

POLITECNICO DI TORINO

Corso di Laurea Magistrale in Ingegneria Informatica

Tesi di Laurea Magistrale

Implementazione di un servizio automotive basato su MEC

Il caso d'uso della prevenzione degli incidenti mediante il monitoraggio
delle intersezioni stradali



Relatore

prof. Carla Fabiana Chiasserini

Candidato

Corrado Puligheddu
matricola: 246948

ANNO ACCADEMICO 2018-2019

Sommario

La tecnologia 5G promette di rivoluzionare il concetto di rete mobile. Tramite il Network Slicing, aziende verticali saranno in grado di offrire servizi del tutto nuovi altrimenti impossibili. Il progetto europeo 5G-TRANSFORMER punta a rendere possibili questi servizi. Lo scopo di 5G-TRANSFORMER è di progettare, implementare e validare una piattaforma 5G che soddisfi le esigenze dei vertical tramite l'aggregazione di reti di trasporto e di nodi di elaborazione, ospitati in infrastrutture di edge e cloud computing.

La tesi prevede lo sviluppo di una dimostrazione di fattibilità per un caso d'uso automotive con la collaborazione del progetto 5G-TRANSFORMER. Il caso d'uso proposto consiste nel servizio Extended Virtual Sensing (EVS) basato sull'architettura Multi-access Edge Computing (MEC). L'applicazione EVS realizzata si occupa del monitoraggio degli incroci stradali: l'applicazione, implementata nell'infrastruttura di rete, riceve periodicamente Cooperative Awareness Message (CAM) da veicoli connessi. I pacchetti contengono informazioni sulla posizione, velocità, accelerazione e direzione del veicolo. Tramite un algoritmo di rilevamento di collisioni basato su traiettorie, l'EVS rileva eventuali collisioni imminenti tra veicoli ed allerta tempestivamente i conducenti tramite dei Decentralized Environmental Notification Message (DENM) di allarme. In questo modo, i conducenti hanno il tempo di arrestare in sicurezza il veicolo ed evitare l'incidente.

Le prestazioni del testbed hardware-in-the-loop sono valutate usando sia l'interfaccia radio LTE OpenAirInterface sia simulando con NetEm diverse configurazioni di rete di accesso basate su Edge e Cloud computing. Il sistema basato su tecnologia MEC è stato in grado di evitare tra il 98,8% e il 100% delle collisioni, mentre facendo affidamento all'infrastruttura cloud la percentuale minima scende al 96%. I risultati confermano l'efficacia della piattaforma MEC in applicazioni automotive critiche, nelle quali la latenza è un fattore chiave. I risultati hanno dimostrato come il testbed sia un valido punto di partenza per la progettazione di un sistema affidabile di sicurezza automotive.

Indice

1	Introduzione	5
2	Tecnologie di rete e comunicazione a supporto di servizi automotive	8
2.1	Cooperative Intelligent Transport System	8
2.1.1	CAM	10
2.1.2	DENM	11
2.2	Comunicazione V2X	12
2.2.1	ITS-G5	13
2.2.2	C-V2X	13
2.3	Rete mobile LTE	15
2.3.1	Evolved UTRAN	15
2.3.2	Evolved Packet Core	17
2.4	Rete mobile 5G	19
2.4.1	5G New Radio	21
2.4.2	Multi-access Edge Computing	22
2.4.3	Network Slicing	25
2.4.4	Progetto H2020 5G-Transformer	26
3	Progettazione e sviluppo testbed	28
3.1	Vehicle Simulator	29
3.2	OpenAirInterface	30
3.3	Cooperative Information Manager	31
3.4	Extended Virtual Sensing	32
3.4.1	Algoritmo di rilevamento delle collisioni	33
3.5	DENM Decider	35
3.6	Testbed MEC	35
3.7	Testbed Openstack	36
4	Metodologia di test	39
4.1	Densità veicolare	40
4.2	Rete di accesso	41
5	Scaling in risposta all'aumento del traffico veicolare	43
5.1	Scale-up e scale-out	43
5.2	Implementazione dello scaling	44

5.3	Test delle prestazioni dopo lo scaling	46
6	Risultati	48
6.1	Consumo di CPU	48
6.1.1	EVS	49
6.1.2	DENM Decider	51
6.2	Ritardo di trasmissione	51
6.2.1	Rete di accesso	52
6.2.2	EVS	52
6.2.3	DENM Decider	55
6.2.4	Latenza End to End	56
6.3	Affidabilità	58
6.3.1	CAM ricevuti	59
6.3.2	DENM ricevuti	59
6.4	Prestazioni complessive	60
6.4.1	Veri positivi	61
6.4.2	Falsi positivi	63
6.4.3	Falsi negativi	65
7	Conclusioni	67
	Bibliografia	69

Capitolo 1

Introduzione

In questo momento, in tutto il mondo, l'accesso a Internet tramite rete cellulare costituisce un servizio fondamentale per la società. La rete di comunicazioni mobili ha trasformato il modo con cui le persone comunicano, trovano informazioni e fruiscono contenuti. Eppure la rete mobile non è più utilizzata solo da persone, ma garantisce accesso a Internet a miliardi di oggetti come altoparlanti intelligenti, televisori, sensori per smart city e anche automobili. Il numero crescente di questi dispositivi preannuncia nuove sfide per le tecnologie di telecomunicazione coinvolte nel trasferimento dei dati, dal momento che una quantità maggiore di informazioni dovrà essere trasferita tra un numero di dispositivi connessi in aumento, in alcuni casi mantenendo i ritardi di comunicazione all'interno di limiti più stringenti rispetto a quelli che la tecnologia attuale consente.

Il 5G, la prossima tecnologia di telecomunicazione mobile cellulare, promette di soddisfare le esigenze per la telecomunicazione wireless del futuro, consentendo latenze inferiori e maggiore velocità e affidabilità. Il 5G, termine con cui si intende la quinta generazione di rete di telecomunicazione cellulare, promosso dall'organizzazione International Telecommunication Union (ITU), si pone come successore dell'attuale tecnologia 4G, conosciuta col nome commerciale LTE (Long Term Evolution). Rispetto alla quarta generazione, vengono introdotte migliorie in: velocità, latenza, capacità, affidabilità, flessibilità ed efficienza [1]. Il 5G rivoluzionerà la comunicazione mobile rispetto a come la conosciamo oggi, consentendo applicazioni fino a qualche anno fa impensabili: IoT, smart city, automobili connesse, realtà virtuale sono solo alcune di queste. La promessa è di espandere le possibilità delle reti mobili e di moltiplicare i servizi che esse possono offrire.

Il Multi-access Edge Computing (MEC) è considerato uno dei pilastri chiave per il rispetto degli elevati requisiti del 5G. Già utilizzato con le reti 4G, è una recente architettura di rete decentralizzata, promossa dall'ETSI (European Telecommunications Standard Institute), che permette le capacità del cloud computing in prossimità dell'utente che le fruisce, sul limite (edge) della rete di accesso. Mentre nella quarta generazione il MEC è stato pensato come un componente aggiuntivo, nel 5G è previsto tra i requisiti iniziali come tecnologia fondamentale. La minore distanza dell'utente dal nodo di elaborazione consente una minore latenza e congestione della rete di trasporto, migliorando pertanto le prestazioni complessive delle applicazioni [2, 3]. Il MEC ha molti punti in comune con il concetto di Network Functions Virtualization (NFV): entrambi trovano fondamento nella tecnologia di virtualizzazione dell'hardware e fanno uso del processo di orchestrazione: in

base alla potenza di calcolo e le capacità di rete disponibili nei nodi di elaborazione, l'orchestratore è in grado di determinare quali risorse allocare al software virtualizzato per garantirne un funzionamento ottimale, tenendo sempre presente i suoi requisiti.

Una delle applicazioni rese possibili dall'uso congiunto delle tecnologie 5G e MEC è un sistema di previsione e prevenzione delle collisioni in tempo reale (ICA, Intersection Collision Avoidance). Ricade in uno dei principali ambiti d'uso del Intelligent Transport System (ITS, ETSI), che fa affidamento sulla comunicazione Vehicle to Everything (V2X) per migliorare la sicurezza, l'affidabilità, l'efficienza e qualità non solo del trasporto su strada, ma anche di altri domini come quello ferroviario, marittimo e dell'aviazione [4].

Il sistema ICA prevede che periodicamente i veicoli, tramite modem cellulari installati a bordo, ma anche pedoni o ciclisti, mediante applicazione su smartphone, inviino Cooperative Awareness Message (CAM, ETSI [5]), che contengono informazioni relative al loro stato. Le informazioni contenute nei messaggi ricevuti dalla stazione base sono salvate ed integrate con i dati provenienti da sensori eventualmente in funzione nell'area, dopodiché vengono elaborate da un software che calcola le traiettorie degli utenti della strada e ne prevede eventuali collisioni.

Le informazioni più utili per un sistema ICA sono posizione, velocità, accelerazione e direzione, ma lo standard consente anche molte altre informazioni aggiuntive, come lo stato delle luci (ad esempio l'attivazione delle luci di emergenza) o le condizioni del manto stradale.

Nel caso venga prevista una collisione, un messaggio di allarme, inserito in un Decentralized Environmental Notification Message (DENM, ETSI [6]), viene immediatamente generato ed inviato ai veicoli coinvolti, per essere mostrato ai conducenti tramite predisposta HMI (Human-Machine Interface). Questi avranno a disposizione il tempo per arrestare il veicolo in sicurezza ed evitare la collisione. Un sistema simile può costituire un fondamentale ausilio alla guida semi-autonoma, intervenendo dove gli apparecchi ADAS mostrano le loro limitazioni.

La tesi si svolgerà nell'ambito del progetto europeo 5G-Transformer, il cui scopo è di promuovere il paradigma del Network Slicing sulle esigenze delle cosiddette "industrie verticali". Nello specifico ci si focalizzerà sul sopracitato sistema ICA, come dimostrazione di un caso d'uso relativo all'industria automotive. Tra i partner del progetto figurano: Centro Ricerche FIAT - FCA, Nextworks, CTTC, EURECOM e naturalmente il Politecnico di Torino [7, 8]. Il lavoro consisterà nella analisi approfondita di un prototipo di una possibile implementazione di un sistema ICA, mediante lo studio delle prestazioni di una piattaforma di test hardware in the loop, realizzata presso il Dipartimento di Elettronica e Telecomunicazioni del Politecnico di Torino, che simula un comportamento realistico di veicoli, apparati di telecomunicazione, calcolatori e software. L'obiettivo sarà di constatare quali siano le possibilità di tale sistema, in particolare in termini di affidabilità, e di collaborare con i partner coinvolti per migliorarne gli eventuali punti deboli, al fine di promuoverne un futuro impiego su strada.

Tale soluzione viene incontro alla necessità di migliorare la sicurezza in strada, sia per gli automobilisti che per i pedoni. Secondo la World Health Organization, ogni anno 20-50 milioni di persone rimangono ferite e circa 1,35 milioni muoiono a seguito di incidenti stradali, che si confermano come la principale causa di morte per bambini e adulti di età compresa tra i 5 e i 29 anni. Gli incidenti stradali comportano inoltre perdite economiche per gli individui, le loro famiglie e l'intera nazione: si stima fino al 3% del PIL [9]. Questi

numeri continuano a spingere la ricerca incessante di soluzioni, non solo tecnologiche, per arginare il problema.

Nel Capitolo 2 sono presentate le tecnologie di comunicazione attualmente impiegate in campo automotive. Il punto di partenza è il Cooperative - Intelligent Transport System, l'applicazione che tramite degli appositi messaggi scambiati dai veicoli permette di realizzare una guida cooperativa più efficiente e sicura. Alla base della comunicazione dei veicoli c'è una tecnologia V2X, che attualmente può essere ITS-G5, basato su 802.11p, oppure C-V2X basato su LTE. Dal momento che il testbed fa uso di OpenAirInterface, che comprende interfaccia radio e Evolved Packet Core (EPC) LTE, si presentano i componenti chiave della rete mobile di 4^a generazione, e poiché il testbed sfrutta il concetto di Edge Computing, si introduce anche il 5G, limitatamente agli aspetti che caratterizzano le novità più rilevanti.

Nel Capitolo 3 è esposto il testbed automotive nelle sue due configurazioni principali. Vengono descritti i principali elementi che lo compongono con attenzione particolare all'algoritmo impiegato per il rilevamento delle collisioni.

Il Capitolo 4 presenta i test svolti sul testbed automotive come parte del lavoro di tesi. Le analisi sono svolte variando i parametri che più influenzano le prestazioni del testbed, in particolare la densità dei flussi veicolari simulati e la tipologia di rete di accesso su cui transitano i CAM inviati e i DENM ricevuti dai veicoli prima di giungere a destinazione.

Il Capitolo 5 è interamente dedicato alla soluzione di scaling out sviluppata dal candidato per il testbed Openstack. Tramite il collegamento ad un System Orchestrator remoto che monitora lo stato di funzionamento del sistema, quando viene rilevato che le prestazioni calano eccessivamente a seguito di un incremento del traffico veicolare, vengono dinamicamente aumentate le risorse assegnate al sistema per preservarne il funzionamento ottimale.

I risultati dei test si trovano nel Capitolo 6; le conclusioni si trovano infine nel Capitolo 7.

Capitolo 2

Tecnologie di rete e comunicazione a supporto di servizi automotive

L'industria dell'automobile sta attraversando un periodo di profondo cambiamento: da un lato, vediamo sempre più veicoli sono connessi alla rete Internet e l'uno con l'altro, dall'altro la guida autonoma viene migliorata costantemente e offre sempre più funzioni. Per affrontare le situazioni di traffico più complesse, le auto autonome dovranno basarsi non solo sui dati raccolti dai propri sensori, ma dovranno fare affidamento anche sulle informazioni condivise dagli altri veicoli, allo scopo di prendere decisioni in modo cooperativo. La transizione da sistemi autonomi a sistemi cooperativi è il concetto alla base del Cooperative Intelligent Transport System. La chiave per consentire ai veicoli di coordinarsi cooperativamente è la tecnologia di rete mobile: la comunicazione Vehicle to Everything (V2X) consente ai veicoli di interagire l'uno con l'altro, con l'infrastruttura stradale e con la rete stessa. Le nuove reti mobili permettono prestazioni migliori, latenze inferiori e migliore robustezza non solo con il miglioramento dell'interfaccia radio, ma anche con l'integrazione di nuovi paradigmi come l'Edge Computing e il Network Slicing. Per questo con il 5G vengono studiati nuovi casi d'uso che facciano da apripista a nuove funzionalità in una moltitudine di settori, tra cui quello automotive.

2.1 Cooperative Intelligent Transport System

Il Cooperative Intelligent Transport System (C-ITS), noto anche come Connected Vehicle Technology negli Stati Uniti, è un'applicazione che fa uso di connettività V2X per aumentare la sicurezza nelle strade e migliorare l'efficienza del traffico, aiutando chi guida a rendersi conto dei veicoli circostanti, distribuendo avvisi per situazioni di pericolo in strada e fornendo informazioni in tempo reale sulle condizioni del traffico. Tipicamente le applicazioni C-ITS fanno uso di connettività always-on tra i veicoli vicini e con le infrastrutture a bordo strada per consentire uno scambio frequente di informazioni.

In Europa, il primo programma di ricerca per il C-ITS risale agli anni 80, con il progetto PROMETHEUS (1987-1994) che segna l'inizio di un sistema di guida cooperativo. Faceva uso di comunicazione interveicolare intorno alla frequenza di 57 GHz. A partire dagli anni 2000, spinti dalla disponibilità di tecnologie più all'avanguardia come GPS, WiFi e sistemi embedded, sono partiti nuovi programmi di ricerca, che in Europa sono stati più di 40.

Nel 2016, la Commissione Europea ha individuato due fasi per lo sviluppo iniziale del C-ITS [10]:

1. Day 1, che contiene l'elenco dei servizi che dovrebbero essere inclusi al lancio iniziale del servizio, comprende avvisi di situazione pericolose e notifiche di segnali stradali. La commissione prevede che ogni euro investito in servizi Day 1 genererà un ritorno di 3 euro, e consiglia pertanto una messa in campo rapida.
2. Day 1.5, i cui servizi potrebbero essere disponibili in un secondo momento poiché ancora in sviluppo, comprende la protezione degli utenti della strada più vulnerabili, segnalazione di punti di interesse con particolare attenzione ai parcheggi e navigazione intelligente.

Il consorzio CAR 2 CAR, che riunisce produttori di veicoli, fornitori e istituti di ricerca europei, molto attivo nello sviluppo del C-ITS, ha definito l'evoluzione del ITS attraverso cinque fasi, spingendosi temporalmente oltre la finestra del lancio iniziale [11]:

1. Awareness driving: vengono condivise le informazioni di stato, così che veicoli siano consapevoli di chi li circonda e di eventuali pericoli rilevati nella strada;
2. Sensing driving: si condividono informazioni dai propri sensori, in questo modo i veicoli conoscono ciò che non possono rilevare direttamente, come oggetti nascosti, e hanno percezione migliore dell'ambiente circostante;
3. Cooperative driving: ci si coordina tramite la condivisione delle proprie intenzioni, in questo modo i veicoli autonomi possono predire più accuratamente il comportamento degli altri utenti, ottimizzando le decisioni e le manovre;
4. Synchronized Cooperative Driving: coordinandosi tramite la condivisione delle proprie iniziative, i veicoli autonomi sono in grado di sincronizzare le loro traiettorie per ottenere una guida ottimale;
5. Accident-free Driving: guida automatizzata con flussi di traffico ottimizzati, ogni funzione di guida può essere svolta in modo autonomo.

Gli standard C-ITS sono fondamentali per garantire interoperabilità tra dispositivi di comunicazione di costruttori diversi; in Europa, lo standard principale è specificato nel documento ETSI EN 302 665 [12]. L'architettura descritta prevede un elemento di base, chiamato ITS station, in grado di comunicare con altre stazioni. Ogni sottosistema ITS, che sia un dispositivo personale tascabile, un veicolo, un elemento a bordo strada (cartello stradale o tutor, ad esempio) o una stazione centrale, integra una stazione ITS. La stazione ITS è costituita da tre livelli: accesso, rete e trasporto, e servizi.

Access layer rappresenta lo strato ISO-OSI 1 e 2, utilizza una combinazione di tecnologie IEEE note come ITS-G5 [13]. Non vengono tuttavia escluse altre tecnologie di accesso,

come quelle basate su rete cellulare. Maggiori dettagli su ITS-G5 sono riportati nel sottoparagrafo 2.2.1.

Networking and transport layer equivalente agli strati 3 e 4 dello stack OSI, usa come protocollo di rete GeoNetworking, che prevede indirizzamento geografico (GeoUnicast, GeoBroadcast, GeoAnycast e topologically-scoped-broadcast) e comunicazione single-hop e multi-hop [14]. La capacità di indirizzamento geografico consente di inviare un pacchetto a una stazione ITS nella stessa area geografica del mittente o a tutte le stazioni che si trovano in un'altra area, di cui è possibile specificare coordinate e forma geometrica (cerchio, rettangolo o ellisse) [15]. Come protocollo di trasporto è invece utilizzato Basic Transfer Protocol (BTP), connection-less e best-effort che consente di fare distinzione sui diversi protocolli di livello servizio [16]. Anche in questo caso non viene precluso l'uso di protocolli alternativi, come quelli tipici di Internet (IPv6, TCP, UDP).

Facility layer si colloca negli strati 5, 6 e 7 della pila OSI, comprende il protocolli Cooperative Awareness Message (CAM) e Decentralized Environmental Notification Message (DENM) [5, 6], descritti più nel dettaglio nel sottoparagrafi 2.1.1 e 2.1.2. Oltre a CAM e DENM, altri protocolli di livello servizio sono stati standardizzati di recente: RTL (Road and Lane Topology), per la topologia statica delle strade, TLM (Traffic Light Maneuver), per aiutare i veicoli a compiere manovre sicure agli incroci, TLC (Traffic Light Control), per il controllo dei segnali stradali, in particolare dei semafori, da parte di veicoli di emergenza e per il trasporto pubblico, e IVI (Infrastructure to Vehicle Information), per inviare ai veicoli segnali come limiti di velocità o avvisi di lavori in corso [17].

Nella fase iniziale dello sviluppo del C-ITS sono stati identificati dei casi d'uso base (Basic Set of Applications, BSA), classificati in quattro gruppi applicativi: sicurezza stradale attiva, efficienza del traffico cooperativa, servizi locali cooperativi e servizi Internet globali [4]. L'Intersection Collision Warning, il caso d'uso oggetto di questa tesi, ricade nel primo gruppo. I requisiti prevedono che:

- i veicoli siano in grado di inviare in broadcast, di ricevere e processare CAM V2X;
- se è ostruita la linea visiva tra i veicoli, sia presente una unità a bordo strada che possa inoltrare il segnale o che possa rilevare e segnalare un rischio di collisione;
- la posizione dei veicoli sia accuratamente riportata su una mappa digitale;
- frequenza minima dei messaggi periodici sia di 10 Hz;
- tempo di latenza sia inferiore a 100 ms.

2.1.1 CAM

Il Cooperative Awareness Message (CAM) è uno dei due protocolli di livello servizi, standardizzato in ETSI EN 302 627-2 [5]. Il CAM è un messaggio periodico che le stazioni ITS inviano in broadcast alle stazioni nelle vicinanze. La sua trasmissione è attivata quando la stazione entra in una situazione che richiede sicurezza, che per un'auto coincide col

momento in cui il motore viene avviato. Un CAM è composto da un header ITS PDU (Packet Data Unit) e da diversi container che raggruppano i dati trasmessi a seconda della frequenza con cui variano e del ruolo del mittente.

Header ITS PDU contiene la versione del protocollo, che sarà usata dal destinatario per decodificare il messaggio, un identificativo del tipo del messaggio, pari a 2 per i CAM, ed un identificativo della stazione ITS mittente.

Basic container contiene le informazioni base della stazione ITS mittente: il tipo della stazione e la sua posizione geografica registrata durante la generazione del CAM. La posizione è associata ad un timestamp prefisso al container.

High frequency container ospita le informazioni più dinamiche della stazione, come la direzione, la velocità, la corsia, l'angolo di sterzo. Dal momento che variano di frequente, il container deve essere presente in ogni CAM generato.

Low frequency container contiene le informazioni che variano lentamente, che sono meno rilevanti ai fini della sicurezza o che sono molto pesanti, dunque non sempre viene aggiunto nel CAM. Queste includono il ruolo del veicolo, ad esempio trasporto pubblico o veicolo di emergenza, lo stato delle luci esterne e le ultime strade percorse.

I container, con il fatto che alcuni di questi siano opzionali, consentono di realizzare un messaggio modulare che riesce a contenere tutte le informazioni importanti minimizzando il carico sul canale di trasmissione. La frequenza di generazione dei CAM per i veicoli è compresa tra 1 Hz, corrispondenti ad un intervallo di 1000 ms, e 10 Hz, ovvero 100 ms. All'interno di questi limiti, l'intervallo di tempo tra due CAM consecutivi dipende dalla dinamica della stazione ITS e dallo stato di congestione del canale radio. Ad esempio, il numero di CAM trasmessi può aumentare in prossimità di un incrocio, per aumentare la probabilità che le altre stazioni lo ricevano. Le condizioni di trasmissione dei CAM devono essere verificate almeno ogni 100 ms; un low frequency container deve essere incluso in un CAM almeno ogni 500 ms.

2.1.2 DENM

Standardizzato nel ETSI EN 302 627-3, il Decentralized Environmental Notification Message (DENM) è il secondo protocollo di livello servizi del C-ITS [6]. Il DENM è generato in corrispondenza della rilevazione di un evento da parte di una stazione ITS, come un pericolo, allo scopo di avvisare le altre stazioni presenti all'interno dell'area coinvolta. La struttura del messaggio è simile a quanto visto con il CAM: sono presenti container obbligatori ed opzionali.

Header ITS PDU molto simile al corrispettivo header del CAM: contiene la versione del protocollo, un identificativo del tipo del messaggio, pari a 1 per i DENM, e l'identificativo della stazione che genera il messaggio o che lo sta inoltrando.

Management container è un container obbligatorio che include i dati necessari a descrivere le caratteristiche base di un evento. Quando una stazione rileva un evento che richiede attenzione, dalla stazione gli viene assegnato un identificativo univoco, che viene incluso nel container. Inoltre sono inclusi il tipo di stazione, tempo di rilevamento e la posizione.

Situation container opzionale, contiene informazioni che descrivono l'evento. Per identificare il tipo di evento vengono utilizzati due codici: `causeCode` e `subCauseCode`. Nel caso l'evento identifichi una possibile collisione tra due veicoli ad un incrocio, questi codici sono rispettivamente 97 (collision risk) e 2 (crossing collision). Se disponibili, vengono incluse le posizioni precedenti in cui è stato identificato l'evento prima della posizione attuale e le cause correlate.

Location container contiene informazioni sulla velocità e direzione dell'evento, e l'itinerario che conduce alla posizione dell'evento. Nel caso di una collisione, si tratta quindi delle traiettorie dei veicoli coinvolti.

À la carte container consente di includere informazioni non presenti negli altri container o dati di una applicazione specifica, come ad esempio la corsia in cui è posizionato l'evento, informazioni per la mitigazione dell'impatto (posizione dei passeggeri, centro di massa del veicolo, posizione dei montanti), temperatura ambientale, ecc.

Il protocollo DENM gestisce il ciclo di vita di un evento: una volta che una stazione genera un identificativo univoco, questo viene riutilizzato per eventuali aggiornamenti sull'evento. Inoltre, un evento può essere cancellato dalla stazione che per prima l'ha identificato o negato da una stazione terza. Sono presenti diversi meccanismi che consentono di distribuire l'informazione all'interno dell'area di interesse durante il ciclo di vita dell'evento: la stazione che lo identifica può ripetere il DENM, solitamente a una frequenza più bassa del CAM, per consentire alle nuove stazioni che entrano nell'area di ricevere l'informazione, e se la stazione originaria lascia l'area o se smette di inviare DENM, un'altra stazione può prendere il suo posto.

