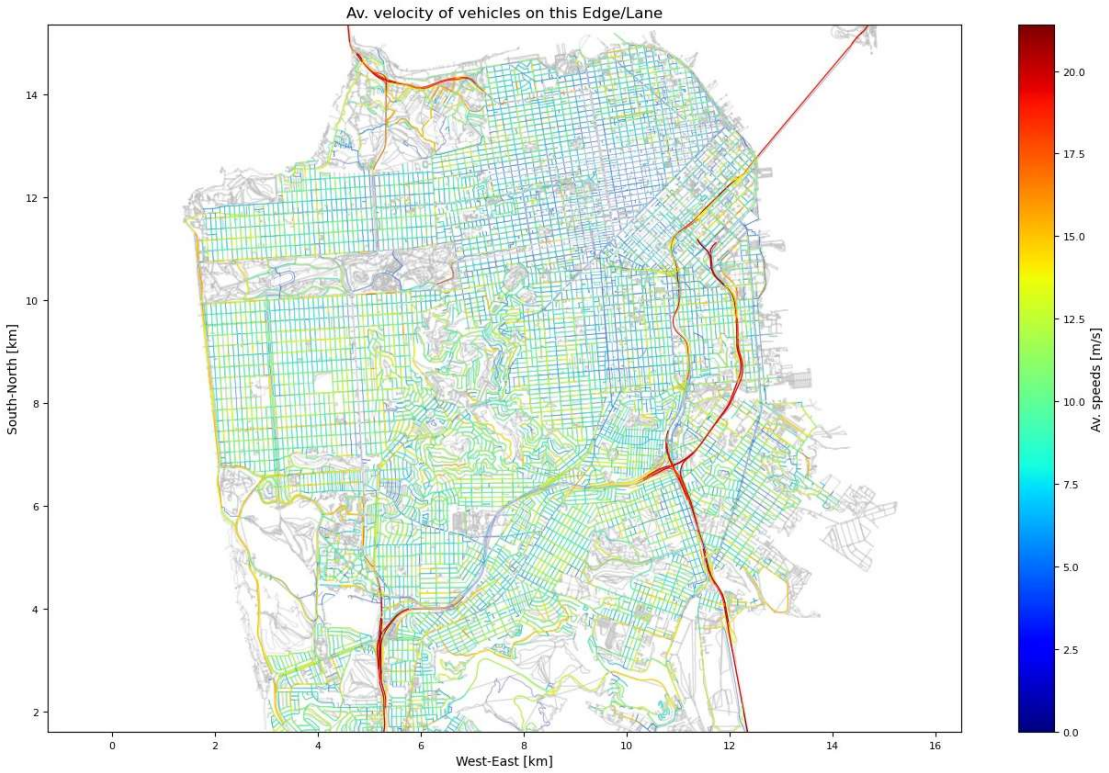


*Figura 4.4: Plot della velocità media nello stochastic routing*

Questa rappresentazione stampa la velocità media dei veicoli che percorrono un arco, la quale dipende direttamente dalla velocità ammessa sull'arco, ma è limitata dalla congestione in esso. Insieme agli altri due dati e *plot*, può dare un'idea generale su quale sia la situazione generale di traffico nella rete.

Il valore ottenuto per l'indicatore citato in precedenza è:

