Alma Mater Studiorum · Università di Bologna

Corso di Laurea Magistrale in Ingegneria Informatica

Formazione e gestione di gruppi in reti veicolari ad-hoc per applicazioni di traffic management

Tesi di Laurea in Sistemi Mobili M

Relatore:
Prof. Ing.
Paolo Bellavista

Presentata da:
Enrico Zamagni

Correlatori:
Dott. Ing. Luca Foschini
Prof. Ing. Michela Milano

III Sessione
Anno Accademico 2012/2013

Indice

Introduzione

Negli ultimi decenni la ricerca sugli *Intelligent Transportation System* si è occupata di promuovere tecnologie finalizzate al miglioramento della sicurezza e della qualità della guida, all'incremento della funzionalità dei sistemi di trasporto e alla riduzione delle emissioni nocive. Gli studi sulla realizzazione di un'infrastruttura stradale "intelligente", cioè capace di raccogliere automaticamente informazioni sul traffico veicolare e intervenire dinamicamente su di esso per realizzare una viabilità più efficiente e sicura, si sono da tempo concentrati sull'utilizzo di reti mobili veicolari (*Vehicular Ad-hoc Network*), in quanto la possibilità di mettere in comunicazione i mezzi di trasporto attraverso semplici interfacce wireless amplierebbe enormemente le potenzialità e l'efficienza di tali applicazioni.

Il presente lavoro di tesi si inserisce nel contesto appena esposto e intende realizzare un insieme di protocolli in ambito VANET relativamente semplici ma efficaci, in grado di rilevare la presenza di veicoli in avvicinamento a un impianto semaforico e di raccogliere quelle informazioni di stato che consentano all'infrastruttura stradale di ottenere una stima il più possibile veritiera delle attuali condizioni del traffico in ingresso per ciascuna delle direzioni previste in tale punto.

La strategia adottata per rispondere a tali requisiti prevede di raccogliere i veicoli in *gruppi* durante il loro avvicinamento al centro di un incrocio. Ogni gruppo sarà costituito esclusivamente da quelle vetture che stanno percorrendo uno stesso tratto stradale e promuoverà l'intercomunicazione tra i suoi diversi membri al fine di raccogliere e integrare i dati sulla composizione del traffico locale. Il sistema realizzato cercherà di trasmettere alle singole unità

semaforiche un flusso di dati sintetico ma costante contenente le statistiche sull'ambiente circostante, in modo da consentire loro di applicare politiche dinamiche e intelligenti di controllo della viabilità.

Per poter aggregare tra loro i veicoli sono state messe a punto due tipologie di protocolli per la formazione di un gruppo e l'elezione del relativo leader: esse si differenziano in base alla strategia reattiva o proattiva utilizzata e ciascuna di esse presenta dei vantaggi e dei punti di debolezza che verranno discussi in maniera approfondita. Sono stati inoltre predisposti quei protocolli indispensabili per la gestione del ciclo di vita di un gruppo, la raccolta dei dati di interesse e il loro trasferimento verso le unità infrastrutturali.

L'architettura realizzata verrà eseguita all'interno di un ambiente urbano simulato nel quale la mobilità dei nodi di rete corrisponde a rilevazioni reali effettuate su alcune porzioni della città di Bologna. Le performance e le caratteristiche del sistema complessivo verranno analizzate e commentate sulla base dei diversi test condotti e si cercherà di evidenziarne i punti di forza o di debolezza a seconda dei diversi scenari e casi d'uso considerati, fornendo eventuali spunti utili per l'evoluzione del progetto sulla base dei risultati ottenuti e dell'esperienza maturata.

Si porrà particolare enfasi sulla necessità di produrre delle logiche in grado di svolgere un'euristica soddisfacente delle reali condizioni del traffico anche in presenza di una scarsa percentuale di penetrazione della tecnologia stessa, ipotizzando eventualmente il ricorso a interfacce di comunicazione di classe inferiore o comunque eterogenee.

Il primo capitolo fornirà una veloce introduzione sulle possibilità offerte dai sistemi ITS in reti veicolari e sulle problematiche ad essi connesse. Il capitolo 2 definirà i concetti più importanti per il sistema che si vuole realizzare e indicherà le principali strategie adottate per soddisfarne i requisiti. Nei capitoli 3 e 4 verrà discussa l'implementazione dell'architettura complessiva e dei diversi protocolli realizzati, mentre nei capitoli 5 e 6 verranno presentati e commentati i risultati ottenuti a seguito dei test condotti e saranno indicati eventuali sviluppi futuri per questo progetto di ricerca.

Capitolo 1

Verso una gestione intelligente del traffico

Tra le principali caratteristiche della moderna società è possibile annoverare una progressiva e costante crescita della popolazione, concentrata molto spesso in aree urbane estese ma comunque caratterizzate da una densità abitativa sempre più elevata. Nei paesi maggiormente sviluppati, questo fenomeno ha portato nel tempo ad accrescere il numero di persone, merci e strumenti che necessitano quotidianamente di compiere numerosi spostamenti.

Come conseguenza di questo fatto, un numero sempre maggiore di città si trova a dover affrontare ogni giorno ingenti problemi legati all'incapacità di gestire una quantità e un flusso di mezzi di trasporto in costante aumento. Queste problematiche si ripercuotono con un generalizzato innalzamento del tempo necessario ai diversi utenti stradali per raggiungere la propria destinazione, con un aumento del numero di incidenti automobilistici e – soprattutto – con un crescente aggravarsi delle condizioni ambientali, fortemente minacciate dall'inquinamento acustico e atmosferico.

Tra le numerose contromisure adottate per arginare o comunque alleviare questo problema, sono emersi negli ultimi decenni numerosi studi e ricerche molto promettenti con il preciso obiettivo di incrementare l'efficienza delle infrastrutture stradali già esistenti attraverso un miglioramento dei sistemi automatici per il controllo del traffico. Tali sforzi si mostrano di partico-

lare interesse in quanto consentono di ottenere sensibili progressi anche in assenza di costosi interventi di ristrutturazione e riprogettazione dei servizi di viabilità.

I sistemi maggiormente diffusi per il controllo automatico del traffico risultano essere gli impianti semaforici per la regolamentazione dell'accesso dei veicoli agli incroci stradali. Tradizionalmente, la modalità di funzionamento principalmente utilizzata per tali dispositivi prevede una temporizzazione fissa e prestabilita di una sequenza temporale di segnali di rosso, arancione e verde per indicare ai veicoli in una direzione se è permesso loro l'accesso all'incrocio. Lo svantaggio principale di questo comportamento è dato dal fatto che esso non tiene minimamente conto delle reali condizioni del traffico circostante, dando luogo molto spesso a situazioni di *unfairness* nella distribuzione degli accessi, con conseguente spreco di tempo e carburante ma soprattutto con l'insorgere di congestioni e rallentamenti che provocano un deterioramento delle condizioni di viabilità e un maggiore inquinamento ambientale.

Il modello di accesso a temporizzazione fissa è oggi considerato del tutto inadeguato e negli ultimi decenni sono stati sperimentati e adottati sistemi *intelligenti* per il controllo del traffico [GSL07]. Questi prevedono di regolamentare il passaggio degli utenti in maniera dinamica, basandosi cioè sulle reali condizioni del traffico in una determinata area e applicando sofisticati algoritmi di *scheduling* per ottimizzare al meglio il flusso dei veicoli nelle diverse direzioni controllate, considerando anche parametri addizionali quali la fascia oraria attuale, l'ordine di priorità di un particolare tratto stradale, numero di corsie disponibili e così via.

Queste strategie evolute di controllo degli accessi hanno tuttavia esigenza di poter conoscere in ogni momento una stima del numero di vetture attualmente presenti nei diversi tratti stradali che precedono il punto di incrocio nel quale l'impianto semaforico è collocato. Il sistema tradizionalmente più utilizzato per rilevare in maniera automatica la presenza di veicoli in punti strategici è il ricorso a sensori a induzione (*induction loop*) installati a pochi centimetri sotto il livello del manto stradale e capaci di indicare l'esistenza o meno di un veicolo sopra a essi. Altri dispositivi frequentemente utilizzati

per tale scopo possono essere dei sensori a pressione, i quali offrono l'ulteriore vantaggio di fornire una stima della velocità corrente del veicolo rilevato.

I sistemi di monitoraggio del traffico mediante sensori interrati presentano purtroppo numerosi svantaggi che ne limitano l'efficacia e di conseguenza l'adozione: essi richiedono anzitutto costi di installazione e manutenzione relativamente elevati, trattandosi di dispositivi collocati al di sotto del livello stradale; inoltre, la loro affidabilità è spesso compromessa da falsi positivi o falsi negativi. Un fattore che in ogni caso ne preclude fortemente le possibilità di uso per impianti semaforici intelligenti è il fatto che il ricorso a queste tipologie di sensori non garantisce un input sufficiente per le logiche di controllo in quanto non sono in grado di fornire informazioni riguardanti il numero di veicoli presenti nelle diverse carreggiate, la lunghezza di eventuali code, la velocità di spostamento, etc ...

Per superare queste difficoltà, sono state proposte nel tempo diverse classi di sensori per la sorveglianza di un flusso di traffico piuttosto che di un semplice veicolo, alcuni esempi possono essere dati da telecamere collegate a complessi software di visione, sensori acustici oppure cellule a infrarossi. Sfortunatamente, tutti questi sistemi comportano una nutrita serie di svantaggi e limitazioni e la loro applicabilità risulta ad oggi ancora seriamente compromessa.