2.2 Comunicazione V2X

Le comunicazioni vehicle-to-vehicle (V2V) e vehicle-to-infrastructure (V2I), collettivamente chiamate V2X, costituiscono la base del C-ITS e hanno il potenziale di migliorare la sicurezza e l'efficienza del traffico. I ricetrasmittitori utilizzati nelle comunicazioni V2X chiudono il gap tra i sensori line-of-sight (LOS), come camere, radar e lidar, e le comunicazioni cellulari a lungo raggio. I sensori LOS non riescono a vedere oltre barriere fisiche o predire il comportamento di oggetti tracciati, ma queste limitazioni possono essere superate dal V2X, che può fornire, nel giro di millisecondi, informazioni su auto dietro l'angolo o nascoste dietro altri veicoli. I veicoli inoltre, tramite protocolli C-ITS, possono ricevere informazioni riguardo le intenzioni delle altre stazioni, con cui poi adattare il proprio comportamento in strada.

Nell'agosto 2008 la Commissione Europea ha allocato una parte dello spettro a 5.9 GHz (5875 - 5905 MHz) per applicazioni di sicurezza in ambito ITS, sebbene ancora non sia chiaro quale tecnologia radio V2X andrà ad utilizzarla. Al momento, i candidati sotto studio sono due: ITS-G5, basato su IEEE 802.11p e promosso dall'ETSI, e C-V2X, basato su LTE e recentemente proposto come alternativa dal 3GPP, che sta già sviluppando una evoluzione basata sul 5G NR. Entrambe le tecnologie hanno vantaggi e svantaggi, e, poiché nessuna domina sull'altra, si sta pensando di farle coesistere tramite un approccio ibrido: si suddivide la banda disponibile in 3 blocchi da 10 MHz l'uno e si assegna a ciascuna

tecnologia un blocco; il terzo blocco è usato da entrambe le tecnologie secondo una tecnica detect and vacate da definire [18].

2.2.1 ITS-G5

L'ITS-G5 utilizza un adattamento per l'Europa del prodotto del progetto statunitense WAVE (Wireless Access in Vehicular Environments), che aveva definito delle modifiche allo standard IEEE 802.11, approvate poi con il nome 802.11p nel 2010, per supportare i requisiti dei canali radio Dedicated Short-Range Communications (DSRC) [13, 19]. È stata la prima tecnologia pensata per lavorare tra veicoli (V2V) e tra veicoli ed infrastruttura (V2I), creando una rete ad-hoc (VANET) nel momento in cui due stazioni V2X entrano nel range una dell'altra e pertanto non richiedendo alcuna infrastruttura aggiuntiva di telecomunicazione.

Al livello fisico, ITS-G5 usa multiplexing a divisione di frequenze ortogonali (OFDM) che fa uso di un grande numero di sottoportanti ortogonali che portano l'informazione su flussi multipli paralleli. Il segnale complessivo è a banda larga (10 MHz) e consente un bitrate teorico da 3 Mb/s fino a 27 Mb/s a seconda dello schema di modulazione e della codifica utilizzati.

Il livello MAC si chiama Enhanced Distributed Coordination Access (EDCA), segue lo stesso approccio della Distributed Coordination Function (DCF) impiegata dal layer MAC usato dal 802.11 base, che usa il noto CSMA/CA, ma in più aggiunge attributi per la gestione della Quality of Service. Con il Collision Sensing Multiple Access with Collision Avoidance (CSMA/CA), una stazione che vuole trasmettere si mette prima in ascolto sul canale; se lo trova occupato, ritarda la trasmissione di un tempo casuale e poi riprova. In questo modo più stazioni possono gestire l'accesso al canale in modo distribuito e non coordinato, non avendo bisogno di sincronizzazione e preallocazione delle risorse. Lo svantaggio è che al crescere del numero trasmissioni aumentano le collisioni, con la conseguenza che la congestione del canale comporta un drastico calo delle prestazioni.

2.2.2 C-V2X

Il Cellular V2X è una tecnologia di accesso radio sviluppata dal 3GPP a partire dalla Release 14 [20]. La versione attuale è chiamata LTE-V2X (Rel. 14 e 15) ed è basata su tecnologia 4G, la prossima, basata su 5G, sarà rilasciata nella Release 16 e si chiamerà NR-V2X. Fa affidamento all'infrastruttura di comunicazione mobile cellulare ormai diffusissima, tuttavia, considerando che la copertura non è totale, è stata progettata per operare anche in sua mancanza. Alla modalità operativa tradizionale su bande licenziate tramite stazione base, chiamata Network, si affianca una modalità complementare, chiamata Direct, indipendente dalla rete cellulare e che consente la comunicazione diretta tra i dispositivi mediante "sidelink". La comunicazione Direct su banda 47 (che include ITS 5.9 GHz) è consentita con la modalità 4 della interfaccia PC5; questo tipo di comunicazione è totalmente assimilabile ad una VANET. Le interfacce radio sono ora descritte.

- **LTE-Uu** è l'interfaccia radio tradizionale tra User Equipment (UE) e eNodeB. Se due UE vogliono comunicare usando questa interfaccia radio, il messaggio deve essere inviato in uplink all'eNodeB che a sua volta lo invia in downlink all'UE di destinazione.

Dato che la maggior parte del traffico è periodica, si può usare uno schema di scheduling semi persistente (SPS) per cui le risorse radio vengono assegnate non solo per la trasmissione imminente, ma anche per le trasmissioni successive. È l'interfaccia adatta per comunicazioni a medio raggio (distanza superiore a 1 km).

- **PC5** consente la comunicazione tra due UE senza richiedere che ogni pacchetto passi per l'eNodeB. Consta di quattro modalità, di cui solo le modalità 3 e 4 sono utilizzabili. Le modalità 1 e 2 sono infatti state pensate con l'obiettivo di prolungare la carica della batteria dei dispositivi mobili al costo di un aumento di latenza. I veicoli connessi richiedono comunicazioni ad alta affidabilità e bassa latenza, perciò tali modalità non sono adatte.

1. Mode 3: l'allocazione delle risorse per la trasmissione sidelink è gestita dall'eNodeB, pertanto questa modalità è disponibile solo in uno scenario in cui sia presente copertura. Sono disponibili tre tipi di scheduling:
 - semi-persistent, come per l'interfaccia LTE-Uu, dal momento che è una funzione offerta dal eNodeB;
 - UE-report based, che prevede che gli UE inviino le loro osservazioni sulla occupazione del canale radio all'eNB così da assisterlo nella allocazione delle risorse del canale sidelink;
 - cross-carrier: se un operatore ha più di un carrier a sua disposizione, l'eNodeB può programmare risorse su uno dei carrier per la trasmissione sidelink sugli altri carrier.
2. Mode 4: è disponibile quando lo UE si trova in mancanza di copertura. Tramite un algoritmo di riservazione, lo UE si assegna autonomamente le risorse radio. L'algoritmo prevede che ogni UE si metta in ascolto sul canale radio per un secondo, dopodiché, elaborando la lettura, si riserva le risorse radio in modo semi-permanente al fine di minimizzare la collisione dei pacchetti. La sincronizzazione tra gli UE avviene mediante l'uso di sistemi satellitari globali di navigazione, come GPS o Galileo [21].

La comunicazione cellulare è storicamente stata asimmetrica, LTE non fa eccezione. Mentre in downlink è usato uno schema di modulazione OFDM, in uplink, per ragioni di efficienza di trasmissione e costo degli amplificatori dei terminali, si è optato per il Single-carrier FDMA (SC-FDMA). La comunicazione LTE in uplink usa a livello fisico una divisione in frequenza della banda radio in sottoportanti (subcarrier) ortogonali contigue, mentre nel dominio del tempo i dati sono organizzati in frame di durata 10 ms, ciascuno dei quali è diviso in subframe di 1ms. Un subframe è composto di due slot di 0,5 ms, in ciascuno dei quali sono trasmessi 7 simboli OFDM. Uno slot e 12 subcarrier consecutivi (180 kHz) compongono un Resource Block (RB). Siccome l'allocazione avviene usando come base il subframe, l'allocazione minima è di 2 RB, che corrispondono a 14 simboli nel dominio del tempo. Di questi, solo 9 sono disponibili per la trasmissione di dati per LTE-V2X. Pertanto, a livello fisico, la minima unità di allocazione di risorse è pari a 108 simboli complessi QAM. Il bitrate ottenibile varia in funzione dello schema di modulazione e codifica (MCS) utilizzato, la dimensione dei dati da trasmettere e dalla banda del canale radio utilizzato. Ipotizzando pacchetti da 200 bytes e banda di 10 MHz, il bitrate può arrivare fino a 17,71 Mb/s [22].

2.3 Rete mobile LTE

La rete Long Term Evolution appartiene alla quarta generazione di reti mobili cellulari. Successore dell'UMTS, è stata standardizzata dal 3GPP a partire dalle Release 8 e 9, ed è stata aggiornata in LTE-Advanced (Rel. 10 - 12) e LTE-Advanced Pro (Rel. 13 e 14). Si compone di una parte RAN (Radio Access Network), chiamata Evolved UMTS Terrestrial Radio Access Network (E-UTRAN), e di una parte core, chiamate Evolved Packet Core (EPC).

2.3.1 Evolved UTRAN

La rete di accesso LTE E-UTRAN è la combinazione di E-UTRA¹, ovvero l'interfaccia radio LTE, l'User Equipment (UE) e la stazione base Evolved Node B (eNB). Offre le caratteristiche seguenti:

- velocità di download fino a 300 Mb/s in configurazione Multiple-Input Multiple-Output (MIMO) 4x4 e 150 Mb/s in 2x2, utilizzando una banda di 20 MHz. LTE-A supporta anche antenne in configurazione 8x8 per una velocità di picco di 3 Gb/s su banda aggregata di 100 MHz;
- velocità di upload di 75 Mb/s su un canale a 20 MHz, fino a 1,5 Gb/s utilizzando 100 MHz di banda aggregata;
- tempi di latenza ridotti, al di sotto dei 5 ms in condizioni ottimali, grazie all'integrazione del controller radio (RNC) all'interno dell'eNB stesso;
- Velocità dei terminali fino a 350 km/h;
- supporto di Frequency Division Duplex (FDD) e Time Division Duplex (TDD);
- occupazione di banda flessibile (1.4 MHz, 3 MHz, 5 MHz, 10 MHz, 15 MHz e 20 MHz), sulle stesse frequenze utilizzate già dall'International Mobile Telecommunications;
- dimensione delle celle variabile da una decina di metri di raggio (femtocelle e picocelle) fino a oltre 100 km di raggio (macrocelle).

Per quanto riguarda l'infrastruttura di rete, la UTRAN è ora costituita solo dal eNodeB, dal momento che non viene più utilizzato un RNC esterno. Questo semplifica l'architettura e migliora la latenza. Gli eNodeB sono collegati tra di loro tramite interfaccia X2 e alla rete core attraverso l'interfaccia S1. Lo pila protocollare del E-UTRAN è composta da:

- IP;
- Non-Access Stratum (NAS): serve a trasportare dati di segnalazione (non radio) tra User Equipment e Mobility Management Entity, componente dell'EPC, per garantire il corretto funzionamento dell'accesso radio LTE. Serve a gestire le sessioni e la mobilità (handover) degli UE;

¹Evolved Universal Mobile Telecommunications System (UMTS) Terrestrial Radio Access

- Radio Resource Control (RRC): serve ad inviare in broadcast messaggi relativi all'Access Stratum e trasporta messaggi del NAS;
- Packet Data Convergence Protocol (PDCP): riceve i messaggi provenienti dal RRC (NAS) e dal livello IP. Provvede a garantire sicurezza, integrità ed eventualmente compressione dei messaggi;
- Radio Link Control (RLC): trasporta i pacchetti del PDCP. A seconda della modalità di lavoro, può offrire correzione di errori tramite ARQ, segmentazione e concatenazione di PDU, riordinamento e rilevamento di duplicati;
- MAC: offre un insieme di canali logici con cui moltiplicare i messaggi RLC a livello fisico. Gestisce la correzione di errori tramite HARQ, gestisce la priorità dei canali logici per lo stesso UE e lo scheduling dinamico tra UE diversi;
- Livello fisico: trasferisce le informazioni in arrivo dal livello MAC sull'interfaccia radio. Si occupa di selezionare la codifica, modulazione e potenza in funzione delle condizioni del canale radio. Ricerca la cella a cui connettersi durante la sincronizzazione iniziale o in fase di mobilità tra celle diverse.

Al livello fisico, viene utilizzato l'Orthogonal Frequency-Division Multiplexing (OFDM) in downlink e Single-Carrier Frequency-Division Multiple Access (SC-FDMA) in uplink. A seconda della categoria del dispositivo, sono disponibili anche beamforming in downlink e Single User Multiple-Input Multiple-Output in uplink (SU-MIMO). LTE supporta FDD e TDD. Il Frequency-Division Duplex, comune nelle reti europee, prevede la separazione delle frequenze dedicate al canale di uplink e downlink; il Time-Division Duplex invece divide una singola frequenza in periodi di tempo alternati dedicati alla trasmissione e alla ricezione. Ogni modalità ha i propri frame, ed il supporto per entrambe consente di fare economia di scala dato che è possibile usare hardware simile per terminali e stazioni base.

Downlink e uplink hanno ciascuno i propri canali fisici e segnali. I canali per il downlink sono:

- Physical Downlink Control Channel (PDCCH), trasporta informazioni sull'allocazione del canale downlink e sulle risorse uplink riservate allo UE;
- Physical Control Format Indicator Channel (PCFICH), usato per inviare il Control Format Indicator (CFI), che serve a trovare la trasmissione PDCCH all'interno del subframe;
- Physical Hybrid ARQ Indicator Channel (PHICH), porta le conferme di ricezione (ACK) relative al canale uplink;
- Physical Downlink Shared Channel (PDSCH), è il canale principale, usato per la trasmissione di dati L1 in downlink;
- Physical Multicast Channel (PMCH) è utilizzato per inviare in broadcast usando una singola frequenza, anche su più celle;
- Physical Broadcast Channel (PBCH) trasmette in broadcast le informazioni base di sistema per la cella, fondamentali per l'aggancio iniziale dei terminali.

I segnali downlink sono:

- segnali di sincronizzazione, che consentono al UE di trovare la cella e di fare la sincronizzazione iniziale;
- segnali di riferimento, usati dal UE per stimare il canale di DL;
- segnali di posizionamento, aggiunti nella Release 9, che consentono la localizzazione dei terminali mediante Observed Time Difference Of Arrival (OTDOA).

L'uplink fa invece uso di tre canali:

- Physical Random Access Channel (PRACH), usato per l'accesso iniziale o quando viene persa la sincronizzazione dal UE;
- Physical Uplink Shared Channel (PUSCH), canale principale, trasmette i dati L1 in uplink;
- Physical Uplink Control Channel (PUCCH), trasporta informazioni di controllo; poiché la codifica di uplink e i parametri di allocazione del canale sono decisi dal eNB, le uniche informazioni di controllo sono gli ACK e i report Channel Quality Indicator.

I segnali usati in uplink sono di due tipi:

- segnali di riferimento (RS), usati dall'eNB per stimare il canale di UL;
- segnali di riferimento sonda (SRS), usati dall'eNB per conoscere le condizioni del canale di UL al fine di decidere la migliore politica di scheduling.

2.3.2 Evolved Packet Core

L'Evolved Packet Core, chiamato anche System Architecture Evolution (SAE) Core, è una evoluzione della GPRS core network [23]. Svolge le seguenti funzioni:

- Controllo di accesso alla rete: quando un utente richiede l'accesso alla rete, viene verificato che sia autenticato e autorizzato. Se sono disponibili le risorse che richiede, queste gli vengono riservate. Se richiesto dalle autorità, possono essere rese disponibili informazioni sul traffico in transito.
- Routing e trasferimento dei pacchetti: i pacchetti vengono instradati all'interno della rete o tra reti diverse. Poiché la rete è basata su IP, sono usati i relativi meccanismi di routing e forwarding. La funzione offre anche compressione dell'header IP per ottimizzare l'uso della capacità radio e lo screening dei pacchetti al fine di controllare che lo UE stia usando l'indirizzo a lui assegnato.
- Gestione della mobilità: serve a tracciare la posizione dello UE al fine di garantire che l'user plane rimanga attivo dopo i trasferimenti tra celle e tra tecnologie radio diverse.
- Sicurezza: garantisce confidenzialità mediante cifratura e integrità dei dati dell'utente lungo il trasferimento nel canale radio.

- Gestione di rete: si occupa di mantenere la rete in uno stato ottimale, svolgendo load balancing del carico degli UE su MME. Se necessario, viene impedita la registrazione di nuovi UE sulla rete.

Ciascuna funzione è svolta da uno o più dei componenti del EPC, schematizzati in figura 2.1. I componenti principali sono qui descritti.

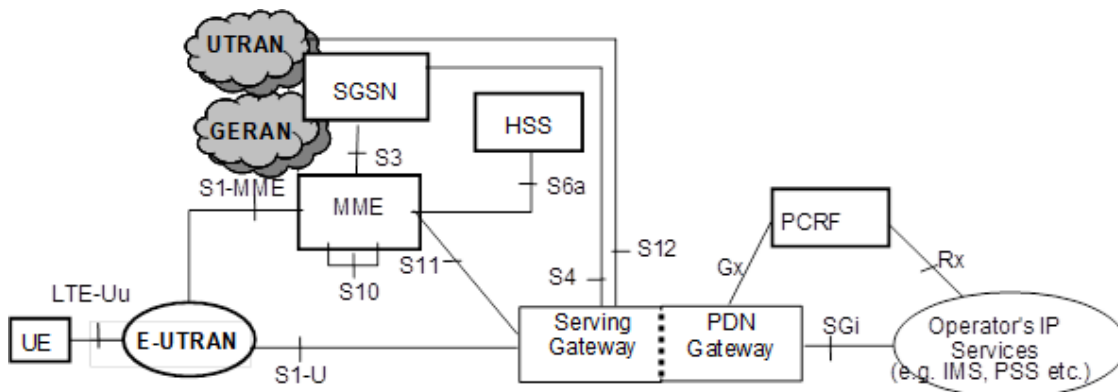


Figura 2.1: Architettura della rete di accesso LTE, in configurazione a singolo gateway

Mobility Management Entity (MME) è il componente chiave per la rete di accesso LTE. Si occupa di recuperare dal HSS i dati relativi allo UE connesso e di scegliere per lui le risorse radio ed il gateway appropriati. Tutti i segnali NAS vengono terminati al MME. Offre la funzione di control plane per mobilità tra LTE e tecnologie precedenti interfacciandosi con il SGSN.

Serving Gateway (SGW) è il gateway che termina l'interfaccia verso l'E-UTRAN. Fa routing e forwarding dei pacchetti utente. Funge da mobility anchor durante l'handover tra eNodeB o tra diverse tecnologie di accesso radio 3GPP. Se lo UE è in idle, interrompe il flusso dati di downlink e attiva il paging dello UE.

Packet Data Network Gateway (PGW) fornisce accesso alle reti IP esterne attraverso interfaccia SGI, assegnando un indirizzo il allo UE. Uno UE può essere connesso a più PGW per poter accedere a più Packet Data Network (PDN). Nel caso di mobilità tra tecnologia di accesso radio 3GPP e non, funge da mobility anchor.

Home Subscriber Server (HSS) è il database centrale che contiene le informazione sull'utente e la sua sottoscrizione, effettua autenticazione e autorizzazione, può fornire informazioni sulla posizione e IP degli UE.

Serving GPRS Support Node (SGSN) supporta tecnologie di accesso 3GPP di precedenti generazioni.

Policy and Charging Rule Function (PCRF) si occupa di applicare policy in real time alle connessioni degli UE, che possono essere livelli di QoS e più in generale servizi differenziati. Inoltre si occupa del conteggio di metriche di interesse al fine di poter addebitare i servizi agli utenti.

2.4 Rete mobile 5G

Nel settembre 2015, l'International Telecommunication Union - Radiocommunications Standardization Sector (ITU-R) ha pubblicato un documento intitolato “IMT² Vision - Framework and overall objectives of the future development of IMT for 2020 and beyond” (ITU-R M.2083-0), con cui ha espresso le raccomandazioni per la nuova tecnologia di rete 5G [1]. Un sistema 5G in linea con le direttive IMT-2020 dovrà soddisfare le seguenti specifiche:

- velocità di trasferimento di picco di 20 Gb/s;
- velocità di trasferimento per singolo utente nell'area di copertura di 100 Mb/s;
- tempo di latenza di trasmissione di 1 ms;
- velocità massima consentita ai dispositivi mobili di 500 km/h;
- densità di dispositivi connessi di 10^6 dispositivi/km²;
- efficienza energetica superiore a quella del 4G di un fattore 100;
- efficienza spettrale superiore a quella del 4G di un fattore 3;
- capacità di traffico per area di 10 Mb/(s · m²).

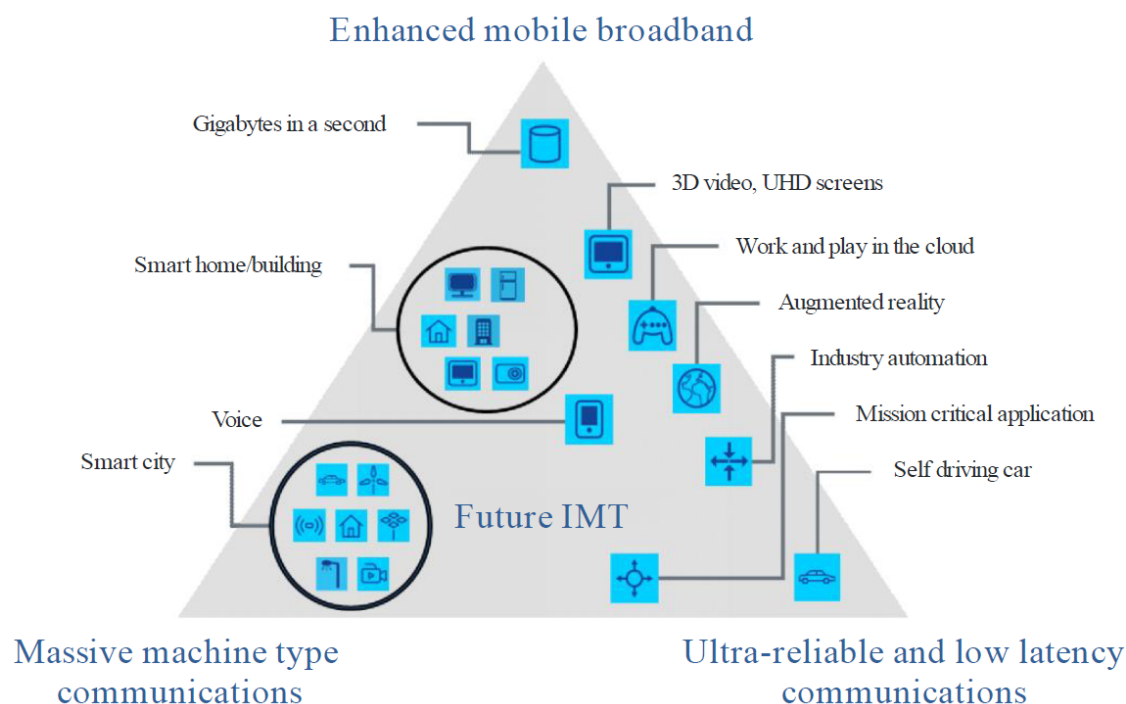
Questi requisiti sono pensati in funzione di tre scenari d'uso principali individuati dall'ITU-R:

Enhanced Mobile Broadband (eMBB) La banda larga mobile fa parte dei casi d'uso relativi alla fruizione di contenuti multimediali, dati e servizi. L'aumento della richiesta di banda rispetto ai livelli attuali necessita la definizione di un caso d'uso ulteriore, l'eMBB appunto, che tenga conto dei nuovi scenari e dei nuovi requisiti per prestazioni superiori.

Ultra-reliable and low latency communications (uRLLC) I requisiti più stringenti di banda, latenza e disponibilità caratterizzano un caso d'uso che comprende processi di produzione industriale, operazioni chirurgiche da remoto, sicurezza dei trasporti, e più in generale tutte le applicazioni mission critical che prevedono uno scambio di dati robusto e privo di interruzioni. Per il sistema ICA oggetto di questa tesi, vedremo nei capitoli successivi che questo è un punto di svolta per ottenere prestazioni superiori rispetto a quelle ottenute in laboratorio con la rete radio di quarta generazione.

Massive machine type communications (mMTC) Quest'ultimo caso d'uso è caratterizzato da un numero molto elevato di dispositivi connessi che trasmettono una quantità relativamente limitata di dati, senza requisiti di latenza massima. Tali dispositivi costano poco e hanno una batteria a lunga durata; i sensori per l'Internet of Things rientrano in questa descrizione.

²International Mobile Telecommunications



M.2083-02

Figura 2.2: Casi d'uso presentati in IMT Vision[1]

Il processo di standardizzazione è previsto che si concluda nel 2020, quando la bozza della nuova raccomandazione ITU-R contenente le specifiche della nuova tecnologia radio sarà sottoposta ad approvazione.

Venendo incontro ai requisiti imposti dall'ITU, il 3rd Generation Partnership Project (3GPP), la stessa organizzazione che ha sviluppato e mantiene tutte le tecnologie di rete cellulare a partire dal GSM (2G), ha iniziato a lavorare al 5G System (5GS) con l'obiettivo di sottoporlo come standard IMT-2020. Il 5GS comprende sia una nuova Next Generation Radio Access Network (NG-RAN), che usa la nuova interfaccia radio 5G New Radio (NR), sia una nuova core network (5GC).

Le specifiche del 5GS si articolano in due parti: la Phase 1, definita all'interno del 3GPP Release 15, e la Phase 2, che sarà definita nella Release 16, prevista per il 2020. A causa delle pressioni degli operatori di telecomunicazioni, la Release 15 è stata a sua volta suddivisa in più parti per dare priorità alle funzioni maggiormente richieste.

- **"Early Rel-15 drop"** (marzo 2018): si concentra sul architecture option 3, nota come non-standalone NR (NSA NR), che è considerata come il primo passo per la migrazione di una rete 4G verso il 5G. Prevede l'affiancamento di una stazione base NR, chiamata gNB, alla stazione base LTE (evolved Node B, eNB) e alla relativa core network (Evolved Packet Core, EPC).
- **"Regular Rel-15 drop"** (settembre 2018): il protagonista è la architecture option 2, ovvero la standalone NR, che consiste in una rete di stazioni base NR connessa

ad una core network 5G, senza il coinvolgimento del LTE. Viene specificata anche l'architecture option 5: connessione di una stazione base LTE alla rete core 5G.