- Velocità media della rete: 32,02 km/h

### 4.2.2. *Macroscopic router*

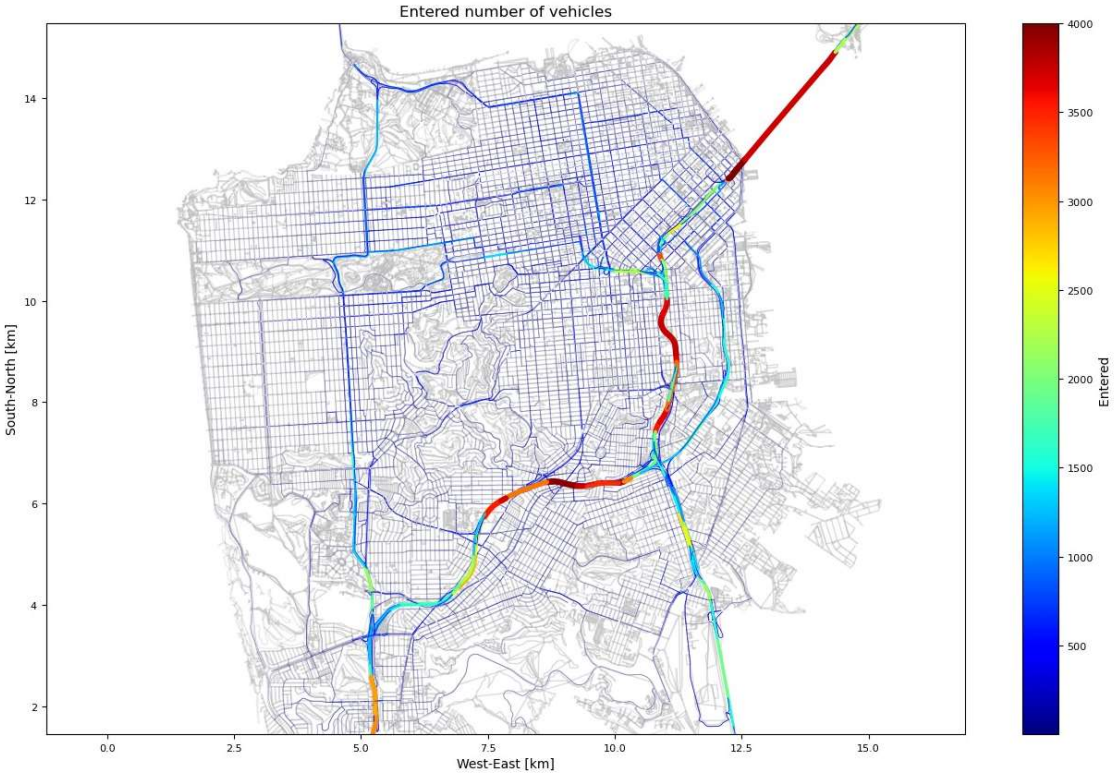


*Figura 4.5: Plot dei veicoli entrati nell'arco con il Marouter*

Facendo un confronto sulle due reti, si può notare un numero maggiore di archi cromaticamente evidenziati. Inoltre, gli archi più caricati sono contraddistinti da un colore più freddo rispetto al precedente (meno veicoli).