Recentemente è stato introdotto l'uso del termine *Intelligent Transportation System* (ITS) per indicare quel vasto campo di ricerca che si occupa di promuovere tecnologie finalizzate al miglioramento della sicurezza e della qualità della guida, all'incremento della funzionalità dei sistemi di trasporto e dei servizi di viabilità, nonché all'ottimizzazione nell'uso dei carburanti per una migliore salvaguardia dell'ambiente. Gli sforzi di tali studi si spingono quindi ben oltre la realizzazione di soluzioni telematiche per il controllo del traffico, puntando più genericamente all'integrazione delle conoscenze e dei contributi raggiunti nel campo dell'elettronica, dell'ingegneria meccanica, delle telecomunicazioni e dell'informatica per la creazione e la manutenzione di un'infrastruttura stradale "intelligente", cioè capace di promuovere la raccolta automatica di informazioni di stato sull'ambiente e lo scambio di comunicazioni tra veicoli, utenti e dispositivi fissi al fine di realizzare una

gestione del traffico più efficiente e sicura.

Nell'ultimo decennio, le ricerche e le sperimentazioni di ITS più promettenti prevedono l'utilizzo di tecnologie di comunicazione wireless in grado di realizzare forme di comunicazione molto efficienti tra veicoli e infrastruttura stradale (V2I), tra veicoli e altri veicoli vicini (V2V) oppure – più frequentemente – in entrambi i casi d'uso (V2X). Queste funzionalità richiedono sia l'installazione di appositi dispositivi di comunicazione nei punti strategici della rete stradale, sia la dotazione di apposite interfacce wireless all'interno dei singoli veicoli. Diversi enti internazionali e compagnie automobilistiche stanno investendo ingenti risorse nella sperimentazione di prototipi in grado di sfruttare queste opportunità. Grazie all'assegnamento di una banda dedicata da parte delle autorità europee nel 2008 e alla recente standardizzazione di appositi protocolli di comunicazione ad opera dell'ETSI, a partire dal 2015 ci si aspetta una graduale immissione nel mercato consumer di autovetture dotate di sensori, attuatori e sistemi di comunicazione specificatamente pensati per applicazioni di ITS [Kra+13].

L'installazione di interfacce di comunicazione wireless sui singoli veicoli permette la creazione di reti VANET (*Vehicular Ad-hoc NETwork*), ovvero reti ad elevata mobilità nelle quali i nodi consistono di autovetture capaci di comunicare con altri loro pari o con eventuali dispositivi fissi (*Road-side Unit* — RSU) facenti parte dell'infrastruttura stradale [HLE10]. Le opportunità che questa tecnologia permette di sfruttare sono molteplici: attualmente i mezzi di trasporto più recenti dispongono di numerosi sensori in grado di raccogliere una vasta quantità di informazioni sullo stato del veicolo in termini di posizione, velocità, direzione e molto altro; con l'introduzione delle reti VANET, la possibilità di scambiare, condividere, trasferire e processare questa base di conoscenza verso altre vetture o verso dispositivi infrastrutturali dedicati alla gestione del traffico amplierebbe enormemente la diffusione e l'efficacia delle applicazioni ITS precedentemente accennate.

La possibilità di mettere in comunicazione veicoli vicini e di permetterne il collegamento non solo al web ma anche all'infrastruttura che regolamenta e gestisce il servizio di viabilità locale apre concretamente l'orizzonte a degli scenari di traffico intelligente precedentemente soltanto ipotizzabili: raccolta

di statistiche sul traffico con conseguenti interventi automatici e decentralizzati nell'eventualità di congestioni, monitoraggio dei veicoli a scopi di sicurezza e coordinamento dei soccorsi in caso di emergenze, assistenza al guidatore attraverso avvisi o deviazioni suggerite, apparecchiature stradali capaci di reagire in tempo reale alle condizioni contingenti del flusso di vetture locale o addirittura sistemi di guida automatica sono solo alcune delle possibilità attualmente in corso di studio e standardizzazione.

Pur basandosi su un'idea concettualmente semplice, la realizzazione di sistemi ITS mediante reti VANET richiede uno sforzo considerevole in termini di ricerca e sperimentazione. Le caratteristiche intrinseche del canale fisico di comunicazione e l'elevata mobilità tra veicoli pongono diversi problemi ai protocolli di comunicazione richiesti, ulteriormente aggravati dalla necessità di coordinare e gestire reti fortemente dinamiche, mutevoli e completamente decentralizzate. Per qualsiasi architettura che si intende realizzare occorre oltretutto disporre di approfonditi studi e sperimentazioni in scenari reali o simulati in grado di poter quantificare il rapporto costi-benefici offerto dalla particolare soluzione proposta.

Per agevolare la ricerca nell'ambito delle reti VANET e semplificarne il processo di sviluppo, il progetto iTetris [14b] mette a disposizione una piattaforma aperta, flessibile e conforme agli standard europei ETSI pensata per facilitare la simulazione e la messa a punto di applicazioni ITS efficienti. Questo studio ha visto la collaborazione di diversi partner e università a livello europeo e ha prodotto e distribuito una considerevole quantità di pubblicazioni e *deliverable* utili per la sperimentazione di scenari per il controllo del traffico intelligente. Una parte del materiale e della documentazione realizzati da iTetris sono stati utilizzati per questo stesso lavoro di tesi.

COLOMBO [14a] è un ulteriore progetto attualmente in corso e finanziato dall'unione europea che punta a recuperare e ampliare gli studi e i risultati ottenuti in iTetris per applicarli al problema del controllo del traffico alle intersezioni. Scopo di COLOMBO è quello di ricercare e sperimentare un insieme di algoritmi e soluzioni per la sorveglianza del traffico e per la gestione dei flussi di veicoli in corrispondenza di incroci regolati da impianto semaforico. Analoghe ricerche condotte in passato hanno proposto alcuni ap-

procci a questo problema e questi sono risultati in genere piuttosto efficienti purché si ammetta di disporre di una certa quantità di utenti dotati di veicoli equipaggiati con i sistemi e le architetture studiate. L'idea che caratterizza COLOMBO è quella di riuscire a realizzare una piattaforma specificamente progettata per poter funzionare anche in presenza di basse percentuali di penetrazione della tecnologia utilizzata.

Il lavoro di tesi presentato in questo testo si inserisce nel contesto di questo progetto europeo e parte dall'esigenza di riuscire a realizzare un insieme di protocolli in grado di rilevare la presenza di autovetture in avvicinamento a un generico semaforo e di raccogliere delle informazioni di stato che consentano a tale RSU (cioè al dispositivo semaforico) di ottenere una stima il più possibile veritiera delle attuali condizioni del flusso di veicoli provenienti dai diversi sensi di marcia previsti per tale incrocio.

I dati prodotti dal sistema che verrà di seguito proposto e analizzato costituiranno l'input di una successiva logica collocata internamente al semaforo e incaricata di applicare delle apposite politiche intelligenti di monitoraggio e gestione dinamica del traffico. Lo studio qui realizzato si concentrerà sulle modalità con cui è possibile recuperare e integrare tali informazioni, tralasciando i dettagli riguardanti le logiche che si occuperanno in un secondo momento di utilizzarle per intervenire in tempo reale sulle condizioni della rete stradale.

Il dominio applicativo del presente lavoro di tesi viene schematizzato sinteticamente in Figura 1.1 dove viene illustrato il processo ad alto livello secondo il quale un sistema ITS interagisce con l'ambiente circostante per realizzare le proprie funzionalità. Il contesto di riferimento è una predefinita area urbana contenente l'incrocio che si intende monitorare e dalla quale si estraggono le informazioni relative alle condizioni del traffico dei veicoli; queste vengono opportunamente rielaborate e consegnate ai dispositivi dell'infrastruttura stradale che si preoccupano di intervenire in tempo reale sul sistema stesso nei modi e nei tempi previsti dalle particolari politiche di gestione osservate, eventualmente sfruttando uno storico delle rilevazioni passate. Attraverso continui processi di azione e retroazione, il sistema si sforza nel tempo di mantenere le condizioni del traffico veicolare in costante equilibrio.

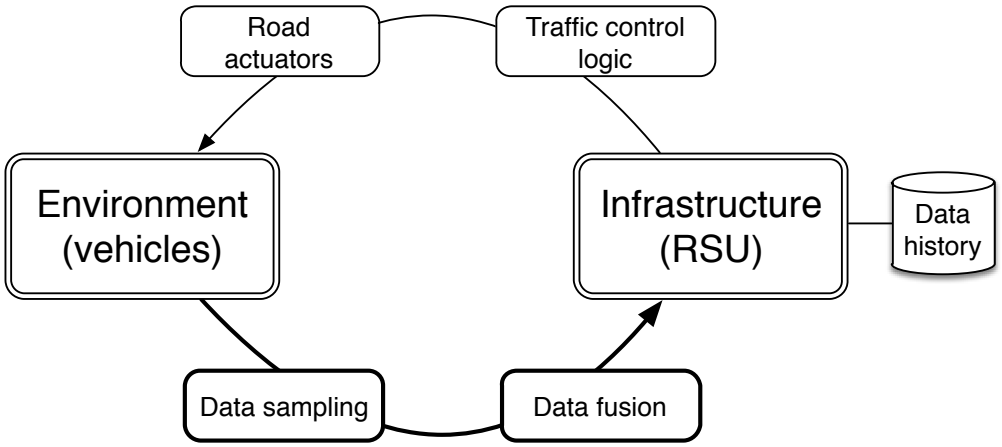

Figura 1.1: Schematizzazione ad alto livello del processo di gestione del traffico veicolare mediante un generico sistema ITS. I blocchi concettuali sui quali si è concentrato il presente lavoro di tesi vengono evidenziati in grassetto.