- **"Late Rel-15 drop"** (giugno 2019): vengono finalizzate le architetture rimanenti, ovvero option architecture 4 e 7. Prevedono l'aggiunta di una stazione base LTE a una rete Standalone New Radio in cui il control plane è gestito dalla stazione base NR, per l'opzione 4, o LTE, per l'opzione 7.

2.4.1 5G New Radio

Il cuore del NG-RAN è costituito da una nuova interfaccia radio: 5G New Radio. Nella Phase 1 del 5GS sono state definite le caratteristiche chiave del 5G-NR.

- Supporto per un maggior numero di configurazioni radio, che consentono una scelta più ampia di bitrate e latenza, così come maggiori possibilità di copertura e scalabilità in funzione della banda richiesta. Mentre LTE supporta bande fino a 20 MHz con spaziatura tra sottoportanti invariabile di 15 kHz, NR fa un passo avanti e lascia la libertà di aumentare lo spacing tra sottoportanti in funzione della larghezza di banda desiderata. Una delle novità di NR è infatti il supporto di frequenze millimetriche, o mmWave, al di sopra dei 6GHz, con canali con banda di centinaia di MHz, che richiedono spaziatura maggiore. La maggiore larghezza di banda disponibile a queste frequenze consentirebbe di raggiungere velocità impressionanti, nell'ordine di 10-100 Gb/s. Di contro, la maggior frequenza è accompagnata da una minore penetrazione a causa delle perdite superiori al crescere della distanza di trasmissione. Questo rende il NR su onde millimetriche ideale per spazi chiusi o per piccoli spazi aperti, dove sarebbe in grado di offrire prestazioni estremamente elevate ad un grande numero di dispositivi, mentre per spazi più ampi, per garantire una copertura uniforme, possono essere usate frequenze inferiori.
- Transmission Time Interval (TTI) variabile, rispetto al TTI di 1 ms fisso di LTE, che consente di scendere ad un valore sino a 143 μ s. La struttura del frame flessibile del 5G NR è pensata per consentire di aumentare e diminuire il TTI a seconda delle necessità di servizio, dando la possibilità quando necessario di migliorare la latenza a scapito dell'efficienza spettrale.
- Introduzione della struttura dei frame di tipo self-contained, ovvero un nuovo tipo di subframe che riesce a contenere i dati ed il loro ACK, inserito dalla stazione ricevente, la quale è in grado di decodificare i dati e di fornire un feedback all'interno dello stesso subframe. Il vantaggio è una diminuzione della latenza e una maggiore efficienza della rete.
- Tecnologia Multiple Input Multiple Output massiva e beamforming, con l'obiettivo di migliorare le prestazioni radio in ambienti rumorosi o a lunga distanza. L'aumento della frequenza e il conseguente degrado di segnale sulle lunghe distanze o a causa della presenza di edifici rende necessario l'uso del beamforming per mantenere un livello di copertura simile a quello di LTE. Grazie alla lunghezza d'onda inferiore è possibile mantenere le dimensioni dell'array di antenne contenute, anche se formate da un grande numero di elementi, pertanto il dispiegamento di tale soluzione è estremamente

conveniente. Grazie a tali tecniche si ottiene un data rate più stabile e una migliore Quality of Service.

- Nuova codifica Low Density Parity Check (LDPC), usata per i canali dati, e la codifica polare, per i canali di controllo, migliorano il tasso di errore consentendo così maggiore qualità del servizio, copertura e prestazioni alle estremità della cella radio, dove il segnale è generalmente peggiore.
- Coesistenza con reti LTE, che consente di fornire servizio ai terminali in caso di mancanza di copertura 5G. Inoltre, specialmente durante il primo periodo di attivazione, la rete 5G sarà complementare a quella 4G, che continuerà ad offrire copertura macro mentre il 5G sarà usato per offrire incrementi di prestazioni locali nelle aree più frequentate.

2.4.2 Multi-access Edge Computing

Una delle tecnologie che consentirà alle reti di nuova generazione di rispettare gli esigenti requisiti in termini di KPI, in particolare quelli sulla latenza, è senza dubbio il Multi-access Edge Computing [2]. Gli sforzi dell'ETSI per lo studio e la standardizzazione di questa tecnologia sono iniziati nel 2014, quando è stato istituito il relativo Industry Specification Group (ISG). Inizialmente l'acronimo MEC stava a significare Mobile Edge Computing, tuttavia in seguito, per sottolineare come la tecnologia non fosse appannaggio solo delle reti mobili ma consentisse una qualunque soluzione di accesso, si è scelto di rinominarla in Multi-access Edge Computing.

Il concetto alla base del MEC, come si evince dal nome, è l'Edge Computing. Si tratta di un paradigma di calcolo distribuito che consiste nello spostare nella topologia di rete le risorse di calcolo da posizioni centralizzate a nodi periferici, dove si possano elaborare i dati più vicino agli utenti che li creano e consumano. Il vantaggio principale ed immediatamente tangibile è il miglioramento della latenza di trasmissione, dal momento che i pacchetti non devono transitare per la rete di trasporto prima di giungere a destinazione. Questo fa sì che le stesse reti di trasporto vedano diminuire la capacità a loro richiesta dal momento che una quantità minore di dati è necessario che le attraversi. Inoltre, poiché per implementare il MEC è possibile utilizzare l'infrastruttura virtualizzata già impiegata dagli operatori di telecomunicazioni per ospitare funzioni di rete virtualizzate (NFV), si ha un vantaggio anche dal punto di vista dei costi.

Dal punto di vista architetturale, le entità che compongono il MEC possono essere classificate in tre gruppi: di livello sistema, livello host o di livello rete. Tenendo presente la Figura 2.3, vengono ora presentate le componenti principali.

MEC host è il componente che contiene la MEC platform e l'infrastruttura di virtualizzazione che provvede a fornire potenza di calcolo, storage e risorse di rete alle applicazioni MEC. L'infrastruttura di virtualizzazione comprende il data plane impostato tramite regole di traffico impartite dal MEC platform. Instrada i pacchetti tra le applicazioni, i servizi, i server DNS, la rete di accesso, le reti locali e quelle esterne.

MEC platform si comporta come un interfaccia tra le applicazioni MEC e l'esterno e tra le applicazioni stesse. Si occupa delle seguenti funzioni:

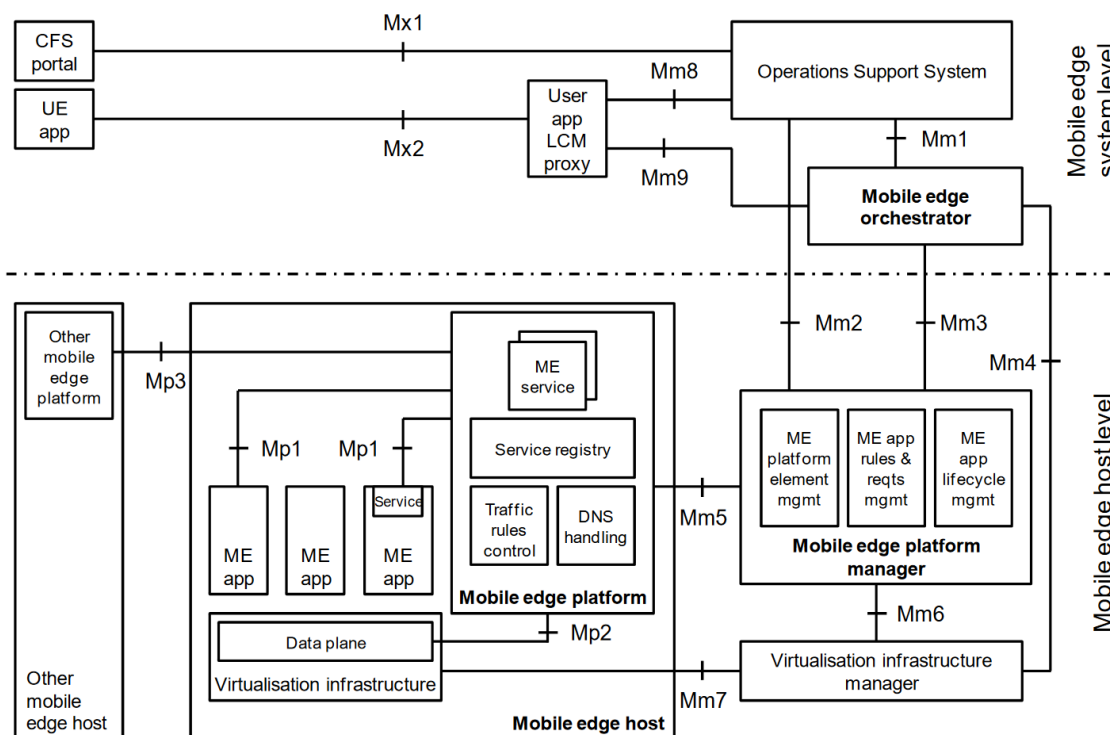


Figura 2.3: Architettura di riferimento del sistema MEC

- offre un ambiente in cui le applicazioni MEC possano scoprire, consumare ed eventualmente offrire servizi MEC alle altre applicazioni. Se supportato, i servizi MEC possono essere disponibili tramite altre piattaforme appartenenti anche a sistemi MEC diversi;
- riceve le regole di traffico dal MEC platform manager, dalle applicazioni o dai servizi e instruisce il data plane di conseguenza. Tali regole possono ad esempio consentire di redirezionare il traffico di uno UE verso una MEC app specifica;
- gestisce il server DNS utilizzato dalle applicazioni MEC, i cui record sono configurati dal MEC platform manager;
- ospita i MEC services, che possono essere ad esempio il Radio Network Information, per esporre informazioni relative alla connessione radio, o il servizio Location, che offre informazioni riguardo la posizione degli UE connessi;
- offre alle applicazioni la possibilità di salvare dati in modo persistente attraverso dello spazio di storage dedicato.

MEC applications sono eseguite su macchine virtuali dedicate all'interno dell'infrastruttura di virtualizzazione fornita dal MEC host e possono interagire con la piattaforma MEC per offrire e consumare servizi MEC. Possono interagire con la piattaforma MEC anche per procedure di supporto legate al ciclo di vita dell'applicazione, come ad esempio per indicare disponibilità. Alle applicazioni MEC possono essere associate

regole e requisiti che vengono validati dal MEC system level management e possono comprendere vincoli come massima latenza, servizi o risorse richiesti.

MEC service è un servizio che viene offerto e consumato dalla piattaforma o da un'applicazione MEC. Se offerto da un'applicazione, viene registrato sulla MEC platform tramite il punto di riferimento Mp1, che allo stesso modo può essere usato dall'applicazione per sottoscrivere un servizio tra quelli disponibili. Al fine di rispettare i requisiti imposti dallo standard, sono richiesti almeno i servizi seguenti:

- **Radio Network Information:** offre alle applicazioni autorizzate informazioni riguardo alla rete radio, misure e statistiche dello user plane e informazioni sugli UE associati con il MEC host;
- **Location:** rende accessibili informazioni riguardo la posizione degli UE collegati al MEC host, come geolocalizzazione e ID della cella radio;
- **Bandwidth Manager:** permette alle MEC applications di riservare capacità di traffico per i propri dati e di definire regole di priorità.

Multi-access edge orchestrator è la funzione principale del livello sistema del sistema MEC. Si occupa di:

- mantenere una visione di insieme del sistema MEC in base agli MEC host attivi, risorse e servizi MEC disponibili e sulla topologia;
- fare on-boarding di pacchetti applicativi, controllandone integrità e autenticazione, validando regole applicative e requisiti in base alle regole dell'operatore. Mantiene traccia dei pacchetti caricati e prepara il gestore dell'infrastruttura di rete per l'avvio delle applicazioni;
- selezionare il MEC host ottimale per l'instanziazione delle applicazioni basandosi sui requisiti come latenza, disponibilità di risorse e servizi;
- avviare e terminare le applicazioni MEC;
- fa partire la ricollocazione delle applicazioni quando necessario.

Operation Support System riceve richieste attraverso il portale CFS e attraverso le applicazioni dei dispositivi connessi per l'instanziazione e la terminazione delle applicazioni MEC, e sceglie se consentirle. Le richieste approvate sono inoltrate al Multi-access edge orchestrator. L'OSS può anche ricevere richieste dalle applicazioni dai terminali connessi per la ricollocazione delle applicazioni da servizi cloud esterni al sistema MEC e viceversa. Le richieste provenienti da dispositivi collegati sono inoltrate attraverso l'User application lifecycle management proxy che fa da tramite tra OSS e dispositivi.

Customer facing service (CFS) portal consente a clienti di terze parti di accedere ai servizi offerti dal sistema MEC dell'operatore di rete, come ad esempio l'avvio di applicazioni MEC proprietarie o la ricezione di informazioni da parte delle applicazioni stesse già in esecuzione.

MEC platform manager fa da intermediario tra il MEC orchestrator e il MEC host. Informa il MEC orchestrator di eventi collegati al ciclo di vita delle applicazioni MEC, gestisce le regole applicative e i requisiti come l'autorizzazione di servizi, regole di traffico e la configurazione del DNS.

Virtualization infrastructure manager si occupa di funzioni di basso livello legate all'infrastruttura di virtualizzazione. Alloca, gestisce e rilascia le risorse virtualizzate dell'infrastruttura come processori virtuali, memoria RAM, supporti di memorizzazione e risorse di rete, memorizza e all'occorrenza carica le immagini dei dischi delle VM e si occupa della rilocalizzazione delle macchine virtuali da o verso i provider cloud.

Il collegamento tra i componenti del sistema MEC avviene attraverso i cosiddetti reference points. Ci sono tre tipi di punti:

- reference points che riguardano le funzionalità della MEC platform (Mp);
- reference points di amministrazione (Mm);
- reference points per il collegamento ad entità esterne (Mx).

Per concludere, è importante sottolineare che l'architettura del sistema MEC è stata pensata perché fosse possibile installarla in molti modi diversi. Forse quello più interessante per un uso pratico è l'installazione del sistema all'interno di un ambiente NFV, allo scopo di riutilizzare l'infrastruttura di virtualizzazione di cui gli operatori di telecomunicazioni sono già dotati. Dal momento che il testbed 5G non fa uso di questa configurazione, non si ritiene opportuno descrivere la relativa architettura.

2.4.3 Network Slicing

Il Network Slicing è forse la funzione che, tra le novità standardizzate nel 5G, più consentirà di adattare la rete di nuova generazione alle esigenze di tutti.

In origine e per lungo tempo le reti mobili sono state pensate, progettate e realizzate per fornire un servizio ai clienti che necessitavano di una capacità sostenuta di trasferimento di dati in mobilità, consentendo, per impieghi con requisiti più specifici, la sola regolazione di parametri come la priorità e la qualità del servizio (QoS). Quindi gli operatori di telefonia hanno spesso trovato difficoltà ad adattare le proprie reti per venire incontro alle richieste peculiari di servizi verticali.

Di recente, la virtualizzazione delle funzioni di rete (NFV) ha offerto un modo per semplificare la messa in campo di questo tipo di soluzioni disaccoppiando il software dall'hardware dedicato ai componenti di rete.

Il Network Slicing è una ulteriore spinta in questa direzione. Consente di separare logicamente l'insieme delle funzioni e delle risorse all'interno di una stessa rete, realizzando reti indipendenti end-to-end, dette "slice", su una infrastruttura fisica condivisa, capaci di fornire una qualità del servizio prestabilita. Una slice può comprendere sia risorse dedicate che condivise, come potenza di calcolo, storage e traffico pur rimanendo isolata dalle altre slice attive. La qualità del servizio è concordata tra operatore mobile e l'azienda cliente attraverso dei Service Level Agreement (SLA), che possono comprendere requisiti

di velocità di trasferimento, affidabilità, latenza, sicurezza e servizi aggiunti [24]. Un operatore mobile può fornire una singola slice che soddisfi le esigenze di verticals multipli oppure può vendere un pacchetto costituito da slice di diverso tipo ad un singolo cliente. Un esempio può essere una casa automobilistica che per la propria auto richiede un pacchetto che comprenda una slice ad alta affidabilità e bassa latenza per l'assistenza alla guida e una slice ad ampia capacità di traffico per l'infotainment.

Il concetto non è completamente nuovo: per certi versi può essere considerato simile a tecniche già esistenti come le VLAN 802.1q, che analogamente permettono di separare i domini di broadcast di reti che utilizzano la stessa infrastruttura. A differenza delle VLAN però si aggiunge un nuovo grado di complessità, dal momento che una Network Slice può estendersi su più parti della rete, ad esempio su reti di accesso mobili, reti core e di trasporto, o addirittura su reti di operatori diversi. Ciò che rende possibile il concetto di Network Slicing sono tecnologie come Network Functions Virtualization (NFV) e Software Defined Networking (SDN) che permettono la flessibilità necessaria a realizzare una soluzione di questo tipo.

2.4.4 Progetto H2020 5G-Transformer

Il 5G-Transformer è un progetto di ricerca europeo che ricade in Horizon 2020, il Programma Quadro dell'Unione Europea per la ricerca e l'innovazione per il periodo 2014-2020. I Programmi Quadro hanno durata sette anni e sono il principale strumento con cui l'UE finanzia la ricerca in Europa.

L'obiettivo del progetto 5G-Transformer è di creare per le compagnie operanti in mercati verticali un ecosistema di innovazione tecnologica e di impresa che serva una moltitudine di applicazioni, ciascuna caratterizzata da specifici requisiti come alta affidabilità, grande velocità di trasmissione di dati e/o bassa latenza. Il progetto studia come due delle novità di maggior rilievo del 5G, ovvero l'architettura MEC e il Network Slicing, possano supportare i partner vertical offrendo l'astrazione, la flessibilità e il dinamismo che richiedono alla rete mobile. I domini verticali che sono considerati nel progetto sono: automotive, spettacolo, e-health, industria 4.0 e operatori di rete mobile virtuali [7].

L'elemento chiave dell'architettura di rete che si sta costruendo in 5G-Transformer è la Mobile Transport and Computing Platform (MTP), che andrà a sostituire la attuale rigida rete di trasporto dedicata alle reti mobili. Le industrie verticali potranno rispettare i requisiti dei loro servizi tramite delle slice MTP su misura, basate su una rete di trasporto e una piattaforma di calcolo distribuito implementate su una infrastruttura di virtualizzazione aggregata e federata. I provider possono quindi condividere la rete di trasporto mobile 5G e l'infrastruttura di calcolo, tra di loro ed i vertical, tramite API aperte, in modo da aumentare l'efficienza della rete e diminuire i costi.

Dal punto di vista architetturale sono stati pensati tre componenti: il vertical slicer e il service orchestrator e il già menzionato 5GT-MTP [8]:

Vertical Slicer (5GT-VS) il 5GT-VS è il punto di ingresso per le aziende vertical che vogliono richiedere un servizio basato sulla piattaforma 5GT. I vertical definiscono i requisiti dei servizi che richiedono tramite dei Vertical Service Descriptor (VSD), dopodiché il compito del 5GT-VS è di mappare la descrizione di questi servizi con i

parametri di istanziazione del servizio di rete virtualizzato (NFV-NS) che implementi la slice di rete secondo i requisiti del vertical. Il VSD viene quindi tradotto in un Network Service Descriptor (NSD), un grafo che contiene funzioni di rete virtualizzate collegate tra loro, che viene poi inviato al 5GT-SO per la creazione del servizio completo NFV-NS.

Service Orchestrator (5GT-SO) il principale compito del 5GT-SO è di fornire una orchestrazione end-to-end del NFV-VS ricevuto dal 5GT-VS attraverso domini amministrativi multipli, interagendo con il 5G-MTP locale e con i 5GT-SO degli altri domini collegati (con cui costituisce una federazione). Il 5G-SO rende trasparente al 5G-VS la gestione dei servizi ospitati nella federazione: anche se vengono utilizzati più domini amministrativi, il 5G-VS si interfaccia sempre con un solo 5G-SO.

Mobile Transport and Computing Platform (5GT-MTP) il 5GT-MTP è responsabile della orchestrazione di risorse e l'istanziazione di funzioni virtualizzate nell'infrastruttura che sta direttamente sotto il suo controllo. Gestisce le risorse virtuali in suo possesso su cui vengono eseguiti i servizi virtualizzati, seguendo le indicazioni del 5GT-SO. Un 5GT-MTP può controllare e coordinare più domini tecnologici, come datacenter, reti mobili o wide area network, e si presta ad essere utilizzato come gestore di infrastrutture sia di cloud che edge computing (ad esempio MEC).

Ciascuno di questi tre componenti include un servizio di monitoraggio in grado di fornire metriche sulle prestazioni e report sui guasti delle entità logiche gestite. I dati raccolti possono essere utilizzati internamente per prendere decisioni all'interno del singolo componente o possono essere forniti a componenti esterni. Ad esempio, il 5GT-SO può prendere una decisione di scaling per un servizio NFV in base alle metriche fornite dal servizio stesso, quando rileva che una determinata soglia viene raggiunta.

Capitolo 3

Progettazione e sviluppo testbed

Presso il Dipartimento di Elettronica e Telecomunicazioni del Politecnico di Torino è installato il testbed 5G automotive. Si tratta di una piattaforma di test hardware-in-the-loop con cui si sono svolti tutti i test che saranno riportati nel capitolo successivo. Per la descrizione del testbed si farà riferimento al paper "A MEC-based Extended Virtual Sensing for Automotive Services" pubblicato per il convegno AEIT Automotive 2019 [25].

Il cuore del testbed è costituito dall'applicazione Extended Virtual Sensing, che si occupa di prevenire incidenti tra veicoli nelle intersezioni stradali tramite la previsione di collisioni e l'avviso dei conducenti. I dati raccolti dai veicoli costituiscono il punto di partenza dell'applicazione. I veicoli possono contare su più modi per raccogliere dati che riguardano loro e l'ambiente in cui si spostano. In primo luogo, le numerose centraline presenti nel veicolo offrono una visione completa sullo stato e la dinamica del veicolo, essendo a conoscenza, tra i molti dati raccolti, anche di posizione, velocità, accelerazione e direzione. I dati relativi al veicolo sono poi integrati con quanto viene rilevato dai sensori esterni. Accanto ai sensori tradizionali, come quelli per luce o pioggia, ora è comune trovare anche sensori dedicati alla guida semiautonomo (ADAS) che arricchiscono i dati a disposizione dei singoli veicoli che sono così in grado di percepire l'ambiente circostante. Infine, se l'automobile è connessa con un sistema di comunicazione V2I, dispone di informazioni che vanno oltre il raggio di azione dei sensori (es. onda verde), che vanno ad aggiungersi a quelle già in suo possesso.

Vehicle Simulator Nel testbed, il veicolo è simulato mediante il software Vehicle Simulator, che si occupa sia di inviare i CAM che di ricevere i DENM. Il Vehicle Simulator ha accesso a tutte le informazioni che riguardano la dinamica e lo stato dei veicoli che simula, si occupa di codificarle ed inviarle in modo temporizzato, attraverso l'interfaccia radio, al CIM.

OpenAirInterface (OAI) È l'interfaccia radio utilizzata per inviare e ricevere i dati dai veicoli. Si tratta a tutti gli effetti di un collegamento mobile LTE realizzato in laboratorio. Integra hardware e software open source per realizzare l'accesso radio (RAN) e per la rete core (EPC).

Cooperative Information Manager (CIM) È il software che si occupa di ricevere e memorizzare i dati che vengono trasmessi dai veicoli all'interno della sua area di copertura, per renderli disponibili ad applicazioni in grado di elaborarli per svolgere un servizio di pubblica utilità come ad esempio rilevare collisioni.

Extended Virtual Sensing (EVS) Uno o più EVS accedono alle informazioni posizione, direzione, velocità e accelerazione raccolti dal CIM per una determinata area di interesse, come può essere un insieme di intersezioni stradali, e, tramite un algoritmo di rilevamento di collisioni basato sul calcolo di traiettorie, determinano la possibile collisione.

DENM Decider (DD) Se una collisione viene rilevata, il DENMDecider si occupa di creare il messaggio DENM da inviare al veicolo per farlo arrestare in sicurezza e prevenire la collisione. Oltre alla codifica del DENM, il DD filtra le stesse collisioni rilevate da diverse istanze di EVS e temporizza l'inoltro delle notifiche ai veicoli.

Sebbene le componenti software siano funzionalmente rimaste le stesse, la composizione del testbed è variata nel tempo, sia a causa della necessità di effettuare prove su hardware alternativo, al fine di discriminare la fonte di problemi incontrati, sia per adeguarsi a configurazioni richieste ai fini dei progetti per cui il testbed è impiegato. È comunque possibile individuare due configurazioni principali:

1. Testbed MEC, che consiste in un MEC Host in cui le applicazioni CIM, EVS e DENMD sono installate come delle MEC App;
2. Testbed Openstack, in cui il software opensource Openstack Devstack è utilizzato come Virtualized Infrastructure Manager per la gestione delle applicazioni in macchine virtuali.

Si procede inizialmente a descrivere le parti comuni alle due implementazioni per poi individuare le differenze in due appositi paragrafi successivi.

3.1 Vehicle Simulator

Il Vehicle Simulator si occupa di simulare il comportamento di veicoli in movimento, di inviare i relativi CAM e ricevere eventuali DENM. I parametri della simulazione sono determinati dalla traccia di simulazione, un file CSV (Comma-Separated Values) in cui per ogni riga sono riportati i valori che il Vehicle Simulator dovrà codificare ed inviare come CAM al tempo specificato. Le tracce di simulazione sono generate mediante SUMO (Simulation of Urban MObility), un software opensource di simulazione di traffico progettato per gestire flussi di traffico attraverso ampie reti stradali. La simulazione è di tipo microscopico: ciascun veicolo è modellato esplicitamente, ha il proprio percorso e si muove individualmente attraverso la rete stradale. La mappa che viene usata per le simulazioni consta di tre strade con una corsia per senso di marcia e due intersezioni non regolate, dove non è consentito svoltare. La strada orizzontale è lunga 1300 m, le verticali 800 m; la strada tra le due intersezioni è lunga 500 m.