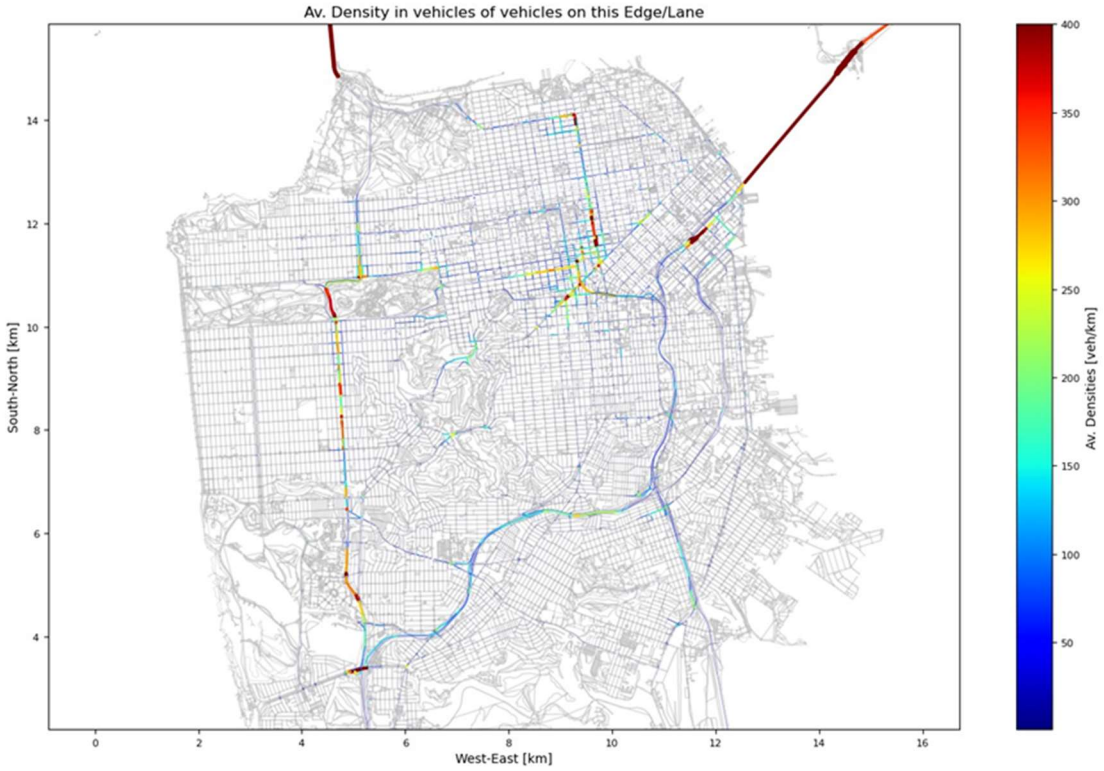


*Figura 4.6: Plot della densità media con il Marouter*

Il dato più significativo, però, è rappresentato dalla densità dei veicoli sugli archi, il quale si differenzia abbastanza da quello dello *stochastic routing*. Si può notare come nel caso precedente sia molto concentrata solo in alcune direttrici principali, mentre qui è distribuito molto di più e la gamma di colori tende principalmente ai colori freddi, che indicano un valore più basso.


*Figura 4.7: Plot della velocità media con il Marouter*

La rappresentazione grafica più vicina tra i due scenari è proprio quella relativa alla velocità media: la differenza si può notare prestando attenzione a certi archi che presentano sfumature differenti, inoltre, dando uno sguardo generale alla rete, il colore è mediamente più caldo.

Il valore ottenuto per la velocità media è:

- Velocità media della rete: 33,44 km/h (+1,42 km/h).

Dal confronto tra le grandezze e i parametri presi in esame, si può osservare come con il *macroscopic router* la distribuzione dei flussi sia migliore rispetto a quella dello *stochastic routing*.

A partire dal *macroscopic router*, è stata condotta un'ulteriore simulazione modificando il parametro relativo alla probabilità di *rerouting*, in modo tale da vedere se un aumento di esso può essere migliorativo o meno.


*Figura 4.8: Plot dei veicoli entrati nell'arco con il Marouter e rerouting probability a 0,8*

Per quanto riguarda il dato sui veicoli entrati negli archi, si può notare come la distribuzione generale sia migliore rispetto al caso precedente poiché quasi nessun arco all'interno dell'area urbana raggiunge un valore attorno ai 3500/4000 veicoli; dall'altra parte, invece, più archi raggiungono valori medio-alti di 2000/2500 veicoli.


*Figura 4.9: Plot della densità media con il Marouter e rerouting probability a 0,8*

Con il dato relativo alla densità media si ribalta un po' ciò che è stato detto nel caso precedente: se da un lato alcuni piccoli archi della rete sono più evidenziati e presenti, dall'altro lato le infrastrutture principali annoverano valori ben più alti (in certi casi anche doppi), il che fa presagire una situazione più congestionata.