Un sistema con un funzionamento conforme a quanto appena descritto può ovviamente ricorrere a svariate tecnologie per l'approvvigionamento dei dati in ingresso; quelli recuperabili sfruttando strategie di tipo VANET risultano tuttavia sensibilmente più completi e accurati rispetto a quelli ottenibili servendosi dei tradizionali sensori a induzione o a pressione oppure ricavabili attraverso complessi sistemi di visione. Infine, la dotazione di interfacce wireless e tecnologie ITS sui singoli veicoli – pur richiedendo ingenti sforzi e risorse da investire in ricerca, sperimentazione e standardizzazione – risulta a lungo termine molto più conveniente in termini di rapporto vantaggio-costi rispetto all'installazione dei tradizionali dispositivi di rilevazione precedentemente accennati, i quali richiedono continui e costosi interventi di manutenzione stradale uniti a una costante supervisione manuale da parte dell'uomo.

La strategia principale che si intende perseguire nella raccolta dei dati dall'ambiente prevede di organizzare i veicoli in *gruppi* ogniqualvolta questi si avvicinano a un incrocio controllato da una RSU. Ogni gruppo sarà costituito esclusivamente da quelle vetture che stanno percorrendo uno stesso tratto stradale e promuoverà l'interazione tra i suoi diversi membri per raccogliere e integrare i dati dei veicoli che vi partecipano. Queste informazioni verrano successivamente inviate alla RSU che provvederà ad rielaborare e registrare

le statistiche in ingresso, organizzandole in base alla specifica direzione di provenienza.

L'idea di coordinare le interazioni tra i nodi di un sistema VANET organizzandoli in gruppi è ampiamente sfruttata e documentata in letteratura [VBK12] e in molte ricerche si fa uso degli analoghi concetti di *cluster* o plotone, talvolta con accezioni leggermente diverse. Nel seguito di questo testo, questi termini verranno intesi come sinonimi.

Per poter raggruppare i veicoli e dirigere le loro interazioni è necessario predisporre degli appositi protocolli per la formazione di un gruppo, oltre a quelli indispensabili per la gestione del proprio ciclo vitale e per lo svolgimento dei compiti assegnati. Nel capitolo seguente verranno meglio definiti i concetti di rilevanza fondamentale per il sistema e si indicheranno le principali scelte adottate per soddisfare i requisiti appena esposti.

Le varie logiche di protocollo implementate nel corso di questo progetto verranno presentate nel capitolo 4 e costituiscono il fulcro principale del sistema complessivo. Nel capitolo 3 verranno invece esposti i componenti che costituiscono la piattaforma di base per l'architettura prodotta e gli strumenti software utilizzati per implementarli e per simulare un ambiente di rete su scala urbana. Infine, nei capitoli 5 e 6 verranno discussi e commentati i risultati ottenuti a seguito dei test condotti, concludendo con un'analisi dei vari punti di forza e imperfezioni e suggerendo eventuali sviluppi futuri per il lavoro realizzato.

Capitolo 2

Raggruppamento dei veicoli

Scopo di questo capitolo è quello di presentare le soluzioni risolutive che sono state considerate e le scelte affrontate per soddisfare i requisiti presentati nelle precedenti pagine. Ci si concentrerà quindi su una visione di alto livello del problema, trascurando in questa fase qualsiasi dettaglio non strettamente correlato all'individuazione di una corretta strategia implementativa. Per un'approfondita analisi delle problematiche legate alla concreta realizzazione dei protocolli proposti, delle scelte architetturali e delle modalità di simulazione e sperimentazione si rimanda al capitolo successivo.

Volendo riassumere brevemente le funzionalità che il progetto si propone di soddisfare si ricorda come la principale esigenza che ha ispirato l'intero lavoro di tesi sia quella di poter fornire a dei dispositivi posti in corrispondenza a dei punti di viabilità critici (principalmente impianti semaforici posti a regolazione di incroci stradali) informazioni relative a una stima del traffico dei veicoli in arrivo, opportunamente differenziate in base allo specifico senso di provenienza. La raccolta di simili informazioni può essere in generale utilizzata per ricavare nel lungo periodo delle statistiche sul flusso di veicoli che attraversa una porzione di rete stradale di particolare interesse. Più concretamente questi rilevamenti possono essere sfruttati per informare in tempo reale ogni dispositivo semaforico (RSU) della quantità di veicoli che sopraggiungono da ciascuna direzione di traffico, in modo da poter realizzare delle politiche di regolamentazione del passaggio più intelligenti e proattive

rispetto alla semplice temporizzazione dei segnali di rosso, verde e giallo.

I requisiti minimi per poter operare in questo scenario prevedono che sia i dispositivi mobili sia i punti infrastrutturali fissi dispongano di una strumentazione GPS o equivalente per la determinazione più o meno precisa della propria posizione e di almeno un'interfaccia di comunicazione remota per lo scambio di messaggi. Per i motivi già esposti nel precedente capitolo, la maggior parte delle ricerche riguardanti network veicolari si concentrano sull'utilizzo di 802.11 come principale standard di rete. In particolare, 802.11p e 802.11s risultano essere gli stack protocollari specificatamente pensati per un utilizzo nel campo delle reti VANET, in particolare per le loro caratteristiche di QoS e per la possibilità di realizzare reti mesh e protocolli di geo-routing tra nodi ad elevata mobilità. Oltre alle soluzioni sopra citate, il recente successo ed enorme diffusione di dispositivi handheld quali PDA, smartphone, tablet e i primi prototipi di accessori wearable[1] hanno spinto notevolmente l'evoluzione dei tradizionali apparati di rete 802.11b/g. Le crescenti necessità di coprire con la stessa tecnologia Wifi spazi di lavoro, ambienti domestici e addirittura scenari di tipo urbano, hanno portato alla ratificazione degli standard 802.11n e – più recentemente – 802.11ac [PG11] in grado rispettivamente di raddoppiare e triplicare il raggio di copertura delle soluzioni wireless tradizionali di tipo G, oltre a garantire un flusso di dati sempre più veloce, latenze contenute e ulteriori funzionalità avanzate. A fronte di simili progressi, viene naturale ipotizzare che tali interfacce possano vedere nell'immediato futuro un'ampia adozione anche a bordo dei tradizionali veicoli consumer destinati al mercato di massa. A supporto di questa previsione è possibile notare come alcune case automobilistiche stiano già investendo nella ricerca sull'integrazione di diverse tecnologie wireless (Bluetooth, UMTS e più recentemente LTE e Wi-Fi) a bordo di autovetture di ultima generazione, anche se per il momento la motivazione principale è quella di soddisfare esigenze di comunicazione, *infotainment* e sicurezza stradale [Jur12; CSZ11].

È comunque necessario specificare che le soluzioni proposte in questo testo non presuppongono l'utilizzo di un particolare standard 802.11 rispetto ad

[1] Si considerino i recenti Google Glass, Samsung Gear, Pebble, MOTOACTV o il tanto vociferato iWatch

altri: la motivazione è dovuta principalmente al fatto che le comunicazioni tra i diversi nodi – come verrà meglio precisato più avanti in questo capitolo – avvengono esclusivamente mediante l'invio di semplici trame di tipo broadcast in configurazioni wireless ad-hoc. Non vengono pertanto sfruttate quelle funzionalità di routing, QoS o controllo degli accessi che costituiscono le caratteristiche che contraddistinguono gli standard 802.11 più avanzati. Questa scelta è dettata dalla volontà di sviluppare una strategia risolutiva di base in grado di affrontare le problematiche sopra esposte in maniera semplice, chiara ed essenziale, svincolandosi il più possibile da particolari vincoli implementativi. Ovviamente, nel caso in cui tale soluzione riveli dei punti di forza in sede di studio e simulazione, sarà possibile in un secondo momento procedere a ulteriori studi e sperimentazioni finalizzate alla messa a punto di una sua realizzazione concreta e all'individuazione di quelle particolari scelte tecnologiche, revisioni e miglioramenti che si possono applicare per renderla maggiormente efficace e competitiva. Si può infine considerare questo lavoro di tesi come un punto di partenza per altre soluzioni – anche differenti come finalità e impostazione – che intendono comunque sfruttare i concetti fondamentali su cui l'intero sistema si fonda: le nozioni di direzione e raggruppamento possono infatti essere riutilizzate e adattate per rispondere ad esigenze che non necessariamente hanno a che vedere con il rilevamento e il controllo del traffico. Le nozioni basilari presenti in questo capitolo verranno quindi esposte in maniera volutamente generica per poter offrire un valido spunto a tutte le loro possibili applicazioni pratiche.

È importante sottolineare a questo punto come le informazioni ricavate per mezzo degli strumenti qui presentati costituiranno soltanto una stima necessariamente imperfetta del reale flusso di veicoli in transito verso la RSU: esistono infatti una moltitudine di variabili, fattori avversi ed eventi imprevedibili che possono inficiare la qualità dell'euristica fornita quali l'eterogeneità dei dispositivi in uso e il loro livello di penetrazione, imperfezioni nei protocolli realizzati, errori e interferenze nella trasmissione dei messaggi, comportamenti inattesi degli utenti del servizio, condizioni climatiche sfavorevoli che riducono la portata e la qualità delle trasmissioni, generici malfunzionamenti e molto altro ancora.

Alla luce di questi presupposti, qualsiasi soluzione proposta in ambito VANET deve tenere particolarmente conto – oltre che dell'elevata mobilità ed eterogeneità dei dispositivi coinvolti – delle condizioni tipicamente avverse in cui ci si trova a dover operare. I protocolli di seguito sviluppati cercheranno dunque di tollerare il più possibile eventuali perdite di pacchetto. Qualora risulti necessaria una maggiore garanzia che un destinatario riceva delle informazioni di particolare criticità verrà richiesta la conferma dei dati ricevuti mediante appositi messaggi di acknowledgement.