I flussi veicolari che vengono usati per la simulazione non sono programmati tramite l'applicazione ma sono generati separatamente tramite uno script Python con cui è possibile

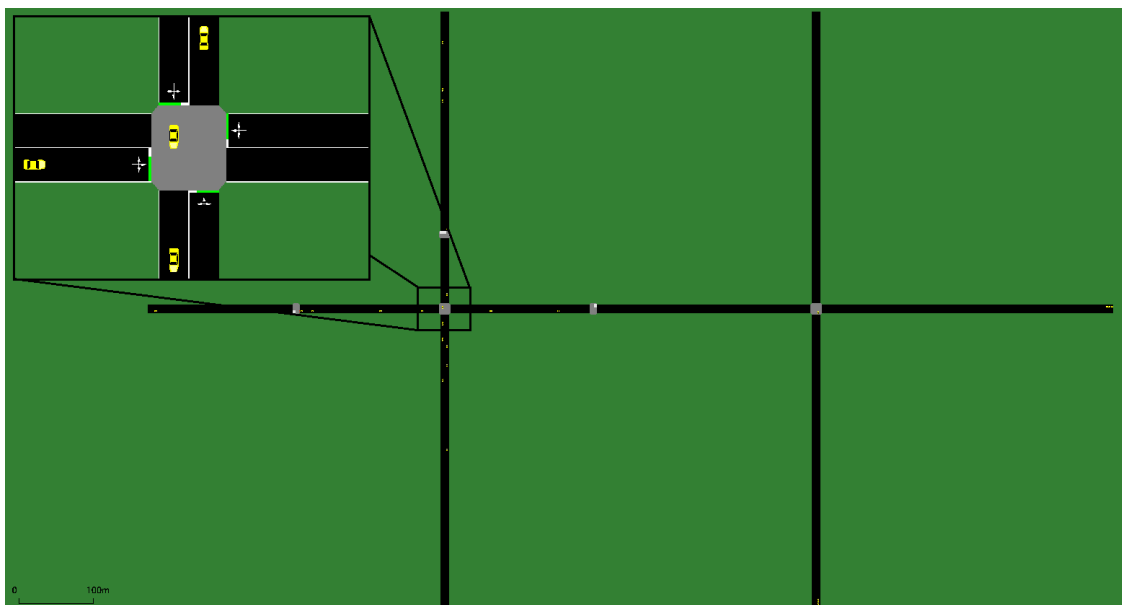


Figura 3.1: Mappa impiegata per la simulazione di SUMO

variare la distribuzione di probabilità di immissione di auto nella mappa; per i test verrà utilizzata una distribuzione di Poisson di intensità variabile. Ciascun veicolo è vincolato alla velocità di 50 km/h ed è impostato per non evitare le collisioni, che possono avvenire solo tra veicoli con direzioni perpendicolari. La simulazione è tempo discreto con step time pari a 0,01 s. Per generare la traccia di simulazione, viene utilizzato il log del programma SUMO, su cui sono riportati i dati dei veicoli per ogni istante di tempo. Poiché desideriamo una frequenza di generazione di CAM costante pari a 10 Hz, viene mantenuto nel file CSV, per ciascun veicolo, un record ogni 10 del log. Durante l'esecuzione, il Vehicle Simulator legge il CSV riga per riga ed esegue le seguenti azioni:

1. legge il tempo in cui deve essere inviato il CAM, ed aspetta fino a che non arriva;
2. realizza la struttura dati del CAM in cui inserisce i dati contenuti nel record letto;
3. codifica la struttura tramite Unaligned Packed Encoding Rules (UPER);
4. trasmette il pacchetto così generato al CIM.

Dal punto di vista hardware, il Vehicle Simulator, a differenza delle altre componenti software precedentemente menzionate, non viene ospitato su macchina virtuale dedicata ma viene eseguito sulla stesso calcolatore incaricato di gestire lo UE. Per questo motivo ci si riferirà al Vehicle Simulator anche col nome UE.

3.2 OpenAirInterface

L'interfaccia radio tra il Vehicle Simulator e il CIM è realizzata tramite OpenAirInterface, un progetto sviluppato e mantenuto dal centro di ricerca francese EURECOM. Si tratta

di una implementazione opensource, conforme allo standard 3GPP LTE Release 10, di UE, eNB, MME, HSS, SGw e PGw. Si rimanda al paragrafo 2.3 per maggiori dettagli su questi componenti. Il software è eseguibile su hardware general purpose x86 e ARM e su apparecchi radio comuni come URSP National Instruments/Ettus e piattaforme PXIe. Lo scopo del progetto è realizzare una implementazione del 5G NR opensource su hardware general purpose, per semplificare l'accesso alla rete, ridurre i costi, aumentare la flessibilità, incrementare la velocità di innovazione e accelerare il time-to-market per l'introduzione di nuovi servizi [26].

La configurazione prevede un eNodeB, collegato via Ethernet a 1Gb/s al EPC, e due UE che si ripartiscono la simulazione dei veicoli tramite due istanze sincronizzate di Vehicle Simulator. Per realizzare gli UE (e quindi i Vehicle Simulator) ed eNB vengono utilizzati notebook con CPU Intel i7-8550U, con Turbo Core, Hyper Threading e C States disabilitati a causa di limiti di OAI riguardo precisione delle frequenze di clock ed vincoli per real time. Questo implica frequenze di clock bloccate a 1,8 GHz su 4 core logici. La RAM ammonta a 16 GB; il sistema operativo è Ubuntu 16.04 LTS. Ai notebook, sono collegate board Ettus USRP B210, che si occupano di inviare e ricevere il segnale LTE. Dal punto di vista radio, è utilizzata la banda 7 a 2,6 GHz in modalità FDD, 10 MHz di larghezza di banda totale ripartita in 5 MHz per direzione per un totale di 25 RB disponibili per link.

I notebook e le board costituiscono la RAN del collegamento LTE tra Vehicle Simulator e CIM; l'EPC è invece virtualizzato insieme agli altri software del testbed in macchina virtuale dedicata. Il HSS è configurato per riconoscere gli UE come terminali validi tramite appositi record nel database MySQL che utilizza come backend, i SGw e PGw sono integrati in un unico processo (S/PGw) che si occupa di assegnare gli indirizzi IP agli UE e di terminare il tunnel GTP verso gli UE stessi. Infatti, a livello data plane, il pacchetto inviato da un Vehicle Simulator al CIM transita prima tramite il S/PGw, che lo decapsula dal tunnel GTP in cui è stato inserito dallo UE e lo inoltra verso la macchina di destinazione. In downlink la situazione è simile: il pacchetto inviato dal DD verso uno dei Vehicle Simulator transita per il S/PGw che lo incapsula in GTP e lo inoltra verso lo UE.

3.3 Cooperative Information Manager

Il CIM è il software che per primo interagisce con i CAM inviati dalle automobili. Alla ricezione di un CAM, esegue le seguenti azioni:

1. decodifica il CAM ricevuto;
2. salva il CAM in un database PostgreSQL, per consentire l'eventuale post-processing sui dati ricevuti;
3. internamente trasferisce una copia del CAM ricevuto a ciascuno dei CAM Manager attivi.

Il CAM Manager è un componente del CIM che mantiene in memoria solo i CAM relativi alla sua area di competenza in attesa che gli vengano chiesti dall'EVS; gli altri li scarta. Più CAM Manager possono essere attivi contemporaneamente, in modo che multiple istanze di EVS possano occuparsi di aree diverse servite da uno stesso CIM. Qualora nessun EVS interroghi un CAM Manager, questo scarta tutti i CAM più vecchi di un secondo in modo

da mantenere solo CAM aggiornati e limitare così il consumo di memoria. Dopo che un EVS chiede al CAM Manager i CAM in suo possesso per una determinata area, i CAM vengono cancellati dalla memoria in modo che per ciascuna query al CAM Manager vengano restituiti solo i CAM che sono arrivati durante il tempo intercorso dalla richiesta precedente. I CAM vengono trasmessi su livello trasporto UDP, pertanto è comune che non tutti i pacchetti inviati dai Vehicle Simulator vengano ricevuti; la comunicazione tra CIM e gli EVS, dopo il setup iniziale su UDP, avviene invece tramite connessione TCP. Un EVS che voglia ricevere i CAM di una determinata area deve compiere i passi seguenti:

1. chiede l'ID di sessione al CIM attraverso UDP;
2. tramite un ID valido, avvia uno del CAM Manager;
3. chiede l'ID di sessione al CAM Manager attivato tramite TCP;
4. tramite ID valido, imposta l'area di interesse del CAM Manager;
5. chiede al CAM Manager tutti i CAM ricevuti nella sua area di competenza dall'ultima richiesta.

Se viene memorizzato l'ID di sessione del CAM Manager e viene mantenuta aperta la connessione, è sufficiente ripetere l'ultimo passo per richiedere periodicamente i CAM più recenti.

Dal punto di vista hardware, il CIM è ottimizzato per il multi-threading ed è pertanto eseguito su una macchina virtuale Ubuntu configurata con 2 processori virtuali e 4 GB di RAM.

3.4 Extended Virtual Sensing

L'EVS è il componente centrale del sistema ICA oggetto di questa tesi. Tramite l'algoritmo di rilevamento di collisioni presentato nel sottoparagrafo successivo, è in grado, a partire dai CAM inviati dai veicoli, di predire le possibili collisioni presso le intersezioni e prevenirle inviando dei messaggi di allarme ai veicoli coinvolti. I dati che vengono utilizzati dall'algoritmo sono posizione, velocità, accelerazione e direzione, sono tuttavia presenti anche l'indirizzo IP del Vehicle Simulator che ha prodotto il CAM, per l'eventualità in cui sia necessario inviare un messaggio di allarme in risposta, e l'ID univoco del veicolo che genera il CAM, al fine di analizzare le prestazioni del sistema tramite post-processing. L'applicazione, con periodicità di 5 ms, esegue le azioni seguenti:

1. richiede al CIM i dati dei CAM più recenti;
2. se sono presenti nuovi dati, esegue l'algoritmo di rilevamento di collisioni utilizzando in input i dati appena ricevuti e i dati ricevuti nei cicli di elaborazione precedenti, altrimenti si mette in attesa del ciclo successivo;
3. al termine della elaborazione, se viene rilevata una collisione, tramite connessione TCP al DENM Decider è inviato un pacchetto contenente i veicoli coinvolti (il loro identificatore e indirizzo IP), il tempo di rilevamento, il tempo e le coordinate del luogo in cui è prevista la collisione;

4. se il tempo di elaborazione è inferiore ai 5 ms, l'EVS si mette in attesa del tempo rimanente, altrimenti dà subito il via ad un nuovo ciclo di elaborazione.

Dal momento che l'azione del EVS è critica per l'affidabilità del sistema ICA, è fondamentale che il tempo di esecuzione del ciclo si mantenga quanto più possibile al di sotto dei 5 ms. Poiché attualmente l'algoritmo di rilevamento di collisioni non è ottimizzato per un'esecuzione parallela, l'EVS viene eseguito su una macchina virtuale Ubuntu dedicata con un solo core virtuale e 2 GB di memoria RAM.

3.4.1 Algoritmo di rilevamento delle collisioni

L'algoritmo di rilevamento delle collisioni è sicuramente la parte più interessante dell'EVS. Non solo è in grado di rilevare collisioni tra veicoli, ma, essendo un algoritmo generico basato sulle traiettorie, è in grado di rilevare collisioni tra qualunque tipo di entità e pertanto può essere usato anche per prevenire collisioni con pedoni o con ostacoli. Sebbene nell'EVS si tenga conto anche dell'accelerazione dei veicoli, in Algoritmo 1, per semplicità di esposizione, questa verrà ignorata.

Algorithm 1 Pseudocodice dell'algoritmo di previsione delle collisioni

Require: $v_0, \vec{p}_0, \vec{v}_0, \mathcal{V}$

- 1: $\mathcal{C} \leftarrow \emptyset$
- 2: $\vec{p}_0(t) \leftarrow \vec{p}_0 + \vec{v}_0 \cdot t$
- 3: **for all** $v \in \mathcal{V}$ **do**
- 4: $\vec{p}_v(t) \leftarrow \vec{p}_v + \vec{v}_v \cdot t$
- 5: $\vec{d}(t) \leftarrow \vec{p}_0(t) - \vec{p}_v(t)$
- 6: $D(t) \leftarrow |\vec{d}(t)|^2$
- 7: $t^* \leftarrow t : \frac{d}{dt} D(t) = 0$
- 8: **if** $t^* < 0$ **or** $t^* > t2c_t$ **then**
- 9: **continue**
- 10: **end if**
- 11: $d^* \leftarrow \sqrt{D(t^*)}$
- 12: **if** $d^* \leq s2c_t$ **then**
- 13: $\mathcal{C} \leftarrow \mathcal{C} \cup \{v\}$
- 14: **end if**
- 15: **end for**
- 16: **if** $\mathcal{C} \neq \emptyset$ **then**
- 17: $\mathcal{C} \leftarrow \mathcal{C} \cup \{v_0\}$
- 18: **end if**
- 19: $\mathcal{V} \leftarrow \mathcal{V} \cup \{v_0\}$
- 20: **return** \mathcal{C}

L'algoritmo è eseguito per ogni nuovo CAM ricevuto da un veicolo che transita nell'area monitorata. In input richiede:

- il CAM del veicolo v_0 appena ricevuto, al cui interno devono essere presenti i vettori posizione \vec{p}_0 e velocità \vec{v}_0 , di cui l'ultimo di questi contiene l'informazione di direzione;

- i CAM più recenti inviati dagli altri veicoli nell'area, memorizzati nel Set \mathcal{V} .

Nelle righe 1 e 2 viene inizializzato il set \mathcal{C} , che contiene i veicoli che potrebbero collidere con il veicolo v_0 , e viene calcolata la posizione $\vec{p}_0(t)$ del veicolo v_0 per tutti gli istanti di tempo futuri. Dopodiché, per ogni veicolo $v \in \mathcal{V}$, viene calcolata la posizione $\vec{p}_v(t)$ di tale veicolo nel futuro e la sua distanza $\vec{d}(t)$ dal veicolo v_0 . In questo modo, tramite $\vec{d}(t)$, è nota la distanza di v_0 dal veicolo v per ogni istante di tempo ed è così possibile trovare il minimo di tale valore. Per farlo, si considera inizialmente (riga 6)

$$D(t) = |\vec{d}(t)|^2$$

$$D(t) = (\vec{v}_0 - \vec{v}_v) \cdot (\vec{v}_0 - \vec{v}_v)t^2 + 2(\vec{p}_0 - \vec{p}_v) \cdot (\vec{v}_0 - \vec{v}_v)t + (\vec{p}_0 - \vec{p}_v) \cdot (\vec{p}_0 - \vec{p}_v)$$

di cui siamo interessati al valore minimo raggiunto negli istanti di tempo successivi. Tale valore è raggiunto al tempo t^* , che è il l'istante in cui la distanza tra i due veicoli sarà minima.

$$t^* = t : \frac{d}{dt}D(t) = 0$$

ovvero

$$t^* = -\frac{(\vec{p}_0 - \vec{p}_v) \cdot (\vec{v}_0 - \vec{v}_v)}{|\vec{v}_0 - \vec{v}_v|^2}$$

Al fine di ridurre i falsi positivi, l'algoritmo cerca di trovare solo collisioni imminenti, dal momento che viene presupposto che un veicolo non mantenga valori costanti di velocità e direzione a lungo. Pertanto un valore di tempo di soglia $t2c_t$ è utilizzato per discriminare le collisioni non imminenti che non verranno segnalate. Naturalmente tale valore non può essere eccessivamente basso, altrimenti c'è il rischio che la collisione, pur rilevata, non sia segnalata in tempo al veicolo. Inoltre c'è anche un altro fattore da tenere presente: le posizioni dei su cui si basa l'EVS si riferiscono al paraurti anteriore dei veicoli. Pertanto, per tenere conto delle dimensioni fisiche delle automobili e per tollerare piccoli errori dei valori di posizione, si considera una ulteriore soglia, ovvero una distanza minima $s2c_t > 0$ tra due veicoli sotto la quale viene prodotto un allarme di collisione.

Se $t^* < 0$ allora i due veicoli si stanno allontanando, se invece $t^* > t2c_t$ allora la collisione non è considerata imminente e pertanto viene scartata (linea 8). È pertanto possibile analizzare il veicolo v successivo. Se invece t^* è compreso tra 0 e $t2c_t$, allora si può calcolare la distanza dei veicoli in esame al tempo t^* . Se la distanza rientra al di sotto della soglia $s2c_t$ allora tale evento equivale ad una collisione, pertanto il veicolo v viene incluso nell'insieme \mathcal{C} dei veicoli con cui v_0 è a rischio collisione, altrimenti si passa all'iterazione successiva.

Dopo che tutti i CAM in \mathcal{V} sono stati processati, se è stata rilevata almeno una collisione, il veicolo v_0 viene incluso all'interno dell'insieme \mathcal{C} dei veicoli a rischio collisione, altrimenti è ritornato l'insieme vuoto. Infine, il CAM del veicolo v_0 è incluso nell'insieme \mathcal{V} dei CAM ricevuti così da poter essere analizzato al prossimo avvio dell'algoritmo.

Fatto n il numero di CAM ricevuti, l'algoritmo ha complessità $O(n)$, tuttavia, sebbene la sua implementazione nell'EVS ancora non sia ottimizzata in tal senso, è altamente parallelizzabile. La scelta dei valori di soglia $t2c_t$ e $s2c_t$ non è banale ed influisce direttamente sulle prestazioni dell'algoritmo. Il paper "Performance Analysis of C-V2I-based Automotive Collision Avoidance" di Malinverno et al. è un buon punto di partenza per la decisione

di tali valori. Nel paper vengono studiate le prestazioni dell'algoritmo qui descritto con una piattaforma di test basata su OMNeT++ e SimulLTE, utilizzando uno scenario simile [27]. Vengono suggeriti valori di $t2c_t$ e $s2c_t$ pari a 4 s e 3 m, sotto delle ipotesi che verranno presentate nel Paragrafo 6.4 del capitolo successivo e che per i test condotti in questa tesi saranno considerate valide. Tramite una serie di prove empiriche, tenendo presenti le seguenti ipotesi, sono stati scelti i valori:

$$t2c_t = 3,7 \text{ s}$$

$$s2c_t = 3,5 \text{ m}$$

3.5 DENM Decider

Il DENM Decider è il componente software che completa il sistema ICA. Il suo compito è di ricevere le collisioni rilevate dall'EVS e codificarle in appositi DENM che poi invia ai veicoli coinvolti. Tuttavia, non si limita a questo, ma svolge anche alcune funzioni di supporto:

- coordina più istanze di EVS attive contemporaneamente;
- temporizza l'invio di DENM destinati ai veicoli.

La più importante di queste è coordinare più istanze di EVS nel caso fossero attive in un determinato momento. All'occorrenza, è possibile parallelizzare il carico del collision detector su più macchine, ciascuna delle quali si occupa di un settore dell'area di copertura del CIM. Il DD si occupa di segnalare agli EVS attivi l'attivazione di nuove istanze, in modo che gli EVS possano prendere decisioni ottimali per il ridimensionamento dei settori di copertura.

Una parte dei settori deve essere in comune tra più EVS; se così non fosse, non sarebbe possibile rilevare una collisione imminente tra due veicoli che transitano in due settori diversi. In questo caso non è possibile evitare che i veicoli ricevano un avviso di collisione duplicato generato da ciascuno degli EVS incaricati dell'area, poiché il codice di evento generati dagli EVS saranno distinti. Quando invece è un solo EVS a rilevare la collisione, poiché il codice evento è univoco per lo stesso EVS, il DD è in grado di impedire l'invio di duplicati e di regolare l'intervallo di tempo di generazione dei DENM relativi all'evento.

La comunicazione tra DD ed EVS avviene tramite una connessione affidabile TCP, quella tra DD e veicoli è invece realizzata tramite trasporto su UDP. Il DD non è computazionalmente esigente; le prestazioni sono più che accettabili usando una macchina virtuale dotata di 1 vCPU e 2 GB di RAM.

3.6 Testbed MEC

Il testbed MEC si basa su una implementazione proprietaria dei punti di riferimento Mp2 specificati dallo standard ETSI MEC [2]. Tramite l'implementazione dei reference point Mp2, è possibile specificare delle regole di reindirizzamento del traffico per reindirizzare i pacchetti all'interno del MEC Host a partire dalle informazioni fornite dalla RAN OpenAirInterface a proposito degli UE registrati. Nel caso specifico, sono definite le regole che

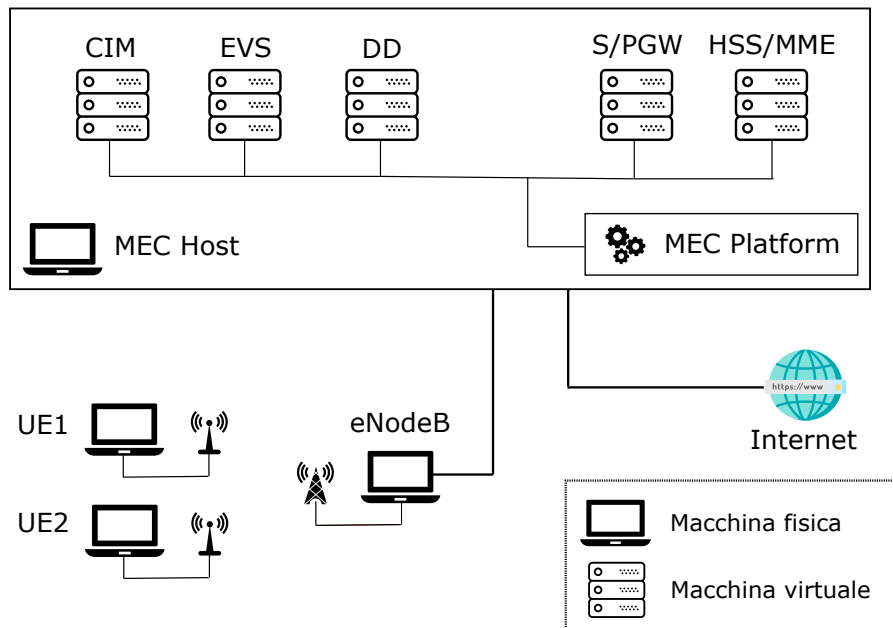


Figura 3.2: Composizione del testbed MEC

inoltrano verso il CIM tutto il traffico UDP inviato dagli UE verso una porta predefinita. Lo UE riesce così a contattare il CIM qualunque sia l'indirizzo IP di destinazione scelto, indifferente sulla configurazione di rete del MEC Host. Tutto il traffico che non verifica la condizione della regola è invece inoltrato verso internet.

Questo è un comportamento voluto: così facendo, in una implementazione reale, si avrebbe la flessibilità di lasciare al vertical il compito di definire le regole appropriate in funzione della configurazione delle sue MEC App tramite il CFS Portal invece di riconfigurare appositamente gli UE, operazione decisamente più difficoltosa, specialmente considerando che nel caso automotive si tratterebbe di veicoli.

Il CIM, EVS e DENM Decider sono installati come applicazioni MEC su macchine virtuali dedicate, così come il S/PGW, HSS e MME, utilizzando kvm come hypervisor. Il MEC Host è installato su un singolo notebook Dell con CPU Intel i7-8550U e 16 GB di RAM.

3.7 Testbed Openstack

Il testbed Openstack è realizzato in modo molto simile al testbed MEC. CIM, EVS e DD sono nuovamente ospitati su macchine virtuali, l'EPC è invece installato su una singola macchina virtuale invece di essere suddiviso su 2 istanze separate. Anche l'hardware su cui è ospitato è simile, si tratta di un notebook HP con i7-8550U, con Turbo Core disattivato, e 16 GB di RAM che in più può vantare un disco allo stato solido, utile per velocizzare l'avvio del sistema e delle macchine virtuali.

La differenza principale col testbed MEC è che qui si abbandona l'architettura MEC standardizzata in favore della 5G-Transformer. Il testbed è configurato come Mobile

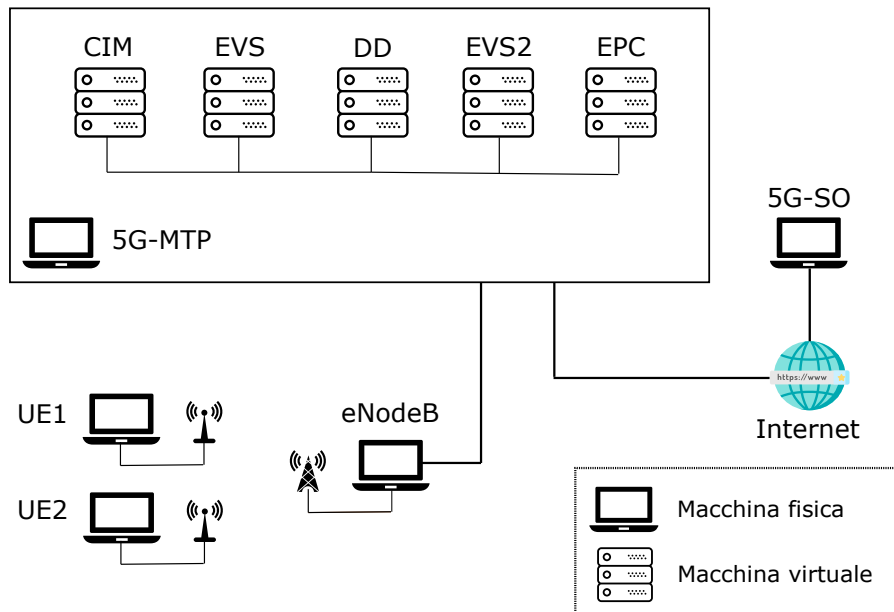


Figura 3.3: Composizione del testbed Openstack

Transport and computing Platform basato su Openstack Devstack in versione Queens.

Openstack è un sistema operativo cloud opensource in grado di gestire un ampio pool di risorse di calcolo, memorizzazione e networking all'interno di un datacenter. È un insieme di diversi componenti, ciascuno con un ruolo ben specifico, che comunicano tra loro tramite delle API REST utilizzando un sistema di autenticazione comune.

Le immagini delle macchine virtuali sono gestite tramite il progetto Glance, con cui sono state create le immagini di ciascuna macchina all'interno di Openstack. Dopodiché le immagini vengono lanciate dal componente Nova, il progetto che rende possibile l'instanziazione delle risorse di calcolo, ovvero le macchine virtuali. Ogni macchina è configurata per avere indirizzi IP privati appartenenti ad uno spazio di indirizzi riservati. Per rendere le istanze gestibili dal 5G-SO, è stato necessario assegnare a ciascuna macchina un ulteriore indirizzo, detto floating IP, che ha la proprietà di poter essere assegnato e rilasciato dinamicamente e di essere raggiungibile all'esterno dell'ambiente virtualizzato.