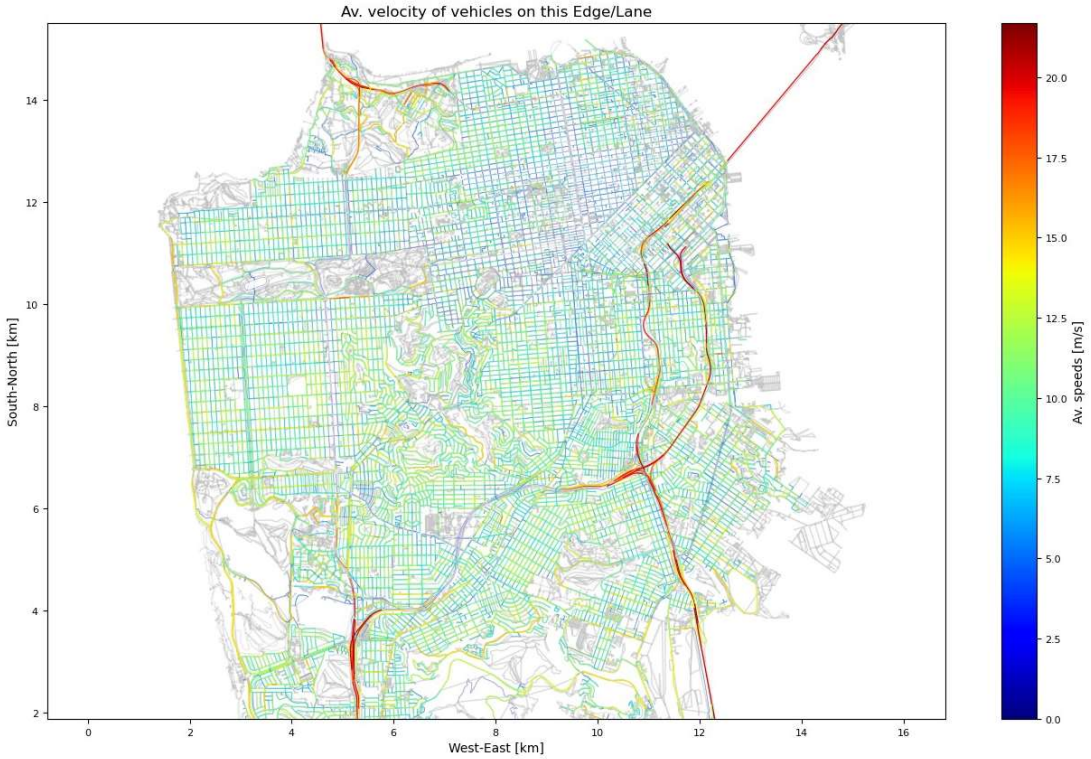


*Figura 4.10: Plot della velocità media con il Marouter e rerouting probability a 0,8*

Anche in questo caso, la rappresentazione grafica della velocità media è simile, ma con uno sguardo attento si può notare come alcuni archi presentino colori o sfumature differenti, in questo caso più tendenti ai colori freddi e quindi caratterizzati da velocità medie più basse. La somiglianza è maggiore con la rappresentazione relativa allo *stochastic routing* piuttosto che con quella dell'altro *macroscopic router*.

Il valore ottenuto per la velocità media è:

- Velocità media della rete: 32,19 km/h (+0,17 km/h rispetto allo *stochastic routing* e -1,25 rispetto al *macroscopic router* con *rerouting probability* di 0,4).

Aumentare il valore relativo alla possibilità di *rerouting*, quindi, non ha portato benefici, anzi è stato peggiorativo. La motivazione può risiedere nel fatto che si venga a creare una situazione instabile in cui un elevato numero di veicoli cambia il proprio percorso, non andando a migliorare le situazioni di congestione ma solamente spostandole da alcuni archi ad altri. Con lo strumento *Marouter* probabilmente si abbinano meglio valori di *rerouting probability* attorno allo 0,5.

# 5. VISUALIZZAZIONE DELLA RETE

Dopo aver completato lo scenario con la microsimulazione, è stato utilizzato il software Unreal Engine per produrre una rappresentazione dinamica tridimensionale del traffico a San Francisco.

*Figura 5.1: Skyline di San Francisco su Unreal Engine*

Al termine dello scenario in hybridPY, i risultati vengono esportati sul software utilizzando C++ e Blueprints. C++ è un linguaggio di programmazione ad alte prestazioni ampiamente utilizzato nello sviluppo di software complessi, giochi, applicazioni di sistema e molto altro. Blueprints, invece, è uno strumento fondamentale all'interno di Unreal Engine, progettato per permettere agli sviluppatori di creare logiche di gioco e interazioni attraverso un'interfaccia visuale intuitiva. Una delle sue caratteristiche principali è la facilità d'uso: grazie alla funzionalità *drag-and-drop* (trascina e rilascia), si possono creare, modificare e testare rapidamente le funzioni senza la necessità di scrivere codici tradizionali in linguaggi come C++, e ciò lo rende accessibile non solo a programmatori esperti. Dall'altra parte, offre anche la possibilità di integrare codici scritti in C++ per ottimizzazioni avanzate o per funzionalità non disponibili direttamente attraverso esso, in modo da sfruttare al meglio entrambi gli strumenti.