2.1 Data fusion

Con il termine *Data Fusion* si intende descrivere un processo finalizzato a integrare un vasto gruppo di dati di basso livello in un insieme di informazioni più generali e rappresentabili in maniera più leggibile, in modo da poter usufruire di una visione più ampia e utile del particolare contesto che si vuole esaminare [HL01]. Il termine è in realtà piuttosto generico e viene spesso meno propriamente utilizzato per indicare una qualsiasi serie di trasformazioni operate su una vasta collezione di dati con lo scopo di ottenerne una visione di più alto livello o comunque maggiormente riassuntiva o *human-friendly*. Esistono poi ulteriori termini volti a suggerire diverse accezioni o particolari contesti applicativi, quali *Information Integration*, *Sensor Fusion* oppure *Image Fusion*. In questo testo utilizzeremo il termine nel suo senso più generale e lo applicheremo allo specifico dominio applicativo delle reti veicolari. Intenderemo dunque con Data Fusion la raccolta di un grosso quantitativo di dati ad opera dei singoli veicoli e la loro successiva integrazione in una struttura informativa che ne riassume in maniera più sintetica le proprietà di maggiore interesse. Anticipando i concetti che verranno di seguito esposti, possiamo prevedere che le informazioni di più basso livello riguardano lo stato del singolo veicolo facente parte del sistema e possono comprendere i dati ottenuti mediante i diversi dispositivi di cui esso è dotato: la sua posizione, la sua velocità o la presenza di eventuali altri nodi vicini; questa vasta base di conoscenza viene quindi raccolta ed elaborata per ricavare un'euristica

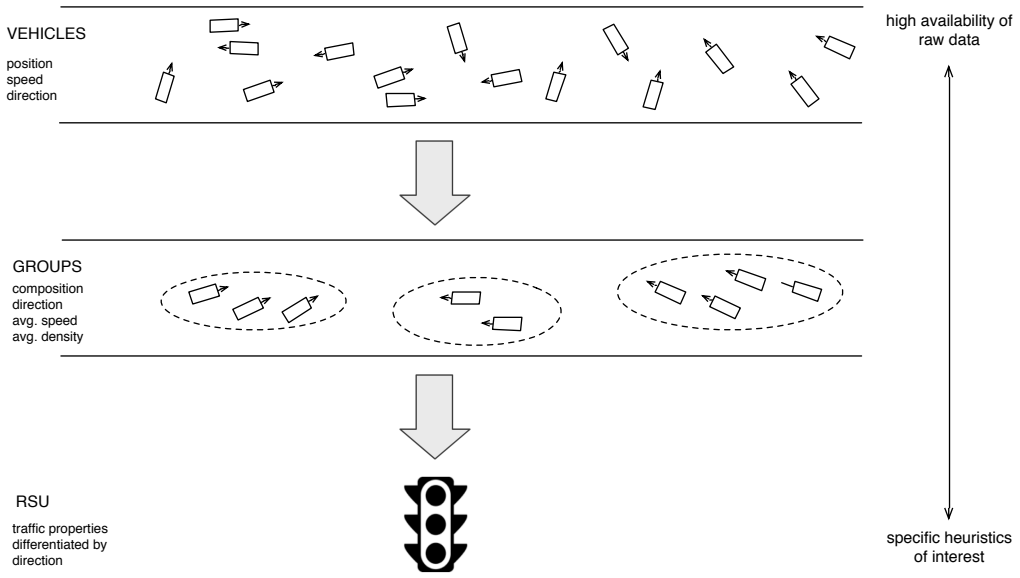

Figura 2.1: Flusso delle informazioni e data fusion dai singoli veicoli alla RSU: l'elevato quantitativo di dati forniti dai diversi sensori viene integrato in una base di conoscenza di più alto livello.

sull'attuale condizione del traffico e sul comportamento degli utenti che lo compongono [FLK11].

La Figura 2.1 riassume sinteticamente questo processo: i numerosi veicoli sparsi nello scenario (vedi figura, in alto) sono in grado individualmente di raccogliere una vasta quantità di informazioni su se stessi e l'ambiente che li circondano. Questo insieme di dati risulta però troppo ampio e dettagliato per potere essere direttamente utilizzato come stima delle condizioni del flusso di veicoli in una determinata zona o verso una specifica direzione. Inoltre, il trasferimento di un quantitativo di informazioni così abbondante verso un unico dispositivo si rivelerebbe una soluzione estremamente centralizzata che potrebbe risultare molto controproducente in diversi casi: anzitutto, in uno scenario ad elevata mobilità e soggetto alle difficoltà che abbiamo appena esposto è particolarmente complicato garantire che un simile flusso di dati possa essere sempre trasmesso senza problemi; in secondo luogo, i dispositivi preposti alla ricezione dei diversi contributi informativi si ritroverebbero – specialmente in condizioni di elevata densità – a dover processare in tempo reale una base di conoscenza di dimensioni considerevoli.

Una soluzione particolarmente naïve al problema della determinazione del traffico in arrivo potrebbe in effetti prevedere che ogni veicolo invii un semplice pacchetto informativo contenente le ultime rilevazioni effettuate dai propri sensori (coordinate GPS, orientamento, velocità ...) non appena un dispositivo interessato a raccoglierle si trovi a portata; quest'ultimo si incaricherà poi di sintetizzare tutte le informazioni in proprio possesso e di tradurle in un output utilizzabile da applicativi software di più alto livello. In alcuni contesti una simile strategia può rivelarsi indubbiamente efficace e di facile realizzazione, tuttavia l'architettura derivante risulterebbe inevitabilmente centralizzata e questo potrebbe porre dei seri limiti al suo utilizzo in termini di scalabilità e performance. Questa scelta si profilerebbe inoltre come una soluzione estremamente specific-purpose, dal momento in cui tutta la logica di trattamento e trasformazione dei dati risulterebbe concentrata in un'unica classe di componenti – la RSU nel nostro caso. Qualora si rendesse necessario modificare la metodologia con cui le informazioni vengono trattate, rappresentate o trasmesse, magari per soddisfare l'esigenza di poter adattare il sistema a diversi utilizzi rispetto a quelli per il quale era stato in origine concepito, si dovrebbe obbligatoriamente intervenire su un modulo che potrebbe velocemente diventare troppo complesso e monolitico da poter essere mantenuto con facilità.
Come abbiamo già avuto occasione di specificare, l'impianto architetturale presentato in questo testo non ha unicamente lo scopo di rispondere alle problematiche di traffic management già esposte in precedenza, ma vuole anche essere un punto di partenza flessibile per soluzioni ITS di diverse tipologie. Ciò si concretizza – come verrà meglio esposto nel seguito – nella possibilità di offrire una visione delle informazioni relative al flusso di nodi in termini non dei singoli individui ma piuttosto di un loro raggruppamento. Questa maggiore granularità dei dati, integrati in maniera distribuita dai vari agenti del sistema e non dalla singola RSU, consente di spostare buona parte del carico computazionale dalle unità infrastrutturali fisse verso i dispositivi mobili, ottimizza il traffico di rete e promuove lo scambio di una maggiore *context-awareness* tra i veicoli.

Come si nota osservando la Figura 2.1, questa funzionalità viene realizzata

introducendo un livello di indirettezza aggiuntivo tra la produzione copiosa di dati *"raw"* e la loro assimilazione in forma più compatta, contestualizzata e astratta da parte dei livelli applicativi più elevati. Una RSU (stilizzata con l'icona di un semaforo) non viene più informata dello stato di ogni singolo veicolo in avvicinamento – nozione che dal suo punto di vista appare troppo dettagliata e di difficile gestione – ma riceve piuttosto informazioni sulla presenza di uno o più insiemi di veicoli e sulle loro proprietà.

2.2 Formazione dei gruppi

In precedenza è già stato anticipato che la soluzione in esame presuppone la capacità da parte di ogni nodo di ottenere adeguate informazioni sulla propria posizione e sulla velocità corrente, sia pure con un certo grado di incertezza e imprecisione. In assenza di sensori specifici in grado di rilevare la velocità istantanea del veicolo[2] è comunque possibile ricavare questa grandezza esaminando una serie di campionamenti della posizione effettuati a distanza sufficientemente ravvicinata e calcolando una media del rapporto tra la distanza che intercorre fra due campioni e il tempo che li separa. Un altro importante parametro che risulta facilmente estraibile dal campionamento nel tempo della propria posizione è la direzione corrente. Siccome questo concetto ricopre un ruolo cruciale nell'impostazione complessiva dell'intero sistema, è di fondamentale importanza definirlo con chiarezza e capire come e perché questa informazione verrà utilizzata per differenziare i vari veicoli presenti nell'ambiente.

Nel precedente paragrafo è stato ampiamente spiegato il principio cardine sul quale si fonda l'intera soluzione proposta: la volontà di integrare i dati di basso livello allo scopo di fornire alle RSU delle informazioni relative a uno o più insiemi di veicoli. È necessario a questo punto chiarire su quali basi si intende assegnare ciascun nodo a un preciso *gruppo* piuttosto che a un altro. L'esigenza è che i dati "fused" di ogni gruppo contribuiscano a delineare lo stato complessivo del flusso del traffico in un determinato senso di marcia:

[2] In particolare accelerometri e giroscopi

per poter applicare delle politiche intelligenti di regolazione del traffico stradale, non è sufficiente che il dispositivo che regola l'accesso dei veicoli a uno snodo conosca il numero di vetture in avvicinamento ma è cruciale che tale informazione – unita ad altre – sia contestualizzata in base alla *direzione* che queste stanno seguendo. Se a scopo di esempio si considera lo scenario rappresentato in Figura 2.2, dove un impianto semaforico ha il compito di consentire o arrestare il passaggio di automobili in un incrocio tra due arterie stradali, si comprende immediatamente come la logica che stabilisce in ogni istante quali sensi di marcia sono autorizzati a transitare necessita di essere continuamente informata sulle condizioni del traffico relative alle quattro diverse strade che si stanno monitorando.