Openstack forza policy di sicurezza molto restrittive all'interno dell'ambiente virtualizzato che ospita. Ad esempio, Neutron, il componente Openstack che si occupa di fornire i servizi di rete alle macchine virtuali, non consente alle macchine, come impostazione predefinita, la generazione di pacchetti IP che abbiano un indirizzo IP sorgente esterno alla rete privata su cui si trovano. Questo implica che l'EPC virtuale non è in grado di instradare il traffico degli UE (172.16.0.0/16) all'interno della rete privata di Openstack (10.0.0.0/24) e quindi non rende il CIM raggiungibile dal Vehicle Simulator. Per risolvere il problema, ad ogni istanziazione è necessario abilitare una eccezione di sicurezza per il vEPC per consentirgli di superare il blocco.

Il punto di forza di questa configurazione è la possibilità di fare scaling: quando la quantità di veicoli presenti è talmente elevata che le prestazioni del sistema ICA iniziano a risentirne, il carico dell'EVS viene ripartito automaticamente con una seconda istanza

avviata appositamente, così da rispettare i vincoli di funzionamento del sistema. Maggiori dettagli sono presenti nel Capitolo 5 dedicato alla funzione.

Si tenga presente che il testbed Openstack è temporalmente successivo alla controparte MEC. Durante lo sviluppo sono state introdotte ottimizzazioni varie ai diversi componenti che con ogni probabilità si rifletteranno in un miglioramento delle prestazioni.

Capitolo 4

Metodologia di test

L'attività di raccolta dei dati è stata senza dubbio la più lunga durante il lavoro di tesi. La moltitudine di componenti presenti e la non sempre stabile connessione radio tramite OAI ha reso il compito di testing, oltre che lungo, delle volte frustrante; ciò nonostante è stato svolto con il massimo rigore. L'esecuzione dei test può essere scomposta in tre fasi:

1. Preparazione: le macchine fisiche e virtuali vengono avviate se non sono già in esecuzione. Si tratta di 5 macchine fisiche (4 nel caso del testbed MEC) e 4 macchine virtuali. Dopo l'avvio dell'EPC, che conta di tre processi distinti (HSS, MME e S/PGW), vengono collegati gli UE. Sono poi preparate le tracce che vengono date in input ai Vehicle Simulator ed infine ogni macchina viene configurata per il salvataggio dei dati in base alla traccia in esecuzione.
2. Svolgimento: per ciascuna traccia, vengono lanciati in sequenza i processi del sistema ICA così come quelli dedicati a raccogliere i dati di log. Terminata la traccia, è verificato che non ci siano state perdite di pacchetti fuori norma, vengono arrestati tutti i processi e riavviati per la traccia successiva. Nel caso ci si accorga di qualche anomalia, dopo aver riconfigurato tutte le macchine, si ripete il test.
3. Conclusione: i dati vengono prelevati da ogni macchina, catalogati e salvati per il post processing. Poiché la maggior parte di queste macchine sono virtuali, il modo più facile per raccogliere i dati è tramite rete (scp), pertanto occorre tenere presenti gli indirizzi IP e le credenziali di accesso di ogni macchina, diversi per ogni testbed.

Nei risultati verranno mostrate le prestazioni al variare di alcune caratteristiche di rilievo della configurazione:

- testbed,
- densità veicolare,
- tecnologia di rete di accesso, e
- numero di istanze di EVS attive contemporaneamente.

4.1 Densità veicolare

Il numero di veicoli contemporaneamente simulati attraverso il Vehicle Simulator è una variabile molto importante per le prestazioni del sistema ICA. Il numero di veicoli complessivamente presenti nell'area di copertura del CIM/EVS influisce direttamente sulla capacità di rete impiegata, per l'aumento del numero di CAM trasmessi e DENM ricevuti, e sulle risorse di calcolo consumate da CIM ed EVS. Quest'ultimo, in particolare, per via della complessità quadratica dell'algoritmo che esegue, è il componente che più soffre nel mantenere un buon livello di prestazioni al crescere del traffico veicolare.

Il numero di veicoli delle simulazioni è determinato al momento della preparazione delle tracce CSV, mediante la scelta del valore di intensità della distribuzione di Poisson associata al numero di veicoli immessi nella rete stradale della mappa SUMO. Dal momento che ai veicoli viene impedito di rallentare, l'intensità dei veicoli in ingresso è proporzionale al numero di veicoli presenti nella mappa a regime così come alla loro densità lineare. I valori di intensità (λ) per i test sono riportati nella tabella 4.1.

Lambda	Veicoli	Densità (km^{-1})
0,5	35	6
1	70	12
1,5	104	18

Tabella 4.1: Numero di automobili e densità veicolare al variare dell'intensità della distribuzione di Poisson che ne regola l'inserimento nella rete stradale

Si noti che questi valori sono corretti solo a regime, ovverosia solo quando la quantità di auto che vengono inserite nella rete stradale è pari al numero di auto che vi escono. In Figura 4.1 è graficamente mostrato come il numero di veicoli cresca nella prima parte della simulazione per poi oscillare intorno ad un valore medio costante. Il tempo t_r dopo

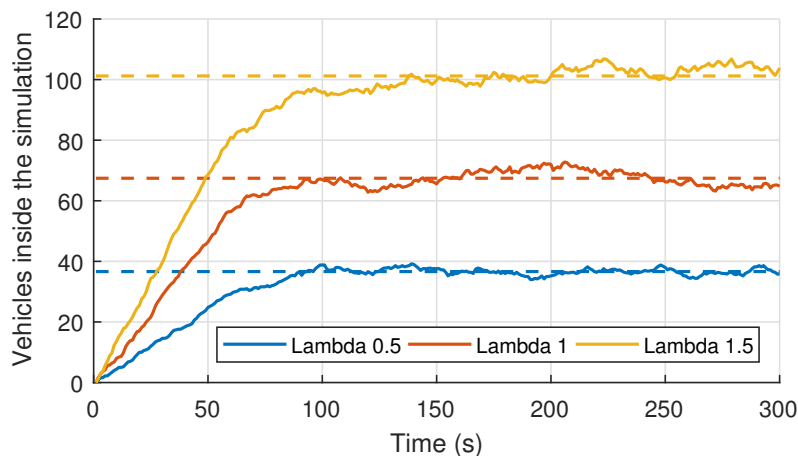


Figura 4.1: Numero di veicoli contemporaneamente presenti all'interno della simulazione

il quale il sistema è a regime, poiché le auto hanno tutte la stessa velocità, è pari al tempo di percorrenza del percorso più lungo della mappa, che è di lunghezza 1300 m. Pertanto $t_r = 94$ s.

4.2 Rete di accesso

OperAirInterface sugli USRP Ettus non si è sempre dimostrata un'interfaccia radio completamente affidabile a causa di sporadiche disconnessioni e perdite di pacchetti. Poiché la rete di accesso è un componente fondamentale per l'applicazione ICA, si è ritenuto opportuno simulare reti alternative, ipotizzando anche di collocare il sistema ICA non più sulla periferia della rete di accesso ma all'interno di una infrastruttura cloud.

La RAN OpenAirInterface è quindi stata sostituita con uno switch Ethernet e per emulare il comportamento della rete di accesso è stato utilizzato NetEm sugli host ai due estremi del collegamento. Le configurazioni usate per i test sono descritte in tabella 4.2, insieme ai valori di media e deviazione standard dei ritardi associati.

Rete di accesso		Latenza	
Radio Access Network	Infrastruttura	Media (ms)	Deviazione standard (ms)
LTE OAI	MEC	23,93	4,39
LTE NetEm	MEC	23,93	4,39
LTE NetEm	Cloud NetEm	68,16	6,58

Tabella 4.2: Tecnologie di rete di accesso utilizzate per lo svolgimento dei test sul sistema

NetEm (Network Emulator) è una funzione dei servizi di controllo del traffico del kernel Linux che consente di aggiungere ritardi, perdite e duplicazioni in uscita dallo stack di rete di una determinata interfaccia. La sua flessibilità nel gestire un ritardo artificiale dei pacchetti è data dalla possibilità di associare una distribuzione di probabilità al ritardo. Per la parte di test svolta tramite NetEm si sono utilizzati valori di media e deviazione standard tratti nel modo seguente.

- LTE NetEm + MEC: tramite il tool ping, sono stati raccolti i dati sulle prestazioni dell'interfaccia OAI collegata al MEC al fine di mantenere un comportamento quanto più possibile simile a quello registrato senza NetEm. L'utility ping permette di misurare periodi di round-trip con precisione al centesimo di millisecondo.
- LTE + Cloud NetEm: in questo caso, i dati sono stati raccolti col tool ping impostato con destinazione il server Web di una università olandese, allo scopo di simulare una latenza realistica tra l'utente e il provider cloud che lo serve. Per simulare la rete di accesso (in particolare in termini di prestazioni dell'interfaccia radio e dell'instradamento dei pacchetti), è stata utilizzato un sistema Ubuntu collegato in tethering USB a un tablet iPad Mini LTE (Cat. 3) registrato su rete mobile Vodafone.

Per modellare il ritardo, per una questione di semplicità, è stata scelta la distribuzione di probabilità normale; non è stata impostata alcuna perdita di pacchetti. NetEm opera solo sui pacchetti in uscita da una interfaccia di rete, pertanto è stato configurato ed avviato

sulle macchine responsabili di produrre i pacchetti che transitano sulla rete di accesso: il Vehicle Simulator e il DENM Decider. I valori di media e deviazione impostati sono stati ripartiti di conseguenza, utilizzando le regole dell'algebra delle variabili aleatorie.

Capitolo 5

Scaling in risposta all'aumento del traffico veicolare

Il software di cloud computing Openstack rende estremamente veloce la creazione di nuove immagini ed istanze di macchine virtuali. Parte del progetto 5G-Transformer consiste nello sfruttare questa caratteristica per consentire lo scaling automatico quando il sistema viene sottoposto ad un carico eccessivo, allo scopo di rispettare i requisiti di funzionamento previsti (che possono essere Service Level Agreements).

L'applicativo EVS ha degli stringenti requisiti per quanto riguarda il tempo di calcolo che richiede per elaborare le collisioni, che si riflette direttamente sul ritardo di generazione di DENM nel momento in cui una collisione è imminente. Se i requisiti non vengono soddisfatti si corre il serio rischio che il ritardo sia tale da rendere la collisione inevitabile. Il tempo di calcolo e dunque il ritardo aumentano quando il traffico veicolare cresce di intensità, infatti un maggior numero di veicoli comporta un numero proporzionalmente superiore di CAM che l'EVS deve analizzare. Lo scaling del sistema è la soluzione: si può parallelizzare l'applicazione EVS su più macchine virtuali tramite il lancio di una ulteriore istanza di EVS, se viene rilevato che quella in esecuzione non è sufficiente a gestire efficacemente il traffico veicolare nella zona di competenza. In questo modo è sempre garantito il rispetto dei requisiti del servizio.

5.1 Scale-up e scale-out

Per scalabilità si intende l'idea che le applicazioni e gli elementi infrastrutturali di un sistema possano essere ampliati per sostenere l'aumento del carico, rispetto ad un valore nominale, a cui il sistema è sottoposto. Sostenere l'aumento del carico equivale a dire che l'applicazione deve continuare ad operare secondo determinati requisiti o SLA considerando la possibilità che le prestazioni decadano per la crescita di carico. Una delle caratteristiche che contraddistingue i servizi di cloud computing, e che è stata ereditata dai sistemi edge computing, è la facilità ed immediatezza con cui i servizi che ospitano possono accedere alle risorse aggiuntive nel momento in cui ne hanno bisogno per poi rilasciarle quando non sono più necessarie.

Per automatizzare lo scaling vengono definite delle regole composte da condizioni. Le condizioni possono essere di diversi tipi: possono essere basate su metriche dell'infrastruttura, come carico della CPU e memoria occupata, oppure possono essere impostate su metriche di alto livello come ad esempio un rapporto tra costi e benefici. Quando ciascuna delle condizioni di una regola è verificata, viene fatto partire il comando di scaling alla piattaforma.

Esistono due categorie di scaling:

Scale-up o Scaling verticale È basato sull'incremento delle risorse già dedicate al sistema per raggiungere le prestazioni desiderate, spesso tramite un aumento di CPU e di RAM riservati all'applicazione. Poiché i sistemi operativi più comuni non supportano la modifica delle CPU e RAM disponibili a run-time, spesso consiste nel muovere l'istanza virtuale del servizio su una macchina più potente. Si tratta del tipo di scaling più immediato, poiché richiede nulle o poche modifiche all'applicazione affinché questa faccia uso delle risorse aggiuntive. Lo svantaggio dello scaling verticale è il costo che aumenta esponenzialmente e il limite di risorse che una singola macchina può offrire.

Scale-out o Scaling orizzontale Solitamente associato ad architetture di calcolo distribuito, consiste nell'assegnare nuove macchine all'applicazione. Virtualmente consentirebbe di aumentare indefinitamente la quantità di risorse assegnate ad una applicazione, tuttavia ha lo svantaggio che l'applicazione deve essere progettata appositamente per sfruttare la parallelizzazione del carico sulle macchine. Pertanto è la soluzione meno immediata da implementare e viene considerata solo se inevitabile.

Così come possono essere assegnate più risorse alle applicazioni quando richiesto, allo stesso modo si possono ridurre le risorse allocate quando il carico lo permette.

Il vantaggio dello scaling è la riduzione di costi determinata dall'assenza di necessità di sovradimensionare le capacità assegnate alle applicazioni per far fronte alle fluttuazioni di occupazione di risorse.

5.2 Implementazione dello scaling

Facendo uso dell'architettura 5G-Transformer, il testbed Openstack è stato configurato come 5GT-MTP, collegato tramite Virtual Private Network al 5G-T Service Orchestrator, collocato presso il centro di ricerca CTTC a Castelldefels, vicino a Barcellona. Il 5GT-SO è responsabile dell'istanziazione e dell'arresto delle macchine virtuali del testbed, ed è in grado di monitorarne lo stato di funzionamento tramite delle metriche esposte dalle macchine stesse. Non essendo il Service Orchestrator ancora capace di eseguire uno scaling verticale, soluzione trasparente per aumentare le risorse a disposizione delle VM, si è lavorato sul testbed per programmarlo a gestire uno scaling orizzontale. La metrica con cui è stata definita la condizione di scaling del testbed è il carico medio della CPU, essendo altamente correlata al tempo di elaborazione dell'EVS.

Lo scaling prevede, al raggiungimento di una soglia prefissata, l'istanziazione di una seconda VM EVS a partire dalla stessa immagine del disco della prima istanza. La macchina deve essere in grado di adattarsi al compito per cui viene lanciata, comprendendo all'avvio il suo ruolo nel testbed. Per farlo, è stato modificato il DENM Decider in modo

che, quando un EVS all'avvio lo contatta, informa gli altri EVS eventualmente già attivi. In figura 5.1 è rappresentata graficamente, tramite un diagramma a traliccio, la sequenza delle comunicazioni delle componenti del testbed per configurarsi correttamente al lancio di un nuovo EVS.

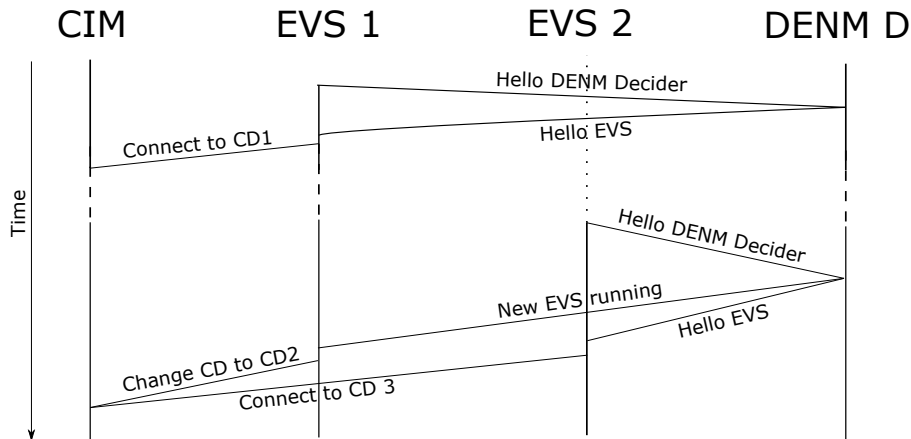


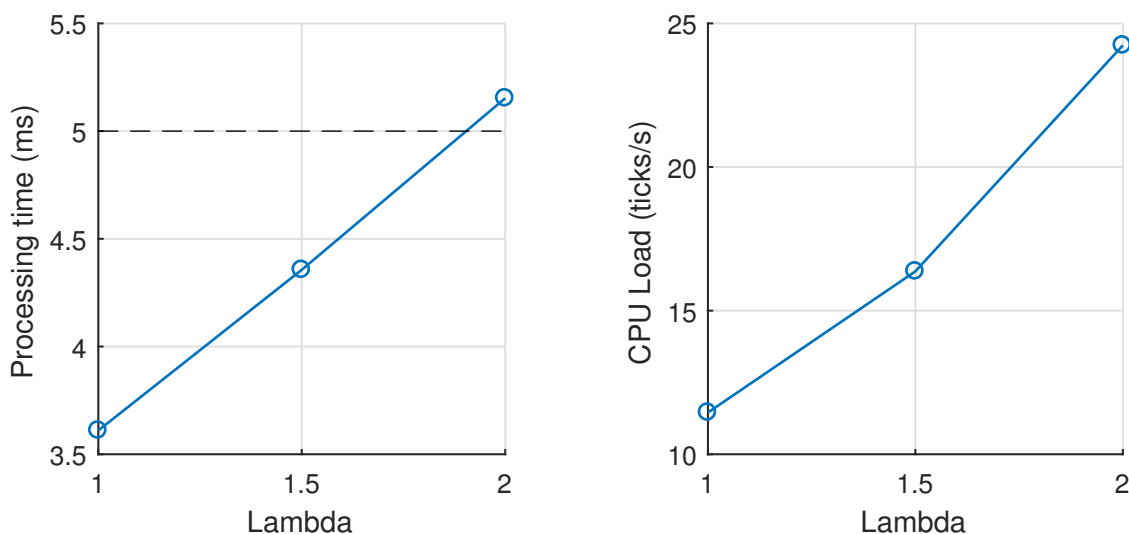
Figura 5.1: Procedura di configurazione degli EVS quando una nuova istanza viene lanciata

La sequenza di operazioni che svolge l'EVS dall'avvio è la seguente:

1. contatta il DENM Decider per stabilire la connessione con cui poi gli inoltrerà i messaggi di allarme destinati ai veicoli;
2. contatta il CIM e si mette in ascolto per ricevere i CAM;
3. se nel mentre che è in esecuzione riceve una notifica di un nuovo EVS collegato da parte del DD, comanda al CIM il cambio di coordinate e raggio di copertura e poi riprende a processare i CAM in arrivo.

Dal momento che ancora non sono disponibili delle API con cui i programmi all'interno dell'infrastruttura virtualizzata possano conoscere l'ambiente in cui sono stati avviati, le informazioni riguardo le coordinate degli incroci da servire sono inserite staticamente nell'EVS.

Il requisito su cui ci si è basati per individuazione della soglia è stato il tempo di ciclo di elaborazione dell'EVS, che deve essere pari a non più di 5 ms. Più in particolare, è necessario che il 99,9-esimo percentile rimanga al di sotto dei 5ms. Come si vede nella figura 5.2a, il valore di soglia è raggiunto per valori di intensità (λ) pari a circa 1,9, che a regime corrispondono a 130 veicoli contemporaneamente tracciati. Per questo livello di intensità il consumo di CPU corrispondente è del 24% (in figura 5.2b), che rappresenta il livello massimo che può essere raggiunto dal processore affinché l'applicativo EVS rimanga all'interno dei requisiti di funzionamento. Lo scaling pertanto deve avvenire prima del raggiungimento del valore critico di carico.



(a) 99.9-esimo percentile del tempo di elaborazione dell'EVS

(b) Carico del processore al variare del traffico veicolare

Figura 5.2: Studio delle prestazioni dell'EVS per l'individuazione del limite di carico

5.3 Test delle prestazioni dopo lo scaling

Allo scopo di verificare che effettivamente il comportamento del sistema migliori, verranno confrontate le prestazioni del testbed Openstack con una e due istanze di EVS contemporaneamente attive, ciascuna delle quali configurata per servire una sottosezione dell'area di copertura. Nella tabella 5.1 sono riportati la lunghezza della rete stradale servita e il raggio di copertura degli EVS a seconda della configurazione impiegata.

Instanze EVS	Rete stradale per EVS (percentuale)	Raggio area di copertura
1	5,8 km (100%)	650 m
2	3,2 km (55%)	400 m

Tabella 5.1: Configurazione del testbed a seconda del numero di istanze EVS attive contemporaneamente

Attualmente è possibile definire solo aree circolari, tuttavia, sebbene la somma dell'aree coperte dagli EVS con doppia istanza attiva sia inferiore al caso con istanza singola, la rete stradale complessivamente servita è esattamente la stessa. Per ottenere questo risultato, come si vede dalla Figura 5.3, nella configurazione con 2 istanze, gli EVS condividono la copertura di una parte di mappa, dunque ciascuno degli EVS non avrà da gestire l'esatta metà del numero di CAM, ma approssimativamente il 55%. In ogni caso, poiché l'algoritmo dell'EVS non ha complessità lineare, ci si aspetta comunque un carico computazionale inferiore alla metà del carico originale.

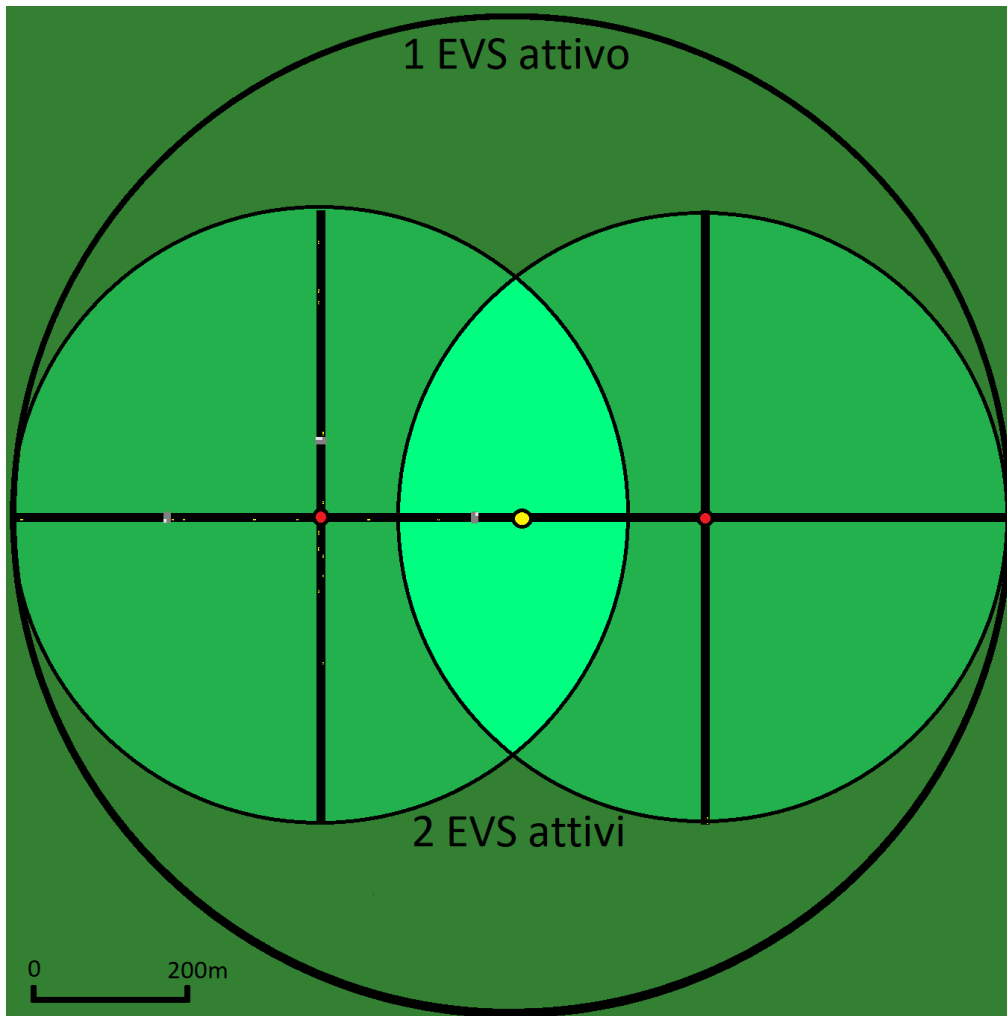


Figura 5.3: Aree servite dagli EVS nel caso di singola e doppia istanza attiva

Capitolo 6

Risultati

6.1 Consumo di CPU

Allo scopo di verificare che le macchine siano correttamente dimensionate, è stato inizialmente studiato il carico di lavoro sottoposto alla Central Processing Unit delle macchine impiegate nel testbed. Una CPU sottodimensionata per l'applicativo con cui viene impiegata può determinare malfunzionamenti dei programmi, ritardi di elaborazione ed in generale prestazioni complessive del testbed peggiori. È pertanto indispensabile che tutte le macchine del testbed abbiano a disposizione tutta la potenza di cui necessitano.

Per compiere questo test, viene periodicamente analizzato il contenuto dello pseudo-filesystem `/proc`. Di ogni macchina viene considerato il carico imposto dal singolo processo che si occupa di svolgere la sua funzione all'interno del sistema ICA. Dei processi coinvolti, vengono salvate le metriche:

- User time: tempo di processore utilizzato dall'applicazione durante la sua normale esecuzione in ambiente utente, espresso in tick;
- System time: tempo di processore utilizzato dal processo durante le chiamate di sistema, in cui viene fatto il context switch ad ambiente kernel, espresso in tick.