Il progetto è stato ulteriormente esteso all'integrazione con ArcGIS SDK, il quale è una *suite* di strumenti sviluppata da Esri (Environmental Systems Research Institute) per integrare mappe e dati geospaziali all'interno di progetti che consente di creare esperienze immersive e interattive sfruttando pienamente le funzionalità di GIS direttamente nell'ambiente di sviluppo di Unreal Engine. Infatti, permette di integrare mappe dinamiche e interattive direttamente nelle simulazioni, includendo la visualizzazione di mappe stradali, immagini satellitari, dati demografici e altri dati geografici.

Grazie ai dati di elevazione esportati dallo scenario, è stato possibile ripavimentare le strade andando a sostituire quelle predefinite presenti nella mappa con una nuova infrastruttura stradale creata a partire da essi. I punti singoli, dotati di coordinate xyz e di un'elevazione, però, non sono sufficienti poiché devono essere collegati tra loro: a tal fine è stata utilizzata una funzione di tipo *spline*. Di conseguenza, dai dati sull'elevazione e da quelli relativi ai veicoli provenienti da SUMO, è stato possibile migliorare la fedeltà della simulazione e garantire che il comportamento del veicolo rispondesse accuratamente e dinamicamente all'ambiente 3D e ai cambi di elevazione, viaggiando regolarmente sulle infrastrutture presenti.

*Figura 5.2: Infrastrutture prima dei dati di elevazione*

*Figura 5.3: Strade ripavimentate dopo i dati di elevazione*

Grazie agli interventi illustrati, l'appeal visivo della simulazione è stato migliorato significativamente, rendendola adatta a potenziali usi in scenari pubblicitari e dimostrativi.