Basandosi su tutti i requisiti esposti fino a questo momento, risulta naturale pensare di definire il concetto di gruppo sfruttando i due concetti di distanza e direzione: stabiliamo quindi che un insieme di veicoli costituisce un gruppo soltanto se questi sono accomunati dal fatto di essere fisicamente vicini uno all'altro e di possedere lo stesso senso di marcia. Questa strategia di raggruppamento non costituisce una novità essendo del tutto analoga all'idea di plotone frequentemente esposta in letteratura ed utilizzata in numerosissime sperimentazioni in ambito VANET. Nel nostro contesto, i due termini sono intercambiabili e verranno utilizzati senza alcuna distinzione. Rimane tuttavia da chiarire quando e in che misura due nodi possono essere sufficientemente vicini per poter essere considerati membri di uno stesso gruppo e in che modo il loro orientamento viene utilizzato per inferire l'appartenenza alla stessa direzione di marcia.

2.2.1 Membri del gruppo e comunicazione

Una soluzione che si ponesse l'obiettivo di essere versatile e completamente personalizzabile potrebbe semplicemente assumere che due veicoli risultano vicini quando la loro distanza reciproca non supera una certa soglia stabilita in fase di configurazione del sistema. Questo permetterebbe di adattarsi a quei casi d'uso dove dei gruppi di dimensione troppo estesa potrebbero risultare indesiderati o problematici da gestire e si preferisce avere il massimo

controllo sui criteri utilizzati per la creazione dei plotoni. Questa funzionalità potrebbe però portare a molteplici difficoltà di implementazione dal momento che occorre tenere conto del fatto che la capacità di comunicazione di ogni nodo – inclusa la stessa RSU – è di fatto limitata dalla portata del segnale utilizzato per la trasmissione dei dati, una grandezza che può variare in maniera impredicibile da dispositivo a dispositivo e che si trova ad essere fortemente influenzata dalle contingenti condizioni dell'ambiente. Nel momento in cui venisse richiesta un'estensione dei plotoni superiore al range dei veicoli che lo compongono, si dovrebbe necessariamente fare a ricorso a un meccanismo di scambio dei dati che consenta ad ogni veicolo di consegnare un messaggio destinato ad un qualsiasi altro membro di quello stesso gruppo, indipendentemente dalla distanza che intercorre tra essi e servendosi eventualmente di uno o più nodi intermediari per recapitare il contenuto nel caso in cui un'eccessiva separazione tra mittente e destinatario non consenta uno scambio di informazioni diretto. Questo problema è stato ampiamente studiato ed esiste una letteratura [HXG02] molto vasta e approfondita sui diversi protocolli e algoritmi di routing che possono essere applicati al fine di permettere la comunicazione tra nodi distanti, sopratutto nel campo delle reti veicolari [ASP08]. In ogni caso, a causa dell'elevata mobilità tra i nodi in esame, quasi tutte le soluzioni disponibili sono di tipo *best-effort* e nessuna di esse è in grado di garantire in ogni momento uno scambio di dati efficace e stabile. Inoltre, per via dei possibili pattern di mobilità differenti con cui si spostano i loro membri, i plotoni possono sempre andare incontro a partizionamenti e perdite di connettività che possono seriamente compromettere le loro interconnessioni.

Avendo precedentemente dichiarato l'intenzione di sviluppare un'architettura il più possibile semplice e poco costosa, l'introduzione di avanzati protocolli di routing tra i partecipanti di un gruppo risulta senz'altro contraddittoria e porterebbe a dover gestire una complessità tale da non essere giustificata a fronte di funzionalità che raramente risultano utili per i casi d'uso previsti: per arrivare a una buona stima del flusso del traffico limitate al breve tratto che anticipa uno snodo stradale non occorre infatti prevedere la formazione di plotoni di veicoli particolarmente estesi e qualora sussistano

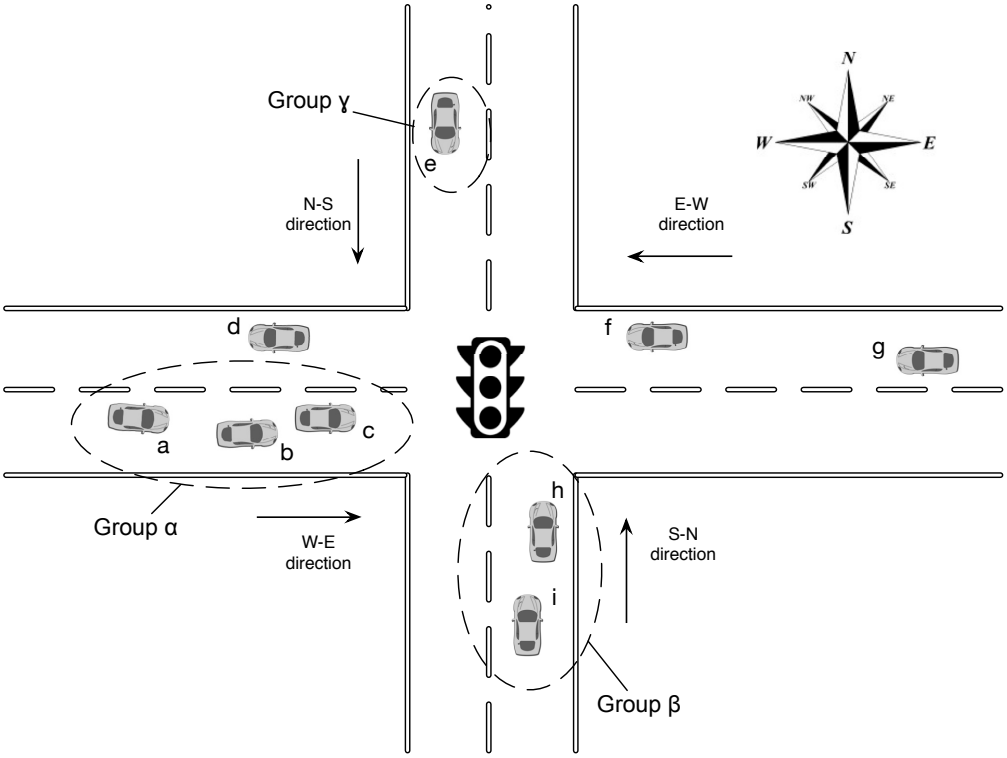

Figura 2.2: Esempio di raggruppamento dei veicoli in base a direzione e vicinanza: i nodi a, b, c e h, i formano rispettivamente i gruppi α e β, ciascun veicolo condivide un rapporto di vicinanza e direzione rispetto ai membri dello stesso gruppo. Sono ammessi gruppi anche se composti di un solo membro (e).

condizioni di elevata densità o congestione, dei gruppi dal diametro relativamente contenuto riusciranno comunque a suggerire alla RSU l'avvento di uno stato critico, eventualmente ricorrendo a informazioni aggiuntive raccolte dai singoli gruppi nelle modalità che verranno esposte nei successivi capitoli oppure avvalendosi di semplici tecniche di comunicazione tra plotoni adiacenti, come suggerito nel paragrafo 6.3.

Una strategia diametralmente opposta potrebbe imporre che una qualsiasi coppia di nodi deve essere in comunicazione diretta per poter appartenere allo stesso gruppo. Questo risolverebbe alla radice qualsiasi problema di comunicazione tra veicoli distanti ma risulta essere una condizione eccessivamente vincolante visto che sotto simili presupposti l'estensione massima di un plotone sarebbe sempre limitata al più piccolo tra i raggi di portata

dei singoli membri. I gruppi formati seguendo tale direttiva potrebbero rivelarsi di dimensione eccessivamente ridotta e potrebbero anch'essi subire dei sensibili cali di performance dovuti a frammentazione.

Per rispondere in maniera semplice a questi inconvenienti si è preferito ricorrere a un compromesso tra i due estremi appena esposti: si stabilisce che all'interno di ogni gruppo esista un nodo che svolge la funzione di coordinatore dei vari membri e si preoccupi di raccogliere da questi ogni sorta di informazione utile per poter sintetizzare una serie di proprietà di più alto livello che descrivono lo stato complessivo del gruppo, in accordo a quanto esposto nel paragrafo 2.1. Le caratteristiche e le responsabilità affidate a questo ruolo verranno meglio discusse nella sezione 2.2.3, anticipiamo comunque che il concetto di leadership appena introdotto è di particolare importanza per rispondere al problema di quale disposizione spaziale devono godere i nodi per poter partecipare a un plotone. Per definizione infatti, si impone che un qualsiasi veicolo deve essere all'interno della portata di un *leader* di gruppo per potersi unire a quest'ultimo. Da questa scelta consegue il fatto che il leader di un gruppo è in grado di comunicare direttamente con qualsiasi suo membro e viceversa, tuttavia non viene in alcun modo garantito che i diversi nodi riescano sempre a scambiarsi informazioni: nell'esempio mostrato in Figura 2.2 i nodi a e c potrebbero essere troppo lontani per comunicare (come avviene per i nodi f e g), ciononostante riescono a partecipare entrambi al gruppo α in quanto è il nodo b che riesce a farsi carico di monitorare le loro attività all'interno del proprio range; a e c restano comunque ignari l'uno della presenza dell'altro. Questa limitazione è comunque perfettamente tollerabile: nonostante il sistema sviluppato non ricorra mai a un contatto diretto tra due nodi membri questo resta comunque facile da realizzare servendosi del group leader come intermediario tra i due pari.