Il carico complessivo è dato dalla somma delle metriche suddette; il tick è una unità di misura che sulle nostre macchine è pari a 10 ms di tempo di CPU. Dal momento che le metriche sono cumulative, per ottenere il valore di consumo in un dato intervallo occorre fare la differenza delle misure catturate agli estremi. Una frequenza maggiore consente delle misure più fini, una frequenza minore implica invece che il valore di consumo è mediato su un tempo più lungo. Come compromesso tra il carico aggiuntivo sulla macchina, determinato dal campionamento dei valori ed il loro salvataggio su disco, e la risoluzione della lettura, si è optato per una frequenza di campionamento di 1 Hz che implica che la misura indicherà il consumo di CPU medio durante un intervallo di 1 s.

Si noti che la misura espressa in tick su secondo, dal momento che un tick equivale a 10 ms di tempo di CPU, equivale al consumo di CPU espresso in percentuale.

6.1.1 EVS

L'EVS è il componente del testbed che più risente dell'aumento di numero di veicoli gestiti dal sistema ICA. L'algoritmo che implementa ha complessità quadratica, pertanto viene naturale aspettarsi che il consumo di CPU cresca similmente al crescere del numero di CAM in arrivo.

In figura 6.1 sono presenti i consumi di CPU medi al variare dell'intensità del traffico veicolare per due testbed MEC e Openstack.

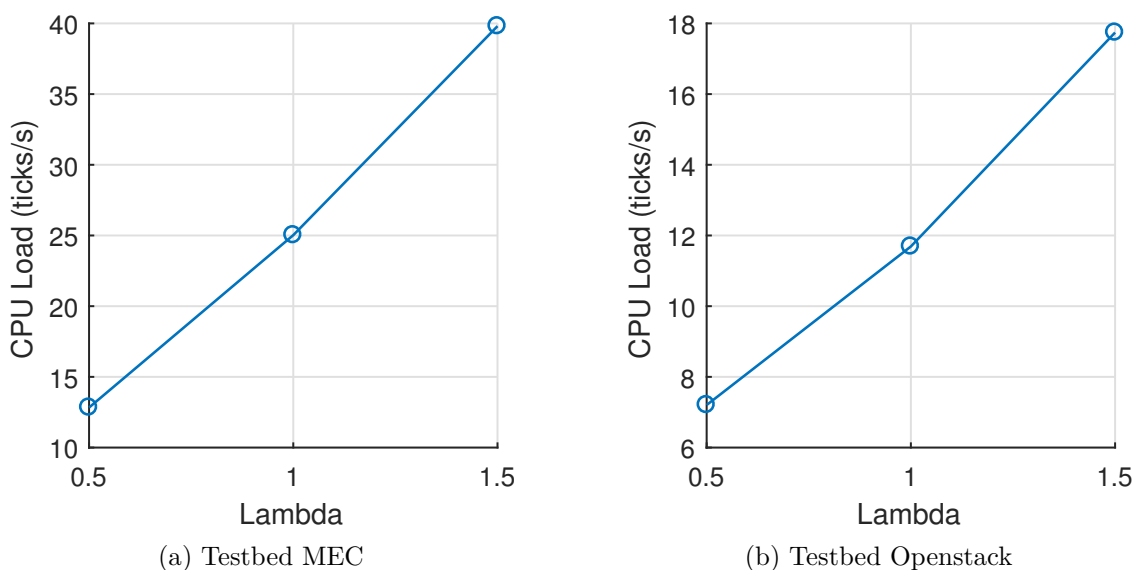


Figura 6.1: Consumo di CPU medio dell'EVS per traffico crescente

Si può notare immediatamente come nell'ultimo di questi, a parità di condizioni, l'EVS richieda meno risorse di calcolo rispetto all'EVS del primo. Approssimativamente si tratta di un consumo di CPU dimezzato. Ciò che determina la differenza sono le migliorie e le correzioni introdotte sul testbed Openstack durante il periodo in cui è stato realizzato.

Non tutto il tempo di calcolo dell'EVS è da imputarsi all'algoritmo di rilevamento delle collisioni. L'EVS consuma risorse anche per gestire le comunicazioni di rete, in particolare col CIM, e per mantenere i suoi dati aggiornati. Di questo si può trovare conferma notando che il consumo di CPU salga meno che proporzionalmente con l'intensità del traffico, come se parte del consumo ne fosse indipendente. Il consumo di base si può stimare pari a circa il 1-2 tick/s.

Per il valore di intensità più basso riportato in figura 6.1, il carico della CPU è ancora fortemente determinato dal consumo di base costante dell'EVS. Per il testbed MEC, corrisponde circa al 15% del totale, per il testbed Openstack, al 25%.

Per valori crescenti, il carattere quadratico dell'algoritmo diventa dominante, pertanto si accentua la flessione della curva del carico. In corrispondenza di $\Lambda = 1,5$, per i testbed MEC e Openstack il consumo di base è pari a rispettivamente circa il 5% e il 10% del totale.

Testbed Openstack con doppia istanza di EVS

Uno dei compiti portati a termine durante la tesi è stato lo sviluppo delle necessarie modifiche ai componenti chiave del testbed per il supporto della parallelizzazione del carico dell'EVS su due istanze. Il motivo che spinge a voler ripartire il carico è puramente prestazionale: a parità di numero di veicoli presenti nella simulazione, il numero di CAM gestiti da ciascuna istanza nella configurazione con doppio EVS è inferiore al numero complessivo gestito dal singolo EVS quando è l'unico in esecuzione.

In figura 6.2 è riportato il confronto di consumo di CPU nei due casi sopra descritti.

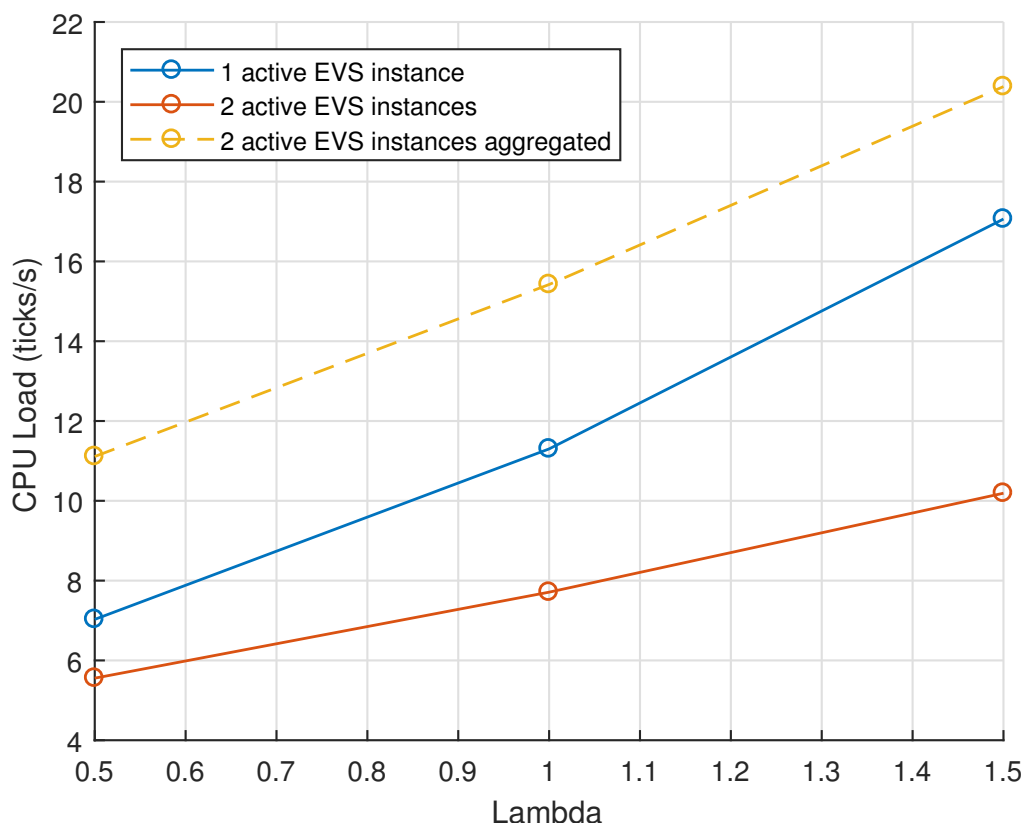


Figura 6.2: Confronto del consumo di CPU medio degli EVS quando 1 o 2 istanze sono attive contemporaneamente nel testbed Openstack

La singola istanza, dovendo gestire da sola la totalità delle auto, ha un consumo di risorse superiore rispetto alle istanze che si ripartiscono il carico.

La linea tratteggiata rappresenta la somma dei consumi di CPU delle 2 istanze parallele di EVS. Si vede che si trova ben al di sopra della linea relativa alla singola istanza, sebbene in entrambe le situazioni il numero di veicoli complessivamente controllati nella simulazione sia identico. Questa differenza non deve sorprendere, infatti la parallelizzazione porta con sé un overhead determinato dal fatto che le istanze parallele non ricevono da processare l'esatta metà del totale dei CAM, ma approssimativamente il 55%, che corrisponde alla frazione di rete stradale coperta.

Inoltre, sempre considerando il consumo complessivo delle due istanze, il carico di base dell'EVS, poiché proprio di ciascuna istanza, influisce per il doppio.

6.1.2 DENM Decider

Il DENM Decider svolge un ruolo assolutamente banale in confronto all'EVS. Il suo compito è di codificare i messaggi di allarme ricevuti dall'EVS e codificarli come DENM prima di inoltrarli ai Vehicle Simulator. Per scrupolo sono comunque stati raccolti i dati sul suo consumo di risorse, riportati in figura 6.3.

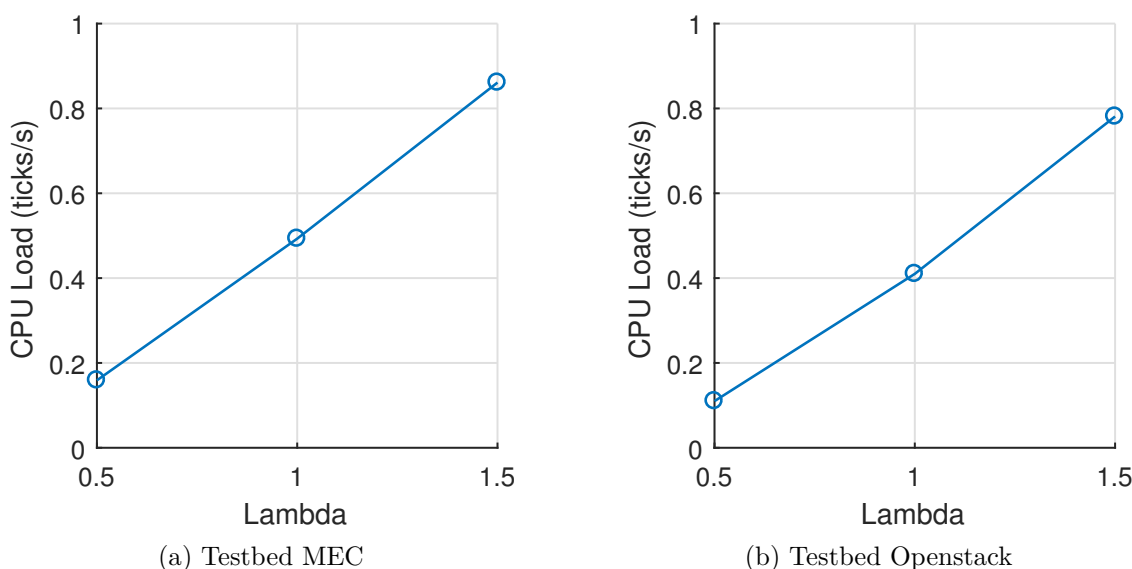


Figura 6.3: Consumo di CPU medio del DENM Decider per intensità di traffico crescenti

Il DENM Decider è estremamente parco col consumo di risorse. In entrambe le configurazioni (fig. 6.3a e 6.3b), il carico sulla CPU a $\Lambda = 1,5$ non eccede il punto percentuale.

6.2 Ritardo di trasmissione

Il ritardo di trasmissione ed elaborazione dei pacchetti influisce pesantemente sulle performance del sistema ICA in esame. Per essere evitata in tempo, una collisione imminente rilevata dal EVS deve essere segnalata il prima possibile ai veicoli coinvolti. Inoltre l'EVS deve avere a disposizione i CAM più aggiornati, per non commettere errori nel valutare la posizione dei veicoli nella mappa in un dato istante. Pertanto, al fine di contenere il più possibile il ritardo di propagazione end-to-end, è necessario che nessuno dei componenti del testbed ritardi l'elaborazione o la trasmissione dei pacchetti in transito.

6.2.1 Rete di accesso

La rete di accesso è l'elemento che contribuisce maggiormente al ritardo end-to-end complessivo del sistema ICA. Questo è l'unico collegamento che viene utilizzato bidirezionalmente. I ritardi di propagazione dipendono dal tipo di rete utilizzata.

Dal momento che con NetEm i ritardi sono preimpostati, l'unica incognita è l'interfaccia radio OpenAirInterface. Per analizzarla è stata usata l'utilità ping per un totale di 20000 pacchetti ICMP. Nella figura 6.4 sono riportate le CDF dei Round Trip Time dei seguenti collegamenti:

- OAI Edge: la RAN OpenAirInterface è collegata direttamente al nodo di Edge Computing;
- LTE Edge: il Vehicle Simulator è collegato al CIM tramite l'interfaccia radio LTE emulata in NetEm;
- LTE Cloud: viene introdotto ulteriore ritardo dal momento che i pacchetti, per arrivare a destinazione, percorrono la rete di trasporto, oltre a quella di accesso.

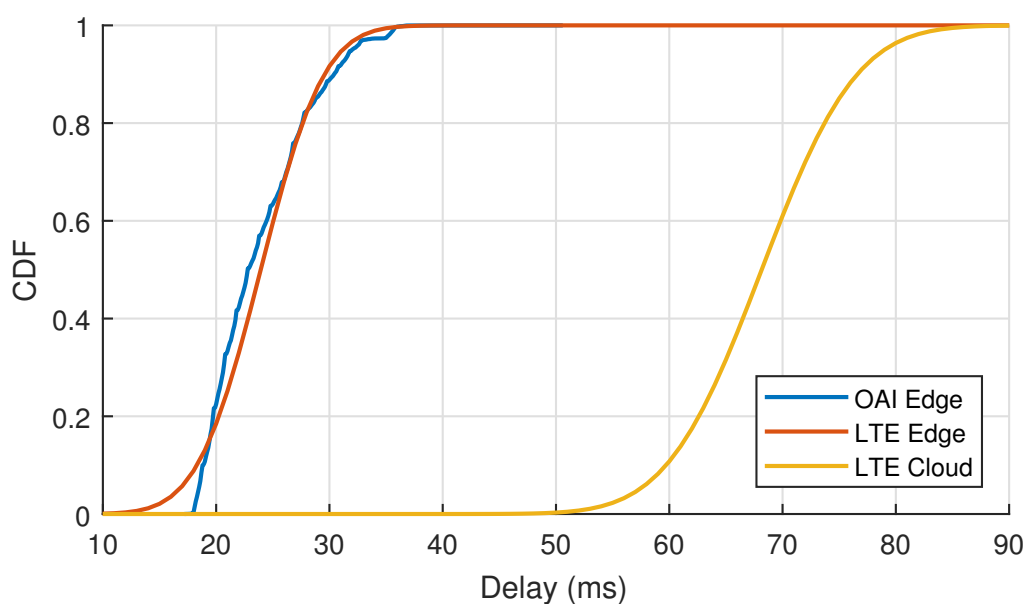


Figura 6.4: CDF del RTT della rete di accesso

Sebbene la connessione LTE Edge sia simulata sulla base dell'interfaccia OAI, non ne riproduce l'esatto comportamento, specialmente per i valori più bassi di ritardo, in cui le relative curve delle funzioni di ripartizione si discostano l'una dall'altra.

6.2.2 EVS

Per riuscire a rilevare in tempo le collisioni imminenti, è importante che il ritardo di elaborazione dell'EVS rispetti i requisiti di funzionamento definiti per il caso d'uso. Il

limite prefissato è pari a 5 ms, entro cui devono rientrare il 99,9% dei cicli di elaborazione totali.

Nelle figure 6.5 e 6.6 sono graficamente rappresentati i tempi di elaborazione degli EVS per i due testbed MEC e Openstack.

Nel caso del testbed MEC, l'EVS ha buone prestazioni per intensità $\Lambda \leq 1$. Dalla funzione di ripartizione in figura 6.5a si evince che il tempo di elaborazione cresce all'aumentare di intensità del traffico, poiché l'EVS si ritrova ad elaborare un numero maggiore di CAM. I cicli più veloci richiedono un tempo nell'ordine dei microsecondi, mentre quelli più lenti eccedono abbondantemente i 3 ms. Nella figura 6.5b sono evidenziate le prestazioni relative ai cicli più lenti. Sebbene il valore medio di ritardo si mantenga al di sotto dei 2 ms, l'EVS non riesce a rispettare i vincoli prefissati: il 99,9-esimo percentile, a $\Lambda = 1,5$, eccede il limite dei 5 ms, rappresentato con la linea nera tratteggiata. Il limite è raggiunto ad un valore di intensità pari ad approssimativamente $\Lambda = 1,2$. Un vincolo meno stringente, fissato ad esempio a 5 ms per il 99% dei casi, sarebbe invece rispettato per intensità anche superiori a quelle testate.

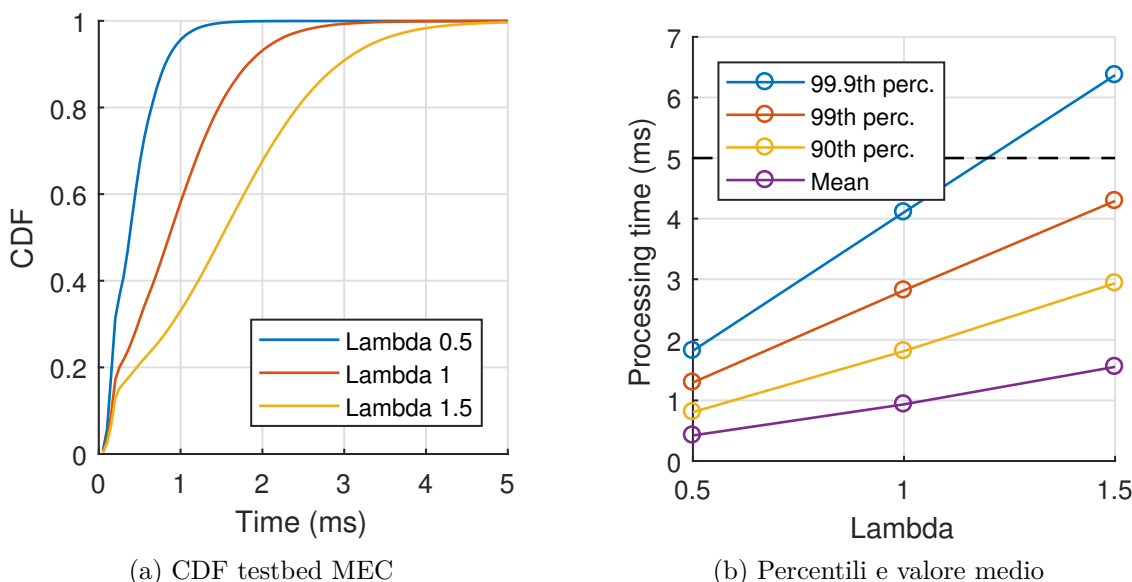


Figura 6.5: Ritardo di elaborazione dell'EVS del testbed MEC

L'EVS del testbed Openstack si comporta ottimamente. La funzione di probabilità cumulata in figura 6.6a mostra dei ritardi perfettamente accettabili lungo tutta la distribuzione. La maggior parte dei ritardi sono estremamente contenuti: tre cicli su quattro terminano entro il millisecondo. La figura 6.6b mostra chiaramente che il 99,9-esimo percentile supera appena i 3 ms per $\Lambda = 1,5$, rimanendo ben al di sotto della soglia dei 5 ms. Dal grafico si può stimare che l'EVS riuscirebbe a rimanere all'interno dei vincoli di funzionamento per intensità superiori a quelle testate, fino a indicativamente $\Lambda = 2$. Il ritardo medio è invece sempre inferiore al millisecondo.

Confrontando le prestazioni dei testbed, la configurazione Openstack si dimostra ben superiore alla controparte MEC. I tempi di elaborazione dell'EVS Openstack sono mediamente la metà di quelli del corrispettivo MEC, riflettendo le differenze osservate nel

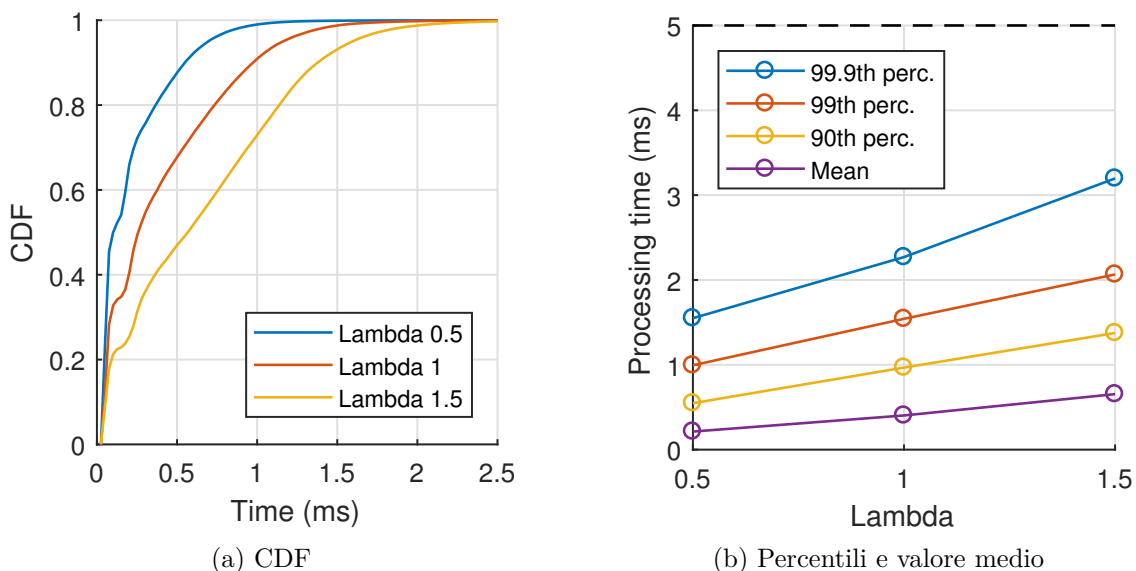


Figura 6.6: Ritardo di elaborazione dell'EVS nel testbed Openstack

sottoparagrafo 6.1.1 in cui si sono confrontati i relativi consumi di CPU. Si era visto che l'EVS Openstack occupasse mediamente la metà delle risorse di calcolo impiegate dall'EVS MEC; allo stesso modo, ora l'EVS MEC è il doppio più lento dell'EVS Openstack. Di nuovo, la causa delle differenze riscontrate è da ricercarsi nelle ottimizzazioni introdotte durante lo sviluppo.

Testbed Openstack con doppia istanza di EVS

Quando due EVS si ripartiscono l'elaborazione dei CAM, ciascuno di essi si ritrova a lavorare con una intensità veicolare equivalente quasi dimezzata. Questa è la caratteristica che con la parallelizzazione si vuole andare a sfruttare. Quando il numero di veicoli aumenta sensibilmente, l'incremento di carico che ne segue potrebbe ridurre le prestazioni al punto di eccedere i vincoli di funzionamento. Uno scaling tempestivo fa sì che il carico sostenuto dall'EVS cali, contrastando il decadimento di prestazioni.

In figura 6.7 sono presenti i grafici relativi a questo caso. Le prestazioni sono eccellenti: dalla funzione di ripartizione in figura 6.7a emerge che i ritardi sono minimi per tutti i valori di intensità grazie al numero ridotto di CAM in arrivo. Nel 80% dei casi, il tempo di elaborazione non eccede 0,5 ms. Secondo la figura 6.7b, nel 99,9% dei casi il ritardo di elaborazione ha durata inferiore a 1,5 ms, ben al di sotto del limite dei 5 ms, mentre il tempo medio è pari a soli 0,3 ms.

In confronto al caso con istanza singola, si rilevano prestazioni pressoché doppie. Si stima che l'intensità veicolare complessivamente sostenibile dalla coppia di EVS attivi in parallelo corrisponda a $\Lambda = 4$, per un totale di 275 veicoli complessivamente monitorati dal sistema. Per quanto riguarda invece le intensità esaminate, sebbene il ritardo si dimezzi, non ci si aspettano grandi differenze al livello del sistema ICA completo: la differenza tra i tempi di elaborazione, tenendo in considerazione i ritardi dell'interfaccia radio nell'ordine delle decine di millisecondi, è trascurabile.

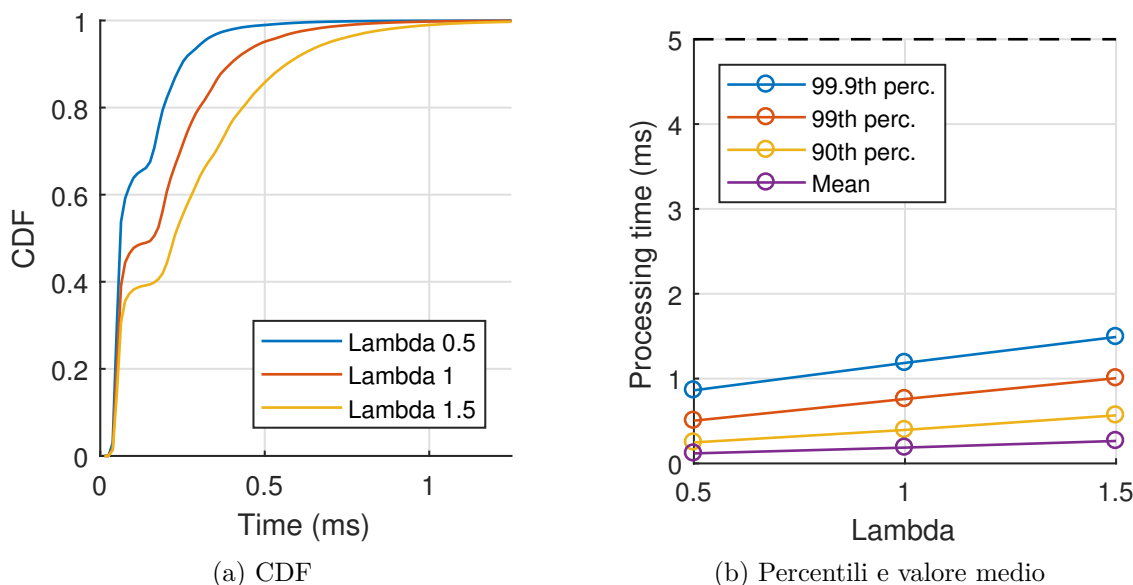


Figura 6.7: Ritardo di elaborazione dell'EVS nel testbed Openstack con 2 EVS in esecuzione parallela

6.2.3 DENM Decider

Il DENM Decider si limita a codificare ed inoltrare i DENM, pertanto il lavoro che compie è decisamente meno gravoso rispetto a quello dell'EVS. Ciò nonostante, poichè si frappone tra l'EVS e i Vehicle Simulator, è importante verificare che non introduca un ritardo eccessivo. In figura 6.8 sono riportate le funzioni di ripartizione empiriche dei tempi di elaborazione e trasmissione del DD di ciascuno dei testbed.