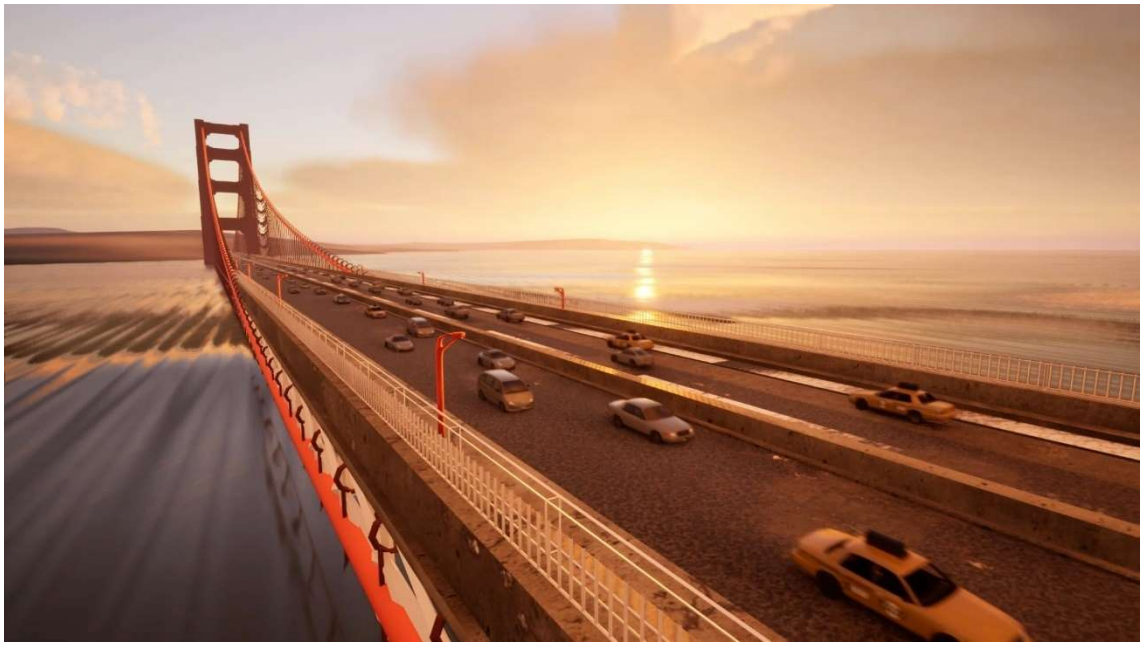

*Figura 5.4: Golden Gate Bridge con veicoli che vi transitano sopra*

# 6. CONCLUSIONI

Nel presente elaborato, è stato sviluppato un *Digital Twin* della città di San Francisco. Dopo aver importato la rete urbana da OpenStreetMap, il lavoro di *editing* della rete è stato fondamentale poiché in questi scenari così estesi ogni operazione richiede molto tempo, e quindi ogni intervento di semplificazione e snellimento della rete porta dei vantaggi a livello computazionale.

Grazie al nuovo software di simulazione hybridPY (evoluzione del precedente SUMOpy), è stato possibile importare la domanda di trasporto che in caso contrario non sarebbe stata trasferibile. Come illustrato nel capitolo della domanda di trasporto, essa proviene da uno scenario mesoscopico della San Francisco Bay Area operato dall'Università di Berkeley con il software MATSim, e proprio in questo passaggio entrano in gioco le nuove funzionalità di hybridPY, il quale è in grado di importare la domanda di trasporto da MATSim nell'ambiente SUMO. Questo vale sia per gli *external trips* sia per la *virtual population*.

Dopo aver importato la domanda è stato necessario creare gli itinerari (*routes*). Tutte queste operazioni possono essere fatte dai comandi presenti nell'interfaccia grafica oppure da uno *script* su Python. Siccome l'ultimo processo è più semplice e immediato poiché basta *accendere* o *spegnere* alcune voci a seconda di cosa si vuole eseguire, è stato scelto proprio questo.

A questo punto, è entrata in gioco la fase più lunga e laboriosa dell'elaborato, vale a dire la modifica della rete basandosi sulle microsimulazioni prova fatte con un semplice *stochastic routing* e una simulazione *En Route*, ovvero quella utilizzata anche nelle simulazioni finali. Gli interventi già illustrati sono stati volti, oltre alla semplificazione e allo snellimento della rete, al miglioramento della viabilità generale per evitare situazioni non realistiche o colli di bottiglia. Trattandosi di una rete estremamente grande per uno scenario di microsimulazione, il processo di modifica richiederebbe tempi molto più lunghi e quindi si è deciso di concentrarsi soprattutto sulle infrastrutture più caricate della rete o quelle che creavano più problemi.

Raggiunto un grado soddisfacente di definizione della rete, è stato il momento delle microsimulazioni con i conseguenti risultati. A seconda della tipologia di simulazione, sono stati ottenuti risultati differenti poiché ognuna di esse ha una metodologia diversa nel trovare l'itinerario o nell'aggiornamento di esso. Sono stati prodotti risultati relativi agli archi in termini di veicoli transitati, densità veicolare, tempi e velocità medie, inquinamento acustico e ambientale, il che rappresenta il vero vantaggio della microsimulazione rispetto agli altri modelli. Infatti, andando ad analizzare l'interazione dei veicoli, si ottiene per ogni veicolo un profilo di velocità, diretto responsabile di dati precisi riguardo alle emissioni e al rumore; nelle simulazioni macroscopiche o mesoscopiche, invece, i dati sono molto più aggregati e quindi si possono avere solamente stime piuttosto grossolane. Purtroppo, utilizzando l'ultima versione di python, ovvero Python3, ci sono alcuni bug che hanno impedito di esportare i risultati relativi ai veicoli e al trasporto pubblico, però quelli già ottenuti sono risultati molto significativi considerando uno scenario di tale portata e così ricco di dati.