2.2.2 Direzione

Il fatto che un veicolo si trovi nel raggio di portata di un group leader non è in nessun caso una condizione sufficiente per stabilire la sua ammissibilità come membro del gruppo. Si consideri ad esempio una carreggiata percor-

ribile in due direzioni: due veicoli che si trovano a percorrere la strada con sensi di marcia opposti arrivano a trovarsi molto vicini nel momento in cui si incrociano, tuttavia – per i requisiti che abbiamo fissato – non è possibile considerarli come potenziali membri di uno stesso gruppo (si confronti il caso del nodo d in Figura 2.2). Per come è stato definito, un plotone viene sempre composto da veicoli che condividono una stessa *direzione*, cioè che in un dato istante si stanno dirigendo in maniera pseudo-solidale verso uno stesso punto. Una RSU posta a gestione di un incrocio stradale o di un qualsiasi snodo sensibile può fare affidamento a questa fondamentale caratteristica dei gruppi per differenziare il traffico in avvicinamento a seconda della sua provenienza.

Dovendo quindi mettere a confronto le direzioni dei diversi veicoli è indispensabile disporre di un sistema di riferimento appropriato per poter operare con questa grandezza. In generale il problema non è di facile soluzione: nonostante sia possibile con buona approssimazione assegnare ad un oggetto posto sulla superficie terrestre una posizione univoca, non è tuttavia immediato individuare un metodo che consenta di confrontare in ogni momento l'orientamento di due o più mezzi di trasporto. Per evitare di ricorrere a complessi sistemi geodetici è sufficiente osservare che nei casi d'uso previsti per le normali applicazioni di traffic management – e in particolare per quella in esame – non vengono mai messe in atto delle interazioni tra nodi talmente distanti da dovere essere considerati come vetture in movimento su una superficie curva. Anche se si prevedesse l'utilizzo delle soluzioni qui esposte per finalità ulteriori rispetto allo studio del flusso di autovetture, sarebbe perfettamente accettabile posizionare queste ultime in un piano bidimensionale con l'asse delle ascisse positive solidale all'est geografico. Un simile riferimento è del tutto analogo al sistema locale East, North, Up (ENU[3]) definito in [Dan03] e [Tor01]. Il passaggio da delle coordinate di tipo geografico utilizzate nei comuni dispositivi GPS alle coordinate cartesiane in accordo con il riferimento appena esposto risulta una procedura piuttosto semplice a patto di fissare l'origine del piano ENU in uno stesso punto. Se si considerano delle grandezze che stimano una distanza tra due punti questo accorgimento è ov-

[3]Si consulti anche `http://en.wikipedia.org/wiki/Geodetic_system`

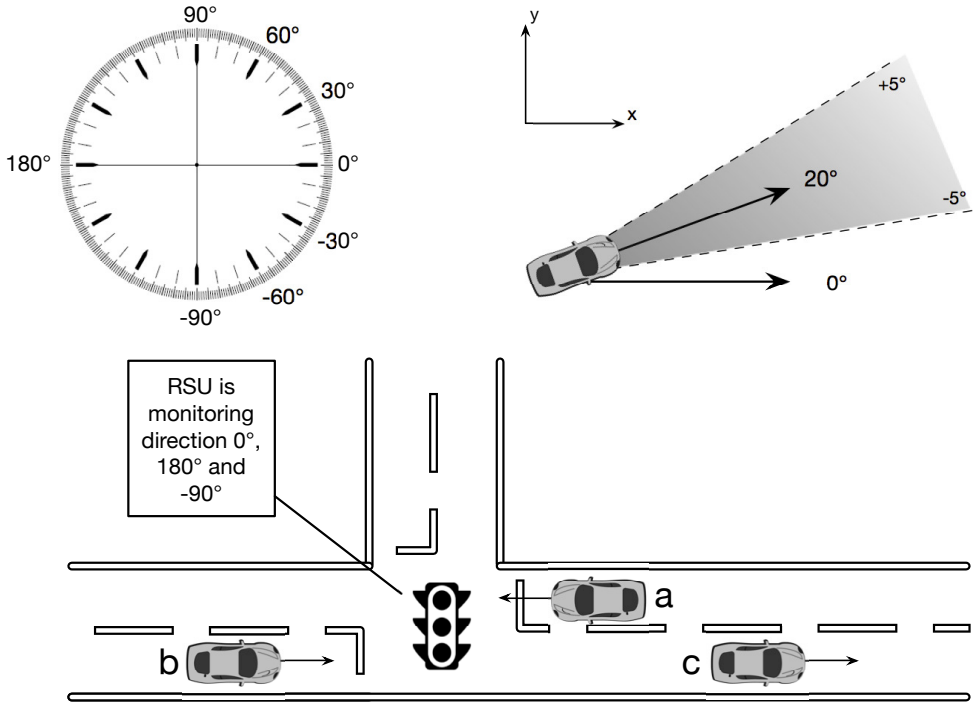

Figura 2.3: Definizione di direzione per un veicolo, cono di conformità e applicazione in uno scenario esemplificativo.

viamente trascurabile, quando viceversa occorre lavorare con una posizione assoluta è possibile definire l'origine in base alle proprie esigenze.

Avendo definito un sistema di riferimento cartesiano per il posizionamento dei veicoli, il problema di assegnare una direzione univoca a qualsiasi oggetto in movimento è di facile soluzione e viene affrontato come descritto in Figura 2.3: la direzione viene espressa in gradi d'arco compresi tra -180° e 180°. La direzione 0° è definita come quella dell'asse delle ascisse crescenti (est geografico) e l'angolo che il vettore velocità di un qualsiasi veicolo forma con esso determina il valore di tale grandezza; le direzioni di 180° e -180° vengono considerate equivalenti. Ovviamente a un mezzo di trasporto fermo o con velocità talmente bassa da non risultare percepibile dagli strumenti non ha senso assegnare un orientamento, tuttavia è possibile aggirare facilmente questo ostacolo preservando l'ultima grandezza valida registrata.

Una volta introdotto questo semplice formalismo è possibile sfruttarlo per

regolare la formazione dei gruppi di veicoli con la procedura già suggerita in precedenza: si uniscono tra loro nodi che riescono a comunicare in maniera diretta con un loro pari eletto a ruolo di leader[4] e che possiedono lo stesso valore di direzione. Ogni plotone contribuisce in questo modo a raccogliere e trasmettere delle informazioni relative esclusivamente allo stato del traffico nel senso di marcia che sta seguendo, queste verranno quindi raccolte da un qualsiasi dispositivo dell'infrastruttura interessata al monitoraggio del flusso di veicoli in un particolare punto sensibile e verranno processate e differenziate a seconda del valore di direzione relativo al gruppo che le ha generate. Si consideri l'esempio in Figura 2.3 dove una RSU (semaforo) controlla l'accesso dei veicoli alle tre strade raffigurate: è ragionevole supporre che i dispositivi di infrastruttura conoscano a priori un vettore di valori corrispondenti alle direzioni da cui si desidera ricevere degli aggiornamenti di stato, visto che questi dipendono dalla particolare topologia stradale in quel punto e rimangono fissi nel tempo. Nel nostro caso questi parametri dovrebbero valere 180° per i veicoli provenienti da destra (il nodo a nell'esempio), 0° per quelli che raggiungono l'incrocio da sinistra (nodo b) e -90° per quelli che provengono da nord. In generale questo sistema può essere applicato a qualsiasi tipo di incrocio o snodo comunemente utilizzato, si noti comunque che occorre assicurarsi che i segni degli angoli siano scelti in accordo alle direzioni previste per i veicoli che si dirigono verso una RSU. Terminando l'analisi dell'esempio fornito, si osservi come il veicolo c non può partecipare a un ipotetico plotone che comprende il nodo a in quanto le direzioni dei due veicoli non corrispondono; esso non deve inoltre consegnare alcun dato alla RSU visto che ha già attraversato l'incrocio e si sta allontanando, tuttavia la sua direzione è compatibile con quelle monitorate. Questa complicazione è risolvibile tenendo conto che il veicolo in questione deve avere necessariamente attraversato l'incrocio in precedenza e ha già svolto un'interazione con il dispositivo, ulteriori comunicazioni possono essere scongiurate prevedendo una lista di RSU già visitate ed escludendo temporaneamente queste ultime dai protocolli utilizzati dai nodi.

Esiste tuttavia un ultimo problema che è necessario affrontare: in uno

[4]Le modalità con cui questa elezione si svolge saranno argomento del capitolo successivo.

scenario realistico, durante la marcia, un veicolo effettua dei continui e imprevedibili cambi di direzione dovuti a molteplici concause: imperfezioni nella convergenza dei treni di ruote, leggere correzioni allo sterzo da parte del guidatore, tentativi di sorpasso, dissesto stradale e molte altre. Un'autovettura che si spostasse da ovest verso est seguendo una strada perfettamente parallela all'equatore terrestre difficilmente riuscirebbe a mantenere una direzione esattamente pari a 0° ma misurerebbe nel tempo dei valori che si discostano leggermente da esso. Oltre a questi fattori si deve sopratutto tenere conto di inevitabili errori introdotti dai sensori utilizzati per monitorare lo stato del veicolo. È opportuno quindi prevedere uno o più meccanismi per compensare queste oscillazioni indesiderate ed evitare dei falsi negativi nel momento in cui la direzione di un veicolo viene confrontata con un angolo di riferimento o con la traiettoria di un altro nodo. Una prima semplice ma efficace accortezza è quella di mantenere per un opportuno periodo di tempo una storia dei precedenti campionamenti relativi alla direzione del veicolo e di integrarli effettuandone una media circolare[5], in questo modo è possibile contenere buona parte dei disturbi introdotti dalle imprecisioni degli strumenti di misura[6] o da leggeri cambiamenti di traiettoria.