Il DD del testbed MEC (figura 6.8a) ha un comportamento insolito: impiega più tempo ad elaborare i DENM per basse intensità veicolari anzichè, come ci si aspetterebbe, avere un ritardo superiore per le intensità più elevate. Questa singolarità, di cui non è stato possibile determinare la causa, si accentua particolarmente tra $\Lambda = 0,5$ e $\Lambda = 1$. In ogni caso, anche nella condizione peggiore, il DENM Decider è in grado di portare a termine il suo lavoro entro un tempo di 2,5 ms.

Nel testbed Openstack, secondo il grafico in figura 6.8b, i tempi di elaborazione sono particolarmente ridotti. Per tutte i valori di intensità esaminati, in quattro casi su cinque il ritardo di elaborazione è inferiore a 0,25 ms, il che rende il componente totalmente trasparente al ritardo end-to-end complessivo. Per quanto riguarda le elaborazioni più lente, si rimane sempre al di sotto del millisecondo. A testimonianza del fatto che il DENM Decider svolga un compito elementare, si registrano prestazioni comparabili per tutti i valori di intensità veicolare esaminati.

Il DENM Decider nel testbed Openstack si dimostra dalle due alle tre volte più veloce della controparte MEC. In ogni caso, anche le prestazioni peggiori del DD MEC sono sufficienti a non introdurre un ritardo eccessivo sull'inoltro dei DENM ai veicoli, pertanto non ci si aspettano grandi differenze per quanto riguarda le prestazioni complessive.

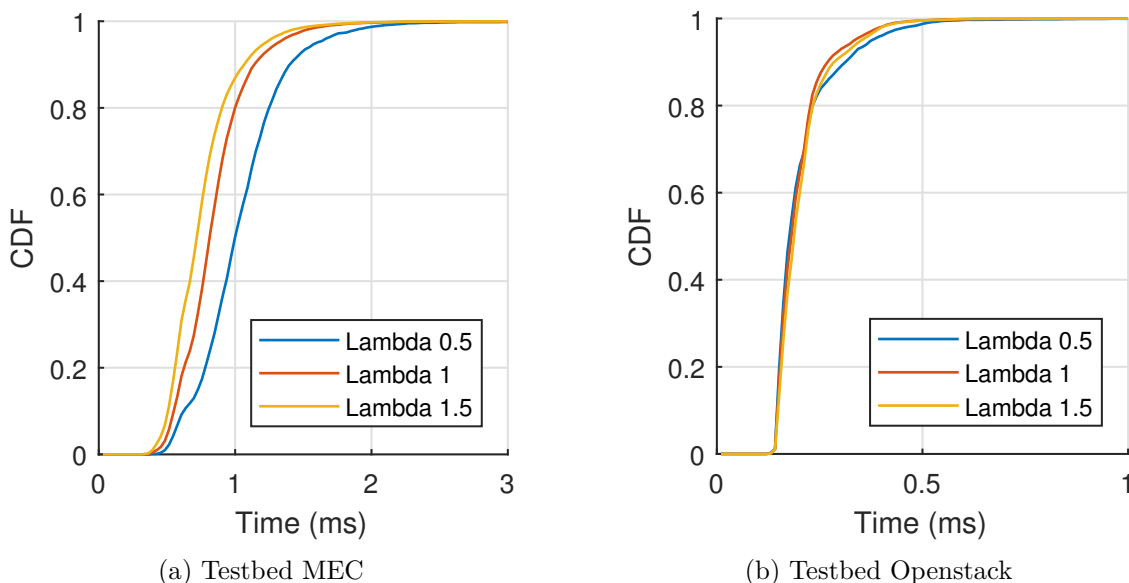


Figura 6.8: CDF del ritardo di elaborazione del DENM Decider per intensità di traffico crescenti

6.2.4 Latenza End to End

La latenza end-to-end è la più importante tra le metriche di basso livello finora analizzate.

Il ritardo E2E esprime il tempo che occorre ad un veicolo che sta rischiando una collisione a ricevere il relativo DENM, a partire dal momento in cui il veicolo stesso invia il CAM con cui l'EVS ha rilevato la probabile collisione. Quindi è la somma dei ritardi di trasmissione ed elaborazione di tutti i componenti del sistema ICA, di cui una parte è già stata analizzata nei sottoparagrafi precedenti.

La latenza end-to-end è fortemente correlata con la capacità di prevenzione delle collisioni del sistema. Infatti, con una bassa latenza, dato che il messaggio di allarme di collisione imminente arriva prontamente al conducente, è più probabile che quest'ultimo sia in grado di evitare l'incidente rispetto alla situazione in cui, a causa di un elevato ritardo, il messaggio di allarme arriva più tardi. Per questo motivo, è fondamentale ridurre il più possibile la latenze E2E.

Per il testbed MEC, sono stati studiati i ritardi end-to-end con tre diverse configurazioni della rete di accesso, secondo quanto già visto nel capitolo 4:

- OpenAirInterface, implementazione di una RAN LTE standard direttamente collegata al sistema ICA;
- NetEm Edge, che simula una rete di accesso LTE ed il nodo di elaborazione ICA posizionato in prossimità dell'intersezione;
- NetEm Cloud, che riproduce il caso di una rete di accesso LTE che fa da tramite per il nodo di calcolo ICA posizionato in un datacenter cloud.

In figura 6.9 sono riportate le funzioni di ripartizione empiriche ottenute dai test sul testbed MEC. In generale il comportamento è buono, non essendoci evidenti disparità di

prestazioni al variare del carico. Le differenze tra l'interfaccia OAI (fig. 6.9a) e NetEm Edge (fig. 6.9b) sono minime: in entrambi i casi il ritardo E2E non supera i 130 ms, con più del 80% dei casi al di sotto dei 100 ms. Il ritardo minimo si attesta a soli 30 ms, che tuttavia copre solo il 20% dei casi. Tra i due, NetEm offre prestazioni leggermente migliori, come ci si aspetta dal momento che è modellato su una distribuzione normale e quindi non è soggetto ai casi limite che invece spingono OAI a ritardare eccessivamente una ristretta percentuale di pacchetti. Discorso a parte merita la configurazione con NetEm Cloud (fig. 6.9c). In questo caso, l'andamento dei ritardi è una traslazione in avanti nel tempo di circa 45 ms dei ritardi visti con NetEm Edge, il che soddisfa le aspettative: in aggiunta all'accesso, è simulato un ulteriore segmento di rete di trasporto, la cui costanza in fatto di tempi di attraversamento è ben superiore alla rete di accesso radio.

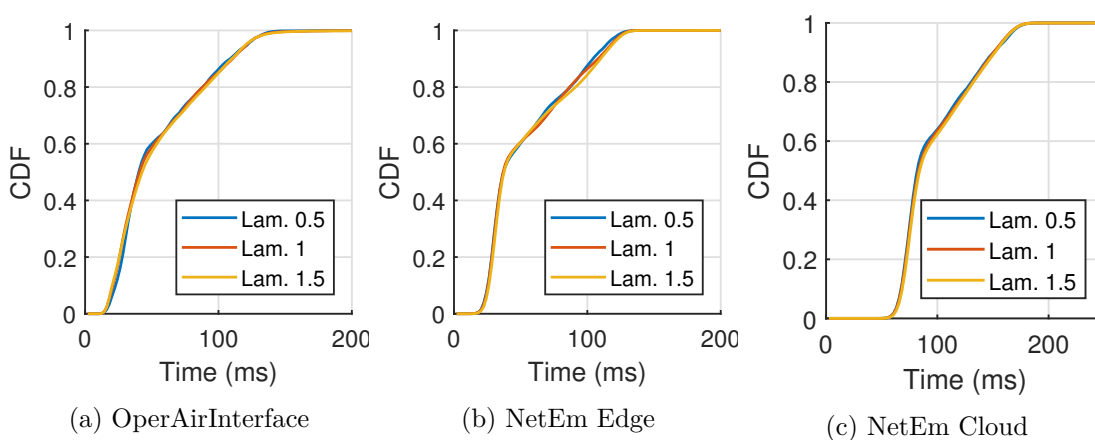


Figura 6.9: CDF del ritardo E2E del testbed MEC per intensità di traffico crescenti

Per quanto riguarda il testbed Openstack, i risultati sono riportati in figura 6.10. L'interfaccia OAI non si comporta allo stesso modo di quanto già visto nel testbed MEC: in questo caso (fig. 6.10a), si può notare una disparità dei ritardi al variare dell'intensità veicolare. Questa disparità non si nota nella configurazione con NetEm (fig. 6.10b), che ricalca fedelmente quanto visto nel testbed MEC, pertanto si può imputare totalmente all'interfaccia radio.

La media del ritardo E2E rimane costante su un valore di 55 ms per ogni configurazione, fatta eccezione per quella con NetEm Cloud, per cui la media è di 98 ms. La rete di trasporto aggiunge un ritardo medio al RTT pari a 43 ms, che corrisponde al 180% del ritardo della sola rete di accesso LTE di test, il cui ritardo medio è di circa 24 ms.

Per quanto riguarda i casi limite, che davvero dimostrano l'affidabilità del sistema, i risultati sono presentati in figura 6.11, in cui è rappresentato il 99,9-esimo percentile del ritardo E2E.

Quando viene utilizzata OpenAirInterface, l'intensità veicolare influisce sulle prestazioni, aggiungendo un ritardo di 25-30 ms a $\lambda = 1,5$ rispetto a $\lambda = 0,5$. Con le reti di accesso simulate, in modo simile a quanto visto nelle figure 6.9 e 6.10, l'intensità del traffico non cambia le caratteristiche del ritardo. In questo caso, i testbed MEC e Openstack si comportano in modo estremamente simile, mostrando punte di ritardo di circa 135 ms, in configurazione Edge, e di 185 ms in configurazione Cloud.

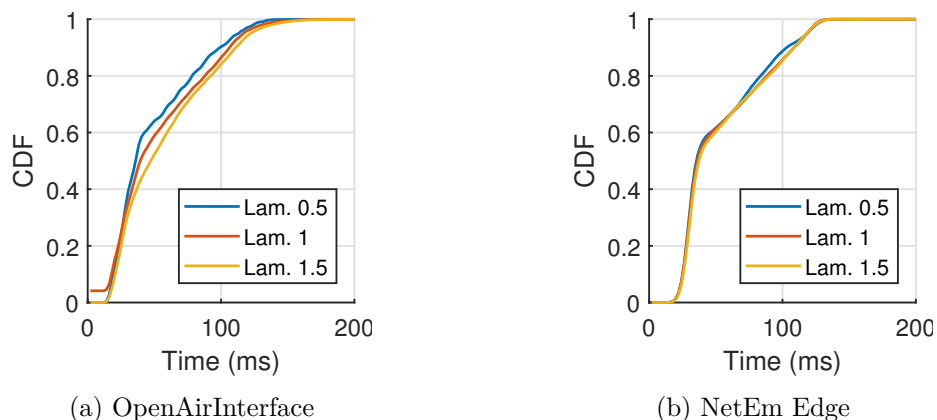


Figura 6.10: CDF del ritardo E2E del testbed Openstack per diversi livelli di traffico

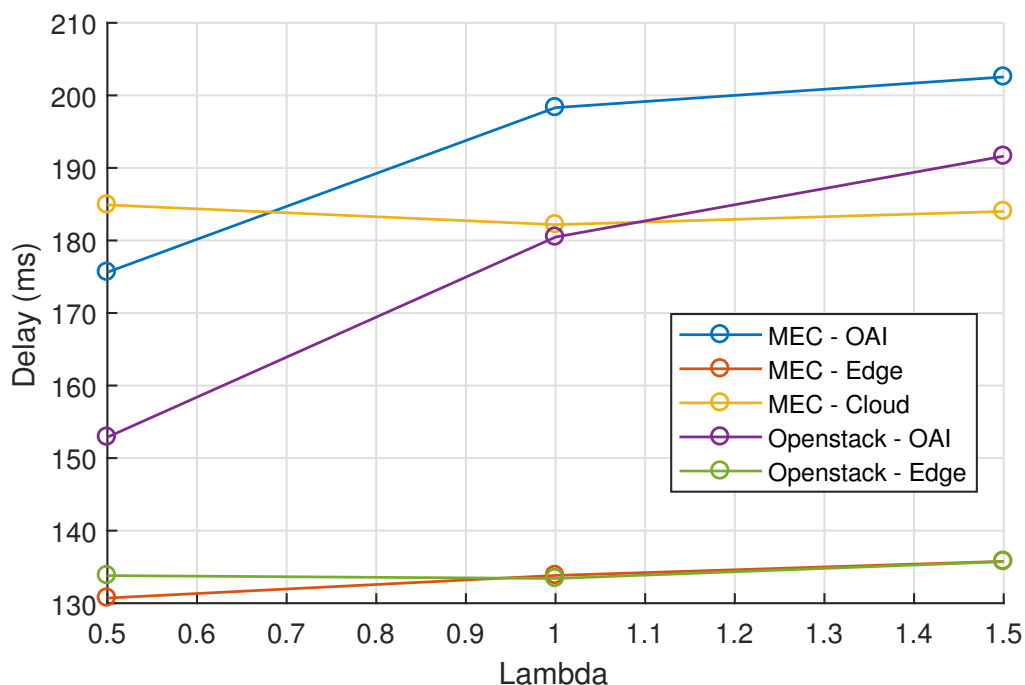


Figura 6.11: 99,9-esimo percentile del ritardo E2E per le diverse configurazioni di test

6.3 Affidabilità

L'affidabilità è un parametro fondamentale da considerare quando si ha a che fare con applicazioni critiche come la sicurezza stradale. Un pacchetto non consegnato può avere conseguenze ben peggiori di uno consegnato in ritardo, ed è per questo che solitamente, in situazioni in cui è importante l'affidabilità di trasmissione, si implementano soluzioni basate su segnali di Acknowledge, per segnalare la corretta ricezione di un pacchetto, e ritrasmissione, nel caso un pacchetto venga perso. In applicazioni real time, in cui il tempo di trasmissione fa la differenza tra successo e fallimento, non sempre questa affermazione

è vera.

Ci sono situazioni, al contrario, in cui un pacchetto consegnato in ritardo provoca effetti equivalenti, se non peggiori, di uno consegnato in ritardo. Il sistema ICA ricade in questo caso. L'EVS non assume che i veicoli abbiano l'orologio di sistema correttamente sincronizzato, pertanto è esso stesso che appone un timestamp ai CAM in arrivo. Questo implica, a differenza di quanto avverrebbe se fossero i veicoli ad apporre l'orario nell'istante in cui trasmettono il beacon, che il ritardo di trasmissione di un CAM equivale al calcolo di una posizione scorretta del veicolo che lo ha generato da parte dell'EVS. Con TCP, quando viene trasmesso un pacchetto, è fatto partire un timer. Se allo scadere del timer non è stato ricevuto un ACK di conferma, si assume che il pacchetto sia stato perso e pertanto viene ritrasmesso. A seconda della lunghezza del timer si rischia quindi che il pacchetto perso venga ritrasmesso molto dopo rispetto a quando è stato generato. Dato che i CAM sono generati periodicamente, per avere i dati il più possibile corretti ed aggiornati, piuttosto che far affidamento sulla funzione di ritrasmissione automatica, è meglio aspettare la trasmissione successiva.

La comunicazione interna tra le macchine virtuali del testbed, per comodità di gestione dei pacchetti, avviene tramite TCP, potendo assumere che in questa situazione le perdite siano nulle. La comunicazione sul link radio avviene invece su UDP.

6.3.1 CAM ricevuti

Nel caso specifico delle simulazioni svolte con il testbed, dato che i veicoli si muovono a velocità costante, la perdita di un CAM non costituisce un problema: l'EVS riesce a prevedere la posizione corretta del veicolo basandosi sul CAM precedente. I valori riportati di seguito in tabella 6.1 sono dati dal numero di CAM ricevuti dal EVS rapportati al numero di CAM trasmessi dai VehicleSimulator.

Configurazione	Lambda		
	0,5	1	1,5
OAI	99,58%	99,38%	99,32%
LTE NetEm	99,99%	99,95%	99,91%

Tabella 6.1: CAM correttamente ricevuti

Sia con il la rete di accesso OAI che con NetEm, che pur non ha perdite, si rileva che l'EVS non si vede recapitare tutti i pacchetti a lui destinati. Le perdite aumentano al crescere della densità di veicoli e quindi dei CAM generati. Dato che NetEm non è stato impostato per perdere pacchetti, si può solo assumere che la perdita avvenga all'interno del CIM o nel collegamento tra CIM ed EVS.

6.3.2 DENM ricevuti

La perdita di un DENM costituisce un problema, specialmente se è il primo inviato per un particolare evento. Un veicolo che non riceve il DENM destinato ad avvisarlo di una collisione imminente, aspettando il DENM successivo, rischia di non avere il tempo sufficiente

ad evitarla. È fondamentale che la trasmissione dei DENM sia altamente affidabile. I risultati in tabella 6.2 sono stati presi rapportando il numero di DENM ricevuti dai Vehicle Simulator con il numero di DENM inviati dall'EVS.

Configurazione	Lambda		
	0,5	1	1,5
OAI	99,99%	99,84%	99,96%
LTE NetEm	100%	100%	100%

Tabella 6.2: DENM correttamente ricevuti

Il downlink LTE di OpenAirInterface è sicuramente la causa della differenza del numero di DENM correttamente ricevuti dato che sostituendolo con NetEm le perdite si azzerano.

6.4 Prestazioni complessive

In questo paragrafo vengono presentati i risultati di alto livello del sistema ICA in esame. Prima è opportuno fare una premessa per spiegare con quali basi viene deciso che una collisione venga rilevata in tempo o meno per essere evitata. Le considerazioni espresse di seguito sono tratte dal paper di Malinverno et al. già citato [27].

Siano definiti:

- T_A , come il tempo che il conducente del veicolo ha a disposizione per evitare la collisione, che quindi è l'intervallo di tempo compreso tra quando inizia la frenata e quando avviene l'impatto;
- T_B , come il tempo di cui il veicolo necessita per arrestarsi, nella realtà determinato da numerosi fattori come lo stato delle gomme e dei freni, la temperatura e l'umidità ambientale, le condizioni del manto stradale, ma alla fine riconducibile alla velocità del veicolo e alla sua massima decelerazione. Per le simulazioni svolte abbiamo considerato una decelerazione massima di $7,5 \text{ m/s}^2$. Dato che la velocità dei veicoli è costante e pari a 50 km/h , si ottiene $T_B = 1,85 \text{ s}$.

Una collisione è considerata come non rilevata in tempo se:

$$T_A < T_B$$

dove il tempo T_A dipende dai termini T_{FA} , T_D e T_H secondo la relazione:

$$T_A = T_{FA} - T_D - T_H .$$

Di seguito è riportata la descrizione dei termini, di cui si può trovare la presentazione grafica in figura 6.12:

- T_D : è l'intervallo di tempo tra l'istante in cui una collisione imminente è rilevata dall'EVS all'istante in cui è l'allarme è mostrato al conducente. Include un tempo di trasmissione T_{TX} e un tempo di processamento T_P . Il tempo di trasmissione è

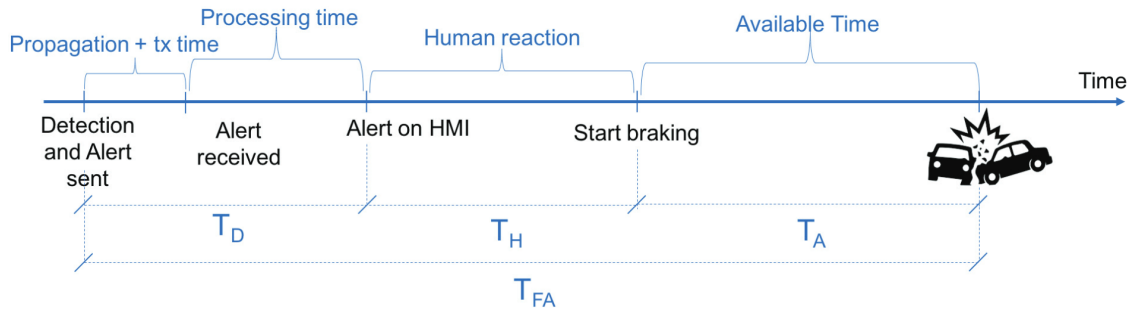


Figura 6.12: Cronologia degli eventi che intercorrono tra la rilevazione di una collisione e l’impatto tra veicoli

il tempo che il pacchetto contenente l’avviso di collisione impiega ad essere inviato al DENM Decider, ad essere codificato come DENM, ad essere inoltrato all’eNodeB e ad essere recapitato al modem del veicolo coinvolto. Il tempo di processamento è l’ammontare di tempo che intercorre tra l’istante in cui il pacchetto è ricevuto e il momento in cui il programma dedicato al C-ITS in esecuzione sulla vettura impiega a mostrare l’allarme al conducente tramite HMI predisposta. Il tempo di processamento è assunto pari a $T_P = 400$ ms;

- T_H : rappresenta il tempo che il conducente impiega a percepire e capire l’allarme, e ad azionare il pedale del freno. Questo parametro è strettamente correlato ai tempi di reazione umana, che variano principalmente in funzione dell’età. Uno dei vantaggi della guida autonoma è che questo valore può essere considerato trascurabile. Per i test, si suppone che l’auto sia controllata da un conducente umano e si assume di conseguenza $T_H = 1$ s;
- T_{FA} : è il tempo che intercorre tra la generazione del primo allarme di possibile collisione e il momento in cui la collisione avverrebbe se non fossero prese misure atte ad evitarla.

Ai fini della simulazione, $T_{VS} = T_{FA} - T_{TX}$ è il tempo disponibile ad evitare la collisione dall’istante in cui l’allarme è ricevuto dal Vehicle Simulator. La collisione viene considerata evitata se:

$$T_{VS} > 3,25 \text{ s.}$$

6.4.1 Veri positivi

I veri positivi rappresentano le collisioni correttamente rilevate dall’EVS e le quali segnalazioni giungono ai veicoli coinvolti. Per determinare se una collisione rilevata dal sistema costituisca un vero positivo, vengono esaminate le collisioni registrate da SUMO. Se c’è corrispondenza, la collisione viene considerata correttamente rilevata. Trattandosi di un servizio di sicurezza critico, è opportuno che i veri positivi siano quanto più possibile prossimi al 100% delle collisioni avvenute in SUMO.

I risultati dei test sono rappresentati in figura 6.13.

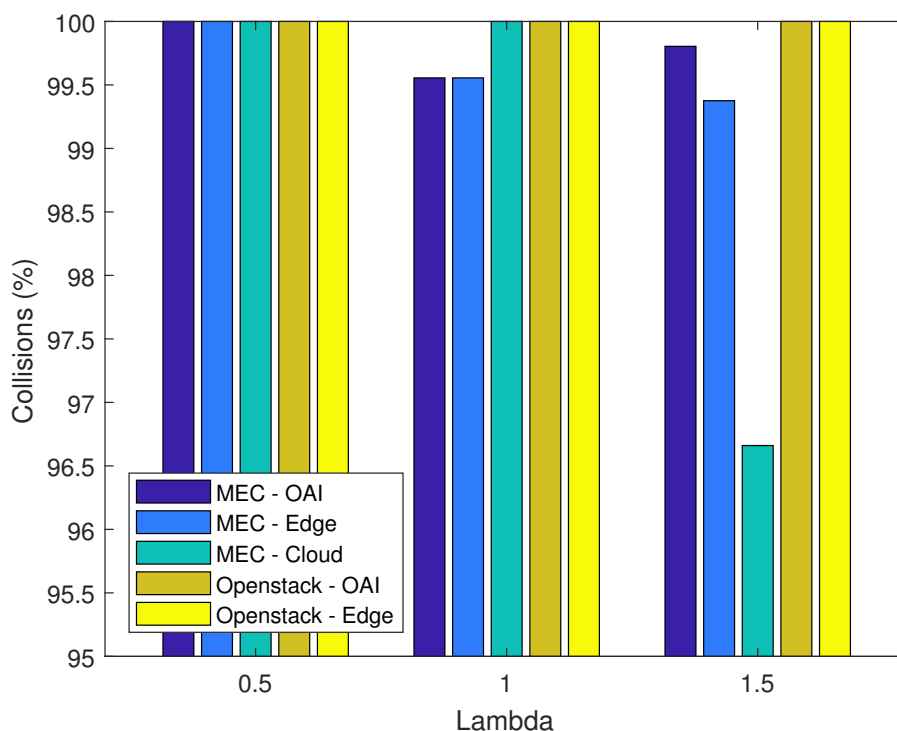


Figura 6.13: Collisioni rilevate

Per intensità $\Lambda = 0,5$ il 100% delle collisioni vengono rilevate in ogni configurazione. Ad intensità crescenti, la percentuale di collisioni rilevate cala progressivamente, con un minimo del 96,66% registrato a $\Lambda = 1,5$ nel testbed MEC in configurazione cloud. La latenza della rete di accesso è determinante per l'ottenimento di alta affidabilità; in tal senso, l'infrastruttura MEC si conferma la soluzione privilegiata per garantire prestazioni elevate. In ogni configurazione, le soluzioni basate su MEC rilevano più del 99% delle collisioni. In particolare, il testbed Openstack è sempre in grado di rilevare tutte le collisioni. Non si registrano differenze sostanziali tra le prestazioni registrate con OAI e quelle rilevate con NetEm Edge.