Tra gli scenari confrontati, è stato possibile notare come quello con il *macroscopic router* abbia garantito una distribuzione migliore dei flussi rispetto al semplice *stochastic routing*. La differenza è stata individuabile soprattutto nel *plot* relativo alla densità, ma anche i veicoli entrati negli archi e la velocità media hanno contribuito ad una comprensione generale. Oltre ai dati grafici, sicuramente il dato più significativo è sicuramente la velocità media sulla rete, la quale ha evidenziato una differenza di 1,5 km/h, il che è piuttosto indicativo. Modificando il parametro relativo alla possibilità di riaggiornare il percorso, portandolo da 0,4 a 0,8, invece, si è registrato un peggioramento globale delle grandezze analizzato e della velocità media sulla rete.

Cambiando i parametri delle simulazioni fatte o provandone anche altre ora in via di definizione nel software (come l'*Antijammer* o il *Dynamic User Equilibrium*), si possono trovare probabilmente anche risultati migliori in termini di distribuzione e indicatori di rete.

Un ulteriore risultato raggiunto del presente progetto è la visualizzazione tridimensionale della rete grazie al software Unreal Engine. La modifica dei punti di elevazione è stata fondamentale in quanto la rete che viene scaricata grazie a OpenStreetMap non possiede dati precisi circa la quota sopra il livello del mare di ogni punto, soprattutto nei tratti in

cui più infrastrutture si intersecano. Dall'altra parte, l'informazione sulla dinamica dei veicoli proviene dai dati di simulazione su SUMO. Collaborando tra di loro, è stato possibile creare uno scenario in cui i veicoli circolano sull'infrastruttura ripavimentata in base alle elevazioni.

Per i lavori futuri, è previsto di migliorare la corrispondenza tra i collegamenti di MATSim e gli archi di SUMO tramite una corrispondenza automatizzata delle mappe dei grafi di simulazione e l'implementazione del trasferimento delle rotte da MATSim a SUMO. In questo modo, la microsimulazione potrebbe beneficiare delle rotte già assegnate, evitando di determinare un equilibrio utente con molte iterazioni dal lato della microsimulazione. Una migliore corrispondenza dei collegamenti consentirebbe anche di trasferire i tempi di percorrenza medi dei collegamenti e i flussi dalla simulazione microscopica al modello del grafo mesoscopico sia per migliorare l'assegnazione del traffico sul lato mesoscopico, sia per validare i costi e i flussi dei collegamenti. Tale confronto diretto dei risultati dei collegamenti potrebbe superare definitivamente le carenze delle simulazioni microscopiche e mesoscopiche.

# 7. FONTI: BIBLIOGRAFIA E SITOGRAFIA


- Aimsun. *Digital mobility solutions*. Recuperato da https://www.aimsun.com/
- Bentley Systems. *CUBE: Software di modellazione del trasporto*. Recuperato da https://it.bentley.com/software/cube/
- Comune di Bologna. (2023). *Documento del Gemello Digitale Urbano*. Comune di Bologna. Recuperato da https://www.comune.bologna.it/myportal/C_A944/api/content/download?id=65096dceb867430099f9438e
- Epic Games. *Architecture design software & 3D rendering visualization engine*. Unreal Engine. Recuperato da https://www.unrealengine.com/en-US/uses/architecture
- Epic Games. *Overview of Blueprints Visual Scripting in Unreal Engine*. Recuperato da https://dev.epicgames.com/documentation/en-us/unreal-engine/overview-of-blueprints-visual-scripting-in-unreal-engine
- Esri. *ArcGIS Maps SDK for Unreal Engine*. Recuperato da https://developers.arcgis.com/unreal-engine/
- Forum PA. (2023, Marzo 28). *Il gemello digitale urbano: un nuovo approccio alla gestione delle città*. Forum PA. Recuperato da https://www.forumpa.it/citta-territori/il-gemello-digitale-urbano-un-nuovo-approccio-alla-gestione-delle-citta/
- German Aerospace Center (DLR). *Dynamic User Assignment*. SUMO Documentation. Recuperato da https://sumo.dlr.de/docs/Demand/Dynamic_User_Assignment.html
- German Aerospace Center (DLR). *Edit Modes Network*. SUMO Documentation. Recuperato da https://sumo.dlr.de/docs/Netedit/editModesNetwork.html
- German Aerospace Center (DLR). *Marouter*. SUMO Documentation. Recuperato da https://sumo.dlr.de/docs/marouter.html
- German Aerospace Center (DLR). *Netedit*. SUMO Documentation. Recuperato da https://sumo.dlr.de/docs/Netedit/index.html