Una soluzione aggiuntiva al problema viene presentata in Figura 2.3: ogniqualvolta sia necessario confrontare tra loro due misure di orientamento (ad esempio per verificare se due veicoli stanno procedendo nella stessa direzione) queste si possono considerare coincidenti anche a fronte di una lieve differenza, purché il loro disallineamento non superi una soglia stabilita a priori. Se il veicolo in alto a destra nella figura viaggia con una direzione media di 20°, questo potrà interagire anche con altri nodi che procedono con un orientamento che non si discosta di oltre 5° in valore assoluto dal suo. Una coppia di valori di direzione che mantengono questa proprietà viene definita *conformante*. Nella soluzione sviluppata questo importante accorgimento è stato utilizzato in diverse occasioni, in particolare per consentire ai diversi protocolli di continuare a funzionare anche in caso di percorsi leggermente

[5]Per maggiori dettagli sul concetto di media circolare si consulti `http://en.wikipedia.org/wiki/Mean_of_circular_quantities`.

[6]Nell'eventualità in cui tali sensori risultassero afflitti da errori con distribuzione normale, la correzione dovuta alla media circolare risulta maggiormente efficace.

irregolari.

Il concetto di direzione conforme è particolarmente utile non soltanto come semplice strumento per individuare e selezionare quei veicoli che rispettano un particolare senso di marcia ma anche come tecnica con la quale è possibile realizzare una comunicazione di tipo multicast tra i diversi nodi presenti nell'ambiente. Un qualsiasi dispositivo o interfaccia che intende scambiare informazioni con un insieme di nodi non necessariamente noti a priori, potrebbe imporre che la direzione attuale dei destinatari debba sottostare a precise condizioni, quali ad esempio la conformità a un gruppo che procede in un dato senso di marcia. In questo modo è infatti possibile circoscrivere lo scambio di informazioni tra un nodo leader e i soli veicoli che condividono la direzione prevista dal suo gruppo. Questa tecnica di selezione dei destinatari di un messaggio verrà applicata sistematicamente nell'implementazione dei diversi protocolli realizzati.

2.2.3 Ciclo di vita di un gruppo

Concludiamo la trattazione sulle modalità impiegate dai nodi del sistema per la formazione e la gestione dei gruppi con un rapido riassunto delle scelte effettuate e con un approfondimento sul loro ciclo di vita complessivo.
I dispositivi interessati a raccogliere nel tempo una base di conoscenza sulle proprietà del flusso locale di autovetture dispongono di una lista predefinita di valori di direzione (misurati in gradi d'arco) per le quali si aspettano di ricevere informazioni provenienti da particolari nodi eletti al ruolo di leader di gruppo. I dati che ogni gruppo trasmette alle RSU riassumono lo stato dei veicoli di cui è composto e possono includere nozioni sul numero di veicoli appartenenti al plotone, la loro velocità media, minima o massima, la densità media rilevata dai singoli membri, l'estensione massima del gruppo o qualsiasi altra caratteristica che contribuisca a delineare le condizioni del traffico stradale in quel punto.

Per realizzare concretamente questo meccanismo, ogni RSU trasmette un segnale di *beacon* a intervalli regolari e ravvicinati il quale descrive la sua identità e le direzioni alle quali è interessata. Qualsiasi nodo *idle* che rice-

ve tale segnale confronta la lista dei valori indicati con la propria direzione media attuale e se quest'ultima risulta conformante esso si attiva per quel senso di marcia. È necessario quindi che i nodi attivi giungano al più presto alla formazione di un plotone secondo le modalità sopra descritte e inizino a raccogliere dati da sintetizzare e da consegnare in seguito alle unità infrastrutturali. A tale scopo essi devono eleggere un leader che si preoccupi di coordinare i diversi membri del gruppo. Siccome plotoni con un maggior numero di membri sono generalmente in grado di raccogliere e integrare un quantitativo di dati superiore e rappresentano più concretamente lo stato del traffico se la densità è elevata, è importante assicurarsi la formazione di plotoni di grandi dimensioni. Questo è reso possibile guidando la scelta del group leader in modo da favorire l'elezione di quei veicoli che vedono all'interno della loro portata il più alto numero di nodi conformanti alla direzione richiesta dalla RSU.

Se si assume l'ipotesi che in assenza di una RSU i nodi viaggino in uno stato di completa inattività, questi si ritroveranno privi di una qualsiasi forma di conoscenza dell'ambiente circostante nel momento in cui saranno chiamati a individuare un nodo adeguato per il ruolo di leadership, ovvero quello in grado di comunicare con il maggior numero di veicoli. Sotto tali condizioni, una strategia piuttosto semplice con la quale essi possono accordarsi è quella di provvedere ad una sorta di asta nella quale i diversi veicoli si propongono come candidati al ruolo di coordinatore: ciascun nodo che intende partecipare all'elezione invia un messaggio ai nodi circostanti indicando la propria identità, la direzione per la quale intende formare il gruppo e il numero di veicoli che esso è in grado di raggiungere. Quest'ultima informazione è ottenibile mediante una prima fase nella quale i diversi partecipanti segnalano la propria presenza e la propria identità attraverso un messaggio multicast indirizzato a tutti i nodi che condividono la stessa direzione prevista per il gruppo che intendono formare. Superate queste due fasi, si ottiene idealmente una conoscenza distribuita dei veicoli circostanti e del loro vicinato ed è possibile individuare un leader idoneo per il gruppo selezionando quel nodo che risulta capace di coordinare un maggior numero di membri[7]. Questa

[7]In caso di parità tra più veicoli e in mancanza di ulteriori esigenze, si può scegliere in

tecnica di elezione risulta piuttosto semplice da realizzare ma comporta lo scambio di un quantitativo di messaggi considerevole tra nodi che tendono a concentrarsi in un punto e può presentare degli svantaggi nel momento in cui la densità dei veicoli in avvicinamento alla RSU supera una certa soglia critica. Se fosse invece possibile assumere che i nodi non ancora attivati da un dispositivo di infrastruttura possano comunque scambiarsi occasionalmente dei semplici messaggi allo scopo di continuare a monitorare le caratteristiche dell'ambiente circostante, essi riuscirebbero a sfruttare tale conoscenza per individuare un leader in maniera più semplice e con uno scambio di dati decisamente più contenuto. Anche questa soluzione non è però esente da difetti dal momento in cui il meccanismo utilizzato per mantenere coscienza dei nodi vicini richiede necessariamente l'invio continuo di messaggi broadcast, anche nei momenti in cui tali informazioni non sono strettamente necessarie.
Queste due diverse tecniche di raggruppamento si differenziano dal fatto di adottare dei processi di tipo *reattivo* o *proattivo* per arrivare a individuare il giusto candidato al ruolo di leadership. Nella soluzione proposta entrambe vengono realizzate in maniera indipendente e verranno in seguito approfondite, discusse e confrontate tra loro.

Indipendentemente dal metodo utilizzato per l'elezione di un nodo leader, una volta attivato questo deve preoccuparsi di controllare continuamente le condizioni dei propri nodi membri per assicurarsi della loro presenza e raccogliere da essi informazioni riguardanti la loro velocità attuale, la densità che essi percepiscono localmente, la loro direzione, etc . . . Una maniera particolarmente semplice ed economica per realizzare questa funzionalità è l'utilizzo di regolari beacon inviati dal leader ai propri membri, questi ultimi rispondono alla richiesta inviando un messaggio contenente la loro identità e i dati recentemente rilevati. In questo modo il responsabile di un gruppo può continuare nel tempo a verificare la presenza dei propri sottoposti e contemporaneamente ricevere da essi le informazioni necessarie a svolgere il suo operato. Se un veicolo di un gruppo – incluso il leader stesso – rileva che la propria direzione non è più conforme a quella prevista dalla RSU deve necessariamente uscire dal plotone e ritornare a uno stato di quiescenza. Ovviamente, se è il leader

maniera indifferente uno qualsiasi tra essi.

stesso a cambiare percorso o a perdere per qualche motivo la propria connettività, conviene che i nodi rimasti orfani procedano al più presto all'elezione di un nuovo gestore. Se il leader di un plotone non riceve alcuna risposta da parte dei suoi membri può comunque mantenere il suo stato di coordinatore e attendere eventualmente che altri nodi si uniscano ad esso in futuro. Sono dunque perfettamente ammissibili dei gruppi composti da un solo individuo, condizione che può manifestarsi con frequenza in caso di un flusso di traffico contenuto. Si prevede infine che qualsiasi nodo in stato idle che avverte la presenza di un gruppo dotato di direzione compatibile con quella che sta attualmente percorrendo possa effettuare l'accesso a quel plotone, a patto che venga garantita la visibilità del suo leader. È altresì prevista una funzionalità opzionale (attiva di default) mediante la quale il singolo nodo membro può autonomamente uscire dal gruppo a cui appartiene nel momento in cui rileva che la distanza dalla RSU è in aumento e che quindi tale veicolo ha già superato l'incrocio stradale che si vuole monitorare; ovviamente il leader di quel gruppo viene direttamente informato di tale evento ed esclude il nodo in oggetto dalle sue successive statistiche. Un'ulteriore estensione prevede la possibilità che un gruppo composto dal solo leader e sprovvisto di membri possa in qualsiasi momento decidere di cessare il proprio operato e unire l'unico nodo a sua disposizione ad un altro plotone conformante in qualità di semplice membro (*group merging*), posto il fatto che il leader di tale gruppo sia raggiungibile e operativo.