Affinchè la collisione venga evitata, è importante che l'allarme venga ricevuto tempestivamente dai veicoli coinvolti. Un DENM ricevuto in ritardo, sebbene possa aiutare a ridurre l'intensità dell'impatto e quindi limitarne gli effetti, è insufficiente¹ per impedire danni a persone e cose. In figura 6.14 sono presentati i risultati relativi alle collisioni evitate poiché segnalate in tempo.

Si registra un peggioramento delle precisioni di rilevamento rispetto a quanto visto precedentemente. Il testbed MEC abbinato a OAI mostra un decadimento di prestazioni all'aumentare di intensità fino a toccare un minimo di 93% di collisioni rilevate in tempo.

¹Per evitare la collisione sarebbe sufficiente che anche un solo veicolo riceva il DENM. Oltretutto, probabilmente non sarebbe necessario l'arresto completo ma basterebbe una frenata tale da scartare l'altro veicolo.

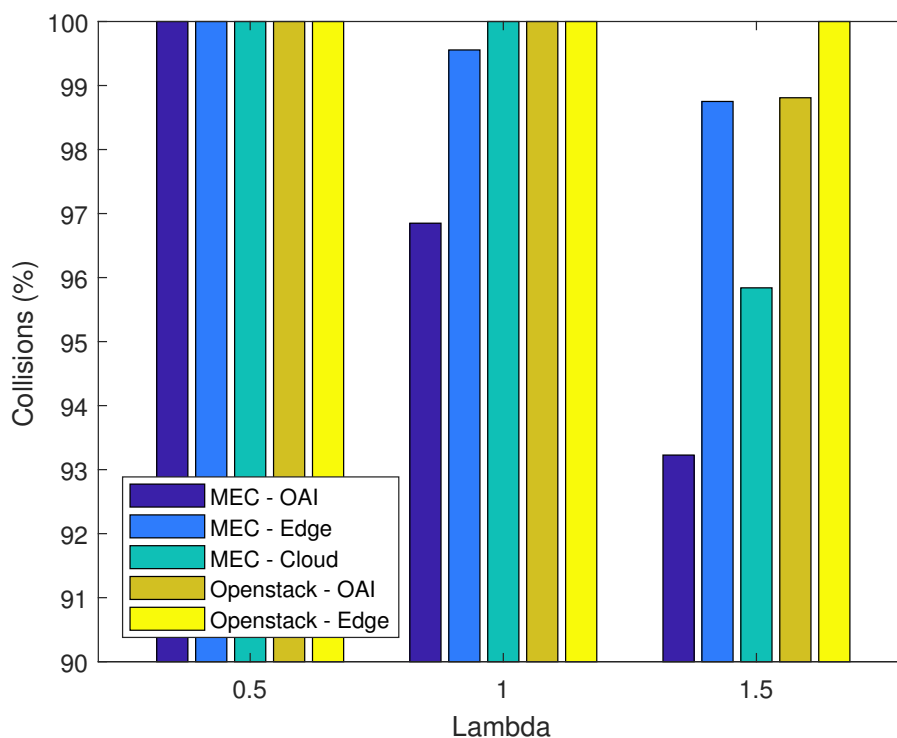


Figura 6.14: Collisioni rilevate in tempo

Con NetEm Edge invece le collisioni rilevate in ritardo sono minime, solo lo 0,62% delle totali. Poco di più sono quelle registrate con NetEm Cloud, pari al 0,82%, che però soffre delle collisioni non rilevate viste in precedenza. OAI dimostra scarse prestazioni anche nel testbed Openstack. Del 100% di collisioni rilevate, 1,2% sono rilevate in ritardo, mentre con NetEm il 100% sono rilevate in tempo.

Se si vanno ad analizzare le differenza dei tempi $T_A - T_B$, si ricavano dati concordi con quanto visto finora. Il tempo $T_A - T_B$ rappresenta il tempo che intercorre tra quando il veicolo si arresta a seguito della ricezione di un DENM di allarme a quando si sarebbe verificata la collisione se non fosse stata segnalata. Un tempo positivo implica che la collisione è stata evitata, un tempo negativo significa che il veicolo non ha fatto in tempo ad arrestarsi e quindi non ha evitato la collisione. In figura 6.15 è riportata la media dello 0,1% dei tempi $T_A - T_B$ più bassi registrati in ogni prova.

Solo il testbed openstack riesce a garantire il 99,9% dei tempi positivi, con la configurazione OAI molto vicina allo 0. Il testbed MEC, pur avendo prestazioni comparabili per $Lambda = 0,5$, al crescere dell'intensità mostra una sensibile diminuzione di affidabilità. Per $Lambda = 1,5$, i DENM più lenti arrivano con ritardo fino a 505 ms nella configurazione NetEm Edge e fino a 680ms con NetEm Cloud.

6.4.2 Falsi positivi

I falsi positivi sono costituiti dalle collisioni che sono state rilevate dal sistema ma che non hanno corrispondenza con quanto registrato da SUMO durante la simulazione. Ci si

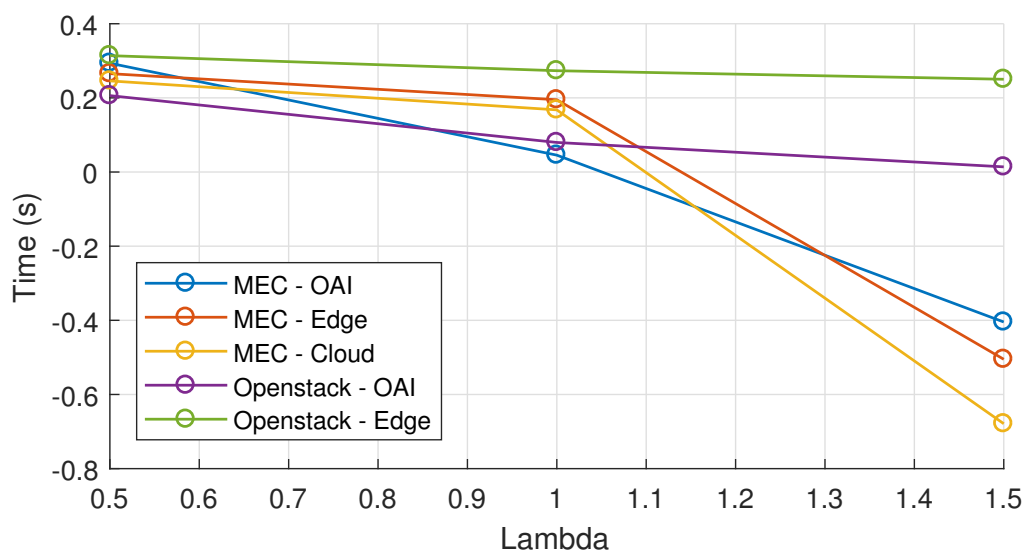


Figura 6.15: Millesimo più basso dei tempi $T_A - T_B$ osservati

aspetta un gran numero di falsi positivi dato come è stato pensato l’algoritmo di rilevamento delle collisioni. Infatti l’algoritmo, non considerando la superficie occupata dagli oggetti, è tarato per considerare una collisione un qualunque avvicinamento della punta dei veicoli a meno di $s2c_t = 3,5$ m. Questo implica che, allo scopo di ridurre il più possibile le collisioni non rilevate (falsi negativi), vengono accettate segnalazioni di un grande numero di collisioni che nella simulazione SUMO non si verificano. Se si diminuisse la soglia $s2c_t$, i falsi positivi diminuirebbero a spese dei falsi negativi, ma data la criticità dell’applicazione non si ritiene opportuno accettare un simile compromesso. In figura 6.16 sono presentati i risultati relativi ai dati raccolti.

Nessuna configurazione è esente da falsi positivi. Il testbed Openstack si conferma nuovamente come il migliore, offrendo il minor numero di falsi positivi rispetto alla controparte MEC. Per l’intensità più bassa, Openstack fa registrare circa 8% di falsi positivi sul totale di collisioni rilevate, valore che sale fino al 14% per $Lambda = 1,5$, mentre il testbed MEC parte da un minimo del 13% fino ad arrivare ad un massimo di 18%. Non sono evidenti sensibili peggioramenti prestazionali usando la rete di accesso NetEm Cloud rispetto a quanto si ottiene con OpenAirInterface, anche NetEm Edge non offre consistentemente le prestazioni migliori.

Sebbene un gran numero di falsi positivi non sia considerabile una condizione ottimale, si tenga presente che un falso positivo è spesso associato comunque ad una situazione di rischio. Sebbene non avvengano collisioni, un avvicinamento esagerato tra auto ad alta velocità è da considerarsi un evento pericoloso e dunque meritevole di attenzione da parte dei conducenti. Ciò non toglie che sarebbe opportuno discriminare situazioni di rischio da collisioni certe.

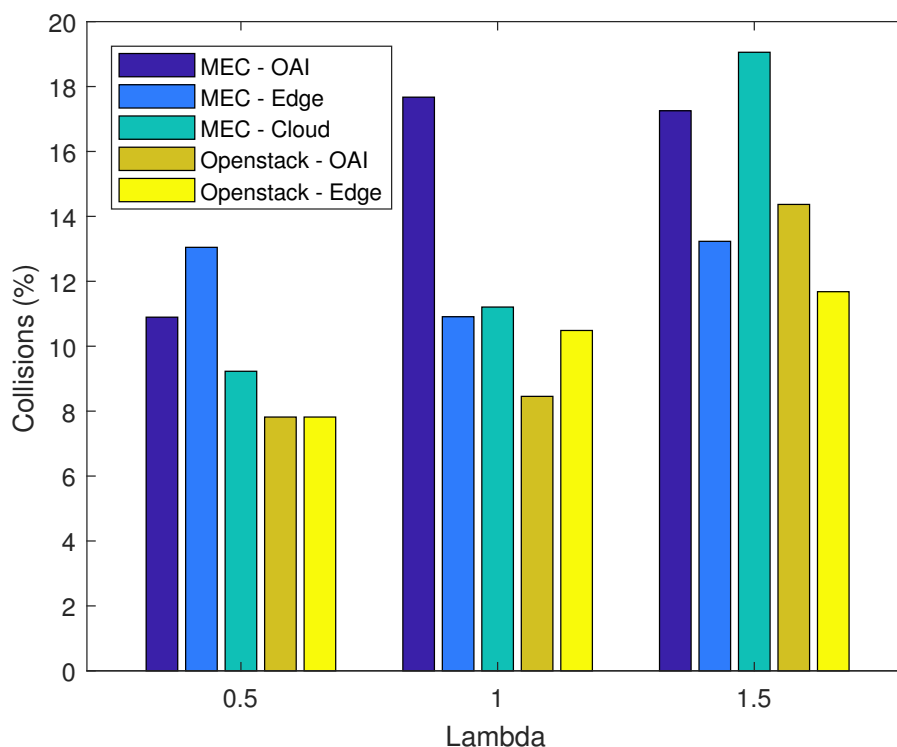


Figura 6.16: Falsi positivi

6.4.3 Falsi negativi

I falsi negativi rappresentano le collisioni che il sistema non è stato in grado di prevedere. Mentre nel caso di una collisione segnalata in ritardo il conducente ha comunque modo di mitigare parzialmente il pericolo, con un falso negativo il conducente rimane totalmente all'oscuro del rischio. Trattandosi di un'applicazione di sicurezza automotive, i falsi negativi dovrebbero essere nulli o trascurabili per certificare l'affidabilità del sistema. In figura 6.17 sono riportati i risultati ottenuti.

Alla più bassa intensità di traffico testata, nessuna collisione viene ignorata dal sistema. Al crescere del numero di veicoli, la situazione cambia. Il testbed Openstack continua a mostrare prestazioni perfette, invece il testbed MEC sia a $\Lambda = 1$ che $\Lambda = 1,5$ non riesce a rilevare il 0,5% circa delle collisioni. La situazione con la configurazione con NetEm Cloud peggiora ulteriormente: si registrano ben il 3,34% di collisioni non rilevate. Per aumentare la precisione si può incrementare ulteriormente la soglia $s2c_t$, a scapito però di un numero maggiore di falsi positivi, che tuttavia non risolve la debolezza intrinseca dell'algoritmo. Il problema è che l'EVS registra la posizione dei veicoli secondo quanto contenuto nel CAM non tenendo conto della latenza di trasmissione. Di conseguenza, all'aumentare del ritardo tra il momento in cui i CAM sono generati al momento in cui sono processati dall'EVS, diminuisce la precisione delle traiettorie calcolate e quindi aumentano i falsi negativi. La soluzione è quella prospettata nel paragrafo 6.3: l'EVS dovrebbe considerare il timestamp contenuto sul CAM per il calcolo delle traiettorie anziché apporlo esso stesso nel momento in cui gli arriva.

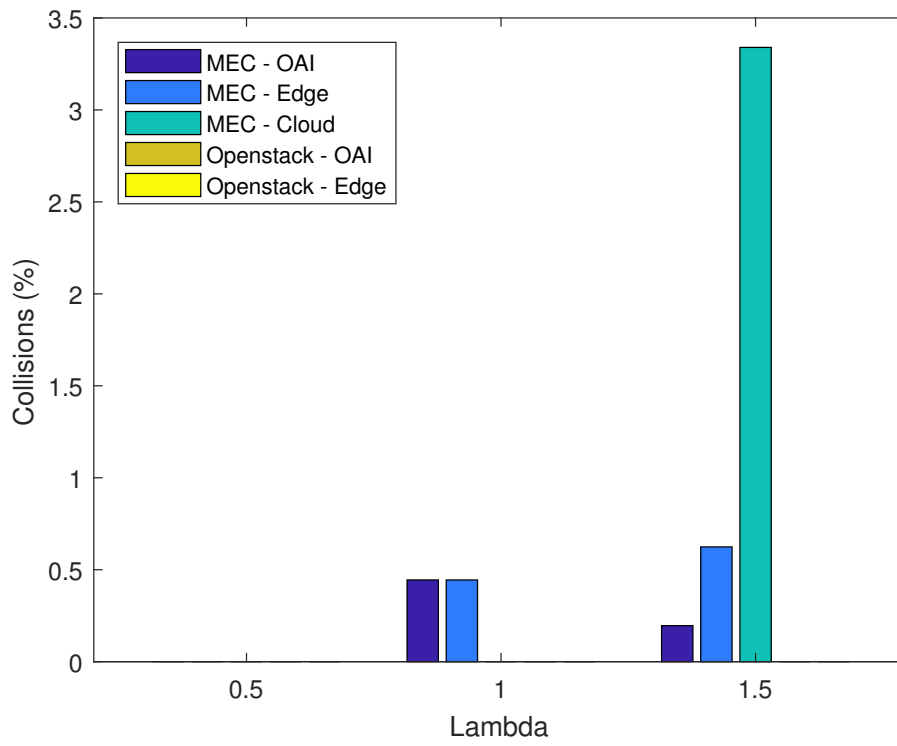


Figura 6.17: Falsi negativi

Capitolo 7

Conclusioni

Gli elementi chiave delle reti mobili 5G come New Radio, Multi-access Edge Computing (MEC) e Network Slicing consentiranno alle aziende verticali di esplorare nuovi casi d'uso, grazie all'alta velocità di trasferimento dei dati, bassa latenza, elevata affidabilità e flessibilità. Questi sono i requisiti che contraddistinguono molte delle applicazioni che progetti come 5G-TRANSFORMER stanno studiando, e la presente tesi ne è un esempio. Per quanto riguarda il settore automobilistico, la prevenzione di incidenti mediante monitoraggio delle intersezioni stradali rappresenta uno dei casi d'uso di maggior rilievo e i risultati ottenuti confermano quali benefici apporti l'uso di tecnologie di quinta generazione.

L'infrastruttura di Edge Computing, grazie alla latenza ridotta, consente una tempestiva e precisa rilevazione delle collisioni imminenti, mentre la piattaforma 5GT risponde alle esigenze di affidabilità e disponibilità del servizio. Si è mostrato come il MEC apporti un vantaggio sensibile all'applicazione ICA rispetto al Cloud Computing, consentendo, grazie al minor ritardo di trasmissione, prestazioni generalmente superiori. Nel primo caso, nella condizione di traffico intenso, il 99% delle collisioni simulate sono correttamente rilevate in tempo, mentre nella configurazione Cloud solo il 96% delle collisioni sono evitate. Lo scaling automatico reso possibile dalla piattaforma 5GT, con l'ausilio di un Service Orchestrator a più di 500km di distanza, ha permesso di contenere i tempi di elaborazione durante le situazioni di maggior stress, garantendo il rispetto dei requisiti di funzionamento dell'EVS. A parità di condizioni, due istanze di EVS operanti in parallelo sono mediamente il doppio più veloci di un EVS singolo e pertanto consentono, a parità di prestazioni, di monitorare il doppio dei veicoli.

Sebbene l'attività di ricerca per la tesi sia conclusa, ci sono modifiche al sistema che ancora possono essere studiate per migliorarne le prestazioni. Per quanto riguarda l'EVS, si potrebbero analizzare nuove soluzioni che consentano di operare al meglio nelle condizioni di latenza di rete e traffico stradale più sfavorevoli. Per rimediare alla perdita di precisione introdotta dal ritardo di trasmissione della rete, l'algoritmo di rilevamento delle collisioni potrebbe far affidamento sul timestamp inserito dal Vehicle Simulator all'interno dei CAM trasmessi, anziché usare quello apposto dall'EVS alla ricezione dei pacchetti. In questo modo, si svincolerebbe la precisione con cui la posizione dei veicoli viene registrata dall'EVS dal ritardo con cui i CAM vengono recapitati. Un ulteriore perfezionamento che può ancora essere studiato consiste nel rilevamento delle collisioni basato sulla intersezione tra sagome, anziché sulla distanza tra punti. Tenendo conto dell'area realmente occupata dai

veicoli, diminuirebbero il numero di falsi positivi registrati e aumenterebbe la precisione di rilevamento.

Il testbed si è dimostrato un valido punto di partenza per lo sviluppo di un'applicazione affidabile di sicurezza automotive. L'Extended Virtual Sensing si dimostrerebbe particolarmente utile nelle situazioni di visibilità ridotta. Condizioni meteorologiche avverse o ostacoli che ostruiscono la linea visiva possono limitare la capacità del conducente e degli Advanced Driver Assistance System (ADAS) di percepire l'ambiente che circonda il veicolo. Le informazioni aggiuntive fornite dall'EVS in tali condizioni di pericolo costituirebbero un importante ausilio per scongiurare incidenti e garantire così la sicurezza delle persone a bordo.

Bibliografia

- [1] ITU-R M.2083-0. *IMT Vision - "Framework and overall objectives of the future development of IMT for 2020 and beyond"*. International Telecommunication Union, 29 set. 2015. URL: https://itu.int/dms_pubrec/itu-r/rec/m/R-REC-M.2083-0-201509-I!!PDF-E.pdf.
- [2] ETSI GS MEC 003 V2.1.1. *Multi-access Edge Computing (MEC); Framework and Reference Architecture*. European Telecommunications Standards Institute, gen. 2019. URL: https://www.etsi.org/deliver/etsi_gs/MEC/001_099/003/02.01.01_60/gs_MEC003v020101p.pdf.
- [3] Sami Kekki et al. *MEC in 5G networks*. ETSI, giu. 2018. URL: https://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp28_mec_in_5G_FINAL.pdf.
- [4] ETSI TR 102 638 V1.1.1. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Definitions*. Giu. 2009. URL: https://www.etsi.org/deliver/etsi_tr/102600_102699/102638/01.01.01_60/tr_102638v010101p.pdf.
- [5] ETSI EN 302 637-2 V1.4.1. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service*. Apr. 2019. URL: https://www.etsi.org/deliver/etsi_en/302600_302699/30263702/01.04.01_60/en_30263702v010401p.pdf.
- [6] ETSI EN 302 637-3 V1.3.1. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service*. Apr. 2019. URL: https://www.etsi.org/deliver/etsi_en/302600_302699/30263703/01.03.01_60/en_30263703v010301p.pdf.
- [7] 5G-TRANSFORMER Project. *Report on vertical requirements and use cases*. URL: http://5g-transformer.eu/wp-content/uploads/2017/12/Report_on_vertical_requirements_and_use_cases.pdf.
- [8] 5G-TRANSFORMER Project. *5G-TRANSFORMER initial system design*. URL: http://5g-transformer.eu/wp-content/uploads/2018/05/D1.2_5G-TRANSFORMER_Initial_System_Design.pdf.
- [9] World Health Organization. *Global status report on road safety 2018*. Dic. 2018. URL: https://www.who.int/violence_injury_prevention/road_safety_status/2018/en/.

-
- [10] European Commission. *A European strategy on Cooperative Intelligent Transport Systems, a milestone towards cooperative, connected and automated mobility*. 30 Nov. 2016. URL: https://ec.europa.eu/transport/sites/transport/files/com20160766_en.pdf.
- [11] Niels Peter Skov Andersen. *Towards accidents free driving*. Car 2 Car Communication Consortium, 11 apr. 2016. URL: https://docbox.etsi.org/Workshop/2016/20160421_5G_FROM_MYTH_TO_REALITY/SESSION_C_MASSIVE_M2M/TOWARDS_ACCIDENT_FREE_DRIVING_ANDERSEN_CAR_COM_CONSORTIUM.pdf.
- [12] ETSI EN 302 665 V1.1.1. *Intelligent Transport Systems (ITS); Communications Architecture*. Set. 2010. URL: https://www.etsi.org/deliver/etsi_en/302600_302699/302665/01.01.01_60/en_302665v010101p.pdf.
- [13] ETSI EN 302 663 V1.3.0. *Intelligent Transport Systems (ITS); ITS-G5 Access layer specification for Intelligent Transport Systems operating in the 5 GHz frequency band*. Mag. 2019. URL: https://www.etsi.org/deliver/etsi_en/302600_302699/302663/01.03.00_20/en_302663v010300a.pdf.
- [14] ETSI EN 302 636-4-1 V1.3.1. *Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality*. Ago. 2017. URL: https://www.etsi.org/deliver/etsi_en/302600_302699/3026360401/01.03.01_60/en_3026360401v010301p.pdf.
- [15] ETSI EN 302 931 V1.0.0. *Intelligent Transport Systems (ITS); Vehicular Communications; Geographical Area Definition*. Dic. 2010. URL: https://www.etsi.org/deliver/etsi_en/302900_302999/302931/01.00.00_20/en_302931v010000c.pdf.
- [16] ETSI EN 302 636-5-1 V2.1.0. *Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 5: Transport Protocols; Sub-part 1: Basic Transport Protocol*. Mag. 2017. URL: https://www.etsi.org/deliver/etsi_en/302600_302699/3026360501/02.01.00_20/en_3026360501v020100a.pdf.
- [17] ETSI TS 103 301 V1.1.1. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Facilities layer protocols and communication requirements for infrastructure services*. Nov. 2016. URL: https://www.etsi.org/deliver/etsi_ts/103300_103399/103301/01.01.01_60/ts_103301v010101p.pdf.
- [18] 5GAA. *Coexistence of C-V2X and ITS-G5 at 5.9 GHz*. 5 Apr. 2018. URL: <http://5gaa.org/wp-content/uploads/2018/10/Position-Paper-ITG5.pdf>.
- [19] «IEEE Standard for Information technology– Local and metropolitan area networks– Specific requirements– Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment 6: Wireless Access in Vehicular Environments». In: *IEEE Std 802.11p-2010 (Amendment to IEEE Std 802.11-2007 as amended by IEEE Std 802.11k-2008, IEEE Std 802.11r-2008, IEEE Std 802.11y-2008, IEEE Std 802.11n-2009, and IEEE Std 802.11w-2009)* (lug. 2010), pp. 1–51. DOI: [10.1109/IEEESTD.2010.5514475](https://doi.org/10.1109/IEEESTD.2010.5514475).

-
- [20] ETSI TS 123 285 V14.2.0. *Universal Mobile Telecommunications System (UMTS); LTE; Architecture enhancements for V2X services (3GPP TS 23.285 version 14.2.0 Release 14)*. Mag. 2017. URL: https://www.etsi.org/deliver/etsi_ts/123200_123299/123285/14.02.00_60/ts_123285v140200p.pdf.
- [21] Rafael Molina-Masegosa e Javier Gozalvez. «LTE-V for Sidelink 5G V2X Vehicular Communications: A New 5G Technology for Short-Range Vehicle-to-Everything Communications». In: *IEEE Vehicular Technology Magazine* 12.4 (dic. 2017), pp. 30–39. DOI: [10.1109/mvt.2017.2752798](https://doi.org/10.1109/mvt.2017.2752798).
- [22] A. Bazzi et al. «On the Performance of IEEE 802.11p and LTE-V2V for the Cooperative Awareness of Connected Vehicles». In: *IEEE Transactions on Vehicular Technology* 66.11 (nov. 2017), pp. 10419–10432. ISSN: 0018-9545. DOI: [10.1109/TVT.2017.2750803](https://doi.org/10.1109/TVT.2017.2750803).
- [23] ETSI TS 123 401 V8.18.0. *3rd Generation Partnership Project; Technical Specification Group Services and System Aspects; General Packet Radio Service (GPRS) enhancements for Evolved Universal Terrestrial Radio Access Network (E-UTRAN) access (Release 8)*. Apr. 2013. URL: https://www.etsi.org/deliver/etsi_ts/123400_123499/123401/08.18.00_60/ts_123401v081800p.pdf.
- [24] X. Zhou et al. «Network slicing as a service: enabling enterprises' own software-defined cellular networks». In: *IEEE Communications Magazine* 54.7 (lug. 2016), pp. 146–153. ISSN: 0163-6804. DOI: [10.1109/MCOM.2016.7509393](https://doi.org/10.1109/MCOM.2016.7509393).
- [25] Giuseppe Avino et al. «A MEC-based Extended Virtual Sensing for Automotive Services». In: *AELIT* (2019). URL: <http://hdl.handle.net/11583/2733721>.
- [26] OpenAirInterface. *5G software alliance for democratising wireless innovation*. URL: <https://www.openairinterface.org/> (visitato il 18/06/2019).
- [27] M. Malinverno et al. «Performance Analysis of C-V2I-Based Automotive Collision Avoidance». In: *2018 IEEE 19th International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*. Giu. 2018, pp. 1–9. DOI: [10.1109/WoWMoM.2018.8449772](https://doi.org/10.1109/WoWMoM.2018.8449772).