- German Aerospace Center (DLR). *OpenStreetMap Download*. SUMO Documentation. Recuperato da https://sumo.dlr.de/docs/Networks/Import/OpenStreetMapDownload.html
- German Aerospace Center (DLR). *OpenStreetMap File*. SUMO Documentation. Recuperato da https://sumo.dlr.de/docs/OpenStreetMap_file.html
- German Aerospace Center (DLR). *OpenStreetMap Import*. SUMO Documentation. Recuperato da https://sumo.dlr.de/docs/Networks/Import/OpenStreetMap.html
- German Aerospace Center (DLR). *SUMOPy*. SUMO Documentation. Recuperato da https://sumo.dlr.de/docs/Contributed/SUMOPy.html
- German Aerospace Center (DLR). *SUMOPy: A Python interface for SUMO*. SUMO Documentation. Recuperato da https://sumo.dlr.de/docs/Contributed/SUMOPy.html
- HTML.it. *Blueprint: Unreal Engine*. Recuperato da https://www.html.it/pag/369129/blueprint-unreal-engine/
- Joerg Schweizer, Cristian Poliziani. (2019, Novembre 13). *Generazione della domanda di trasporto*. Università di Bologna.
- Joerg Schweizer, Cristian Poliziani. (2019, Novembre 13). *Offerta e domanda nella microsimulazione*. Università di Bologna.
- Joerg Schweizer, Cristian Poliziani. (2021, Settembre 22). *Introduzione alla micro-simulazione*. Università di Bologna.
- Joerg Schweizer, Cristian Poliziani, Federico Rupi, Davide Morgano, Mattia Magi. (2021). *Building a Large-Scale Micro-Simulation Transport Scenario Using Big Data*. ISPRS Int. J. Geo-Inf.
- Joerg Schweizer, Fabian Schuhmann, Cristian Poliziani. (2024). *HybridPY: The Simulation Suite for Mesoscopic and Microscopic Traffic Simulations*. SUMO User Conference.
- MATSim. *About MATSim*. Recuperato da https://www.matsim.org/about-matsim
- PTC. *Gemelli digitali: Un approccio innovativo alla gestione dei prodotti e dei processi. PTC*. Recuperato da https://www.ptc.com/it/industry-insights/digital-twin

- PTV Group. *Simulazione del traffico con PTV Vissim*. Recuperato da https://www.ptvgroup.com/it/prodotti/ptv-vissim?utm_source=google&utm_medium=cpc&utm_campaign=it_vissim_ads&utm_content=dsa&gad_source=1&gclid=CjwKCAjw9cCyBhBzEiwAJTUWNRtp2RImKZ1b6VBTCIQ4KT1xhaLmZjOY0FqQym7iKJ9WwuLfWhYRsRoChXMQAvD_BwE
- PTV Group. *Simulazione del traffico: tutto ciò che devi sapere. PTV Group*. Recuperato da https://www.ptvgroup.com/it/aree-di-applicazione/simulazione-del-traffico
- Redazione Digital4. (2023, Giugno 5). *Digital Twin: cos'è e come funziona il modello del Gemello Digitale*. Digital4.biz. Recuperato da https://www.digital4.biz/executive/digital-twin-cose-e-come-funziona-il-modello-del-gemello-digitale/
- Systematica. *Paramics*. Recuperato da https://www.systematica.net/software/paramics/
- Visit California. *Spostarsi a San Francisco*. Recuperato da https://www.visitcalifornia.com/it/experience/spostarsi-san-francisco/
- Wikipedia contributors. *Transportation in the San Francisco Bay Area*. In Wikipedia, The Free Encyclopedia. Recuperato da https://en.wikipedia.org/wiki/Transportation_in_the_San_Francisco_Bay_Area