Tutti questi accorgimenti si rendono necessari dal momento che ci troviamo ad operare in scenari dove la mobilità dei singoli nodi è piuttosto elevata e non vi è garanzia alcuna – ed è anzi piuttosto improbabile – che una volta conclusa la fase di formazione di un gruppo la sua conformazione rimanga invariata e i diversi veicoli che lo compongono continuino a mantenere nel tempo la stessa posizione relativa e una direzione sempre conforme a quella prevista. Un plotone di veicoli è dunque un'entità dinamica e in continua evoluzione, in cui gli elementi che lo costituiscono possono unirsi, aggiungersi e disgregarsi a seconda del percorso che stanno seguendo e della propria posizione corrente. L'essenza principale di un gruppo nel particolare contesto in cui operiamo è quella di differenziare i diversi veicoli in avvicinamento a

dei punti di smistamento critici a seconda della loro direzione e promuovere la loro comunicazione allo scopo di raggiungere una forma di conoscenza sufficientemente accurata dello stato del flusso di vetture in movimento secondo direzioni prestabilite. In altri domini applicativi l'entità di gruppo può essere opportunamente modificata o ampliata per adattarsi ad esigenze diverse o più sofisticate ma in grado comunque di trarre vantaggio dallo stesso principio ispiratore.

Nell'intervallo di tempo che va dall'avvenuta formazione di un gruppo alla sua chiusura, il leader si preoccupa di inviare alla RSU, in istanti più o meno equidistanziati, un aggiornamento sul proprio stato e sulle statistiche raccolte fino a quel momento dall'intero gruppo. In questo modo le unità dell'infrastruttura stradale possono godere di informazioni aggiornate sui plotoni attivi in un dato momento e possono oltretutto disporre di una storia di campionamenti per ciascuno di essi. Questa serie di informazioni può risultare talvolta indispensabile per poter applicare delle logiche di gestione del traffico sofisticate e in grado di differenziarsi in base all'evoluzione delle condizioni del traffico nel tempo.

Il ciclo di vita di un gruppo termina nel momento in cui il gruppo raggiunge una distanza sufficientemente piccola dalla RSU ed è necessario procedere alla sua chiusura dal momento che, superato l'incrocio stradale, i diversi veicoli del gruppo inizieranno a percorrere strade differenti e il concetto di gruppo per come è stato definito viene del tutto a mancare. Le modalità con cui questo evento viene innescato possono essere molteplici: si potrebbe individuare un punto centrale per il gruppo considerando le posizioni dei diversi membri e utilizzarlo come riferimento. Questo comporterebbe però un problema aggiuntivo se si considera che non vi è sicurezza che il leader del gruppo venga posizionato al centro o comunque mantenga nel tempo una posizione centrale e potrebbe quindi uscire dall'incrocio prima di tale ipotetico baricentro. Per non introdurre ulteriori complessità si è deciso di adottare una soluzione decisamente semplice ma in grado di fornire comunque buoni risultati: si stabilisce che il leader tenga costantemente monitorata la propria distanza dalla RSU e inneschi il processo di chiusura per il suo gruppo non appena questa scende sotto una certa soglia. Una volta che tale condizione

viene soddisfatta il gruppo ha di fatto concluso il suo compito e i veicoli che lo compongono possono tornare a uno stato di quiescenza, evitando ovviamente di riattivarsi fintanto che il dispositivo che hanno appena servito rimane nella loro portata. Anche la RSU viene informata dell'avvenuta terminazione del plotone, in modo che possa adeguatamente tenere traccia dei gruppi che terminano correttamente il proprio ciclo di vita e differenziare questi ultimi da altri gruppi che possono invece cessare le comunicazioni in maniera anomala.

La soglia esatta per la chiusura di un gruppo deve essere però valutata con attenzione: se questa è particolarmente elevata il gruppo cessa di esistere quando i suoi membri si trovano ad essere ancora piuttosto distanti dalla RSU e potrebbero piuttosto continuare a raccogliere informazioni sensibili; viceversa, se un plotone attende eccessivamente prima di procedere alla sua chiusura, alcuni veicoli posizionati nella frontiera più vicina all'incrocio potrebbero superare in anticipo la RSU, con il conseguente rischio che vengano contati anche se di fatto non sono più membri del gruppo. Variando adeguatamente (eventualmente anche in maniera dinamica) tale soglia è possibile adattarsi a diverse necessità e controllare in maniera sufficientemente precisa la *freshness* e l'affidabilità delle informazioni pervenute alle diverse RSU. I molteplici dettagli relativi alle modalità con cui sono state realizzate le specifiche introdotte in questo paragrafo verranno discussi in maniera approfondita nei capitoli 3 e 4.

2.3 Classi di veicoli

Abbiamo già avuto occasione di ricordare come nei casi d'uso reali i molteplici dispositivi preposti alla comunicazione fra i nodi e gli strumenti utilizzati per rilevare le grandezze fisiche di interesse siano estremamente eterogenei tra loro e possano fornire risposte diverse a fronte di identici stimoli dati dall'ambiente. Più concretamente, si prevede che i diversi sensori utilizzati per registrare posizione, direzione e velocità di un veicolo si comportino con una precisione e un'accuratezza diverse rispetto a quelle di analoghi sensori presenti in veicoli diversi. Anche gli stessi componenti per la trasmissione dei dati – pur ricorrendo allo stesso standard per le comunicazioni wireless

– potranno reagire con diversi tempi di latenza, performance leggermente dissimili e raggi di copertura di elevata variabilità.

Occorre inoltre notare che gli studi su qualsiasi tipo di tecnologia VANET partono dal presupposto che esista almeno una certa percentuale di veicoli effettivamente dotata di sistemi in grado di rilevare la propria posizione e di comunicare con altri nodi vicini. Nonostante la prima tipologia di sensori sia ormai molto diffusa grazie all'enorme successo di soluzioni di navigazione assistita, i dispositivi per la comunicazione V2X hanno una penetrazione di mercato al momento irrisoria e gli investimenti da parte delle principali case automobilistiche risultano ancora contenuti e limitati prevalentemente a prototipi, sperimentazioni locali e ricerca. Nell'attesa che questa strumentazione raggiunga un adeguato livello di maturità è possibile ipotizzare l'utilizzo dei già citati e diffusissimi *smart device* come interfacce di comunicazione e semplici apparecchiature di geolocalizzazione. Ci si aspetta tuttavia che le prestazioni fornite da tali dispositivi siano piuttosto inferiori in termini di accuratezza rispetto a quelle ottenibili dalle soluzioni specific-purpose appena suggerite.

Naturalmente, le prestazioni dei singoli componenti responsabili del monitoraggio dello stato del veicolo e della comunicazione tra i diversi nodi incidono enormemente sulla resa complessiva dei protocolli sviluppati e sulla qualità dei dati ricevuti dalle RSU. Per tenere conto di questa elevata variabilità e per poter studiare in che modo e con quale rapporto le risposte del sistema variano a seconda dei sensori e delle interfacce utilizzate sono state previste tre diverse classi di veicoli, differenziate in base alle loro capacità e all'insieme dei dispositivi offerti:

Shadow vehicle Veicoli che non possono avvalersi di un'interfaccia di comunicazione wireless e risultano pertanto invisibili al sistema e ai protocolli. Pur disponendo di eventuali sensori utili per ricavare la loro posizione, velocità e direzione, non riescono a contribuire alla creazione e al mantenimento di informazioni utili sullo stato del traffico. Si tratta del tipo di autovetture oggi principalmente in uso.

Mid-class vehicle Vetture che non dispongono nativamente di un'interfaccia di comunicazione ma in cui l'utente sopperisce a tale mancanza utilizzando un dispositivo smart personale in grado di stimare la posizione del veicolo e di comunicare con altri nodi purché dotati di analoghe interfacce. Ci si aspetta che la precisione degli strumenti e la portata dei segnali scambiati con altri veicoli siano sensibilmente inferiori rispetto a quelle offerte da vetture di classe full.

Full-class vehicle Veicoli di ultimissima generazione dotati di sensori di geolocalizzazione di precisione elevata e con sistemi di comunicazione *car-to-car* embedded e di buona portata. Questa classe risulta essere ideale per i protocolli sviluppati e non soffre delle limitazioni presenti nei mid-class vehicles, tuttavia le sue proprietà appaiono ancora leggermente irrealistiche a causa della loro scarsa diffusione nel mercato attuale.

2.4 Nodi marker

In buona parte degli scenari nei quali si prevede di applicare il sistema appena descritto, riuscire ad avere una serie di informazioni da parte di ciascun gruppo raccolte negli ultimi 150 - 180 metri che intercorrono prima del punto di snodo stradale è del tutto sufficiente per poter operare delle politiche intelligenti e reattive per il controllo dinamico del flusso dei veicoli. In altri casi d'uso più avanzati potrebbe essere necessario ottenere dei dati relativi a una porzione di tragitto più ampia, magari per tenere meglio sotto controllo eventuali code che possono estendersi anche oltre alcune centinaia di metri di distanza dalla RSU, oppure per la raccolta di informazioni più approfondite relative al traffico o ancora, per suggerire ai singoli veicoli la scelta di percorsi alternativi in caso di congestione, lavori in corso o incidente. In ciascuno di questi ipotetici scenari si ha quindi l'esigenza di attivare dei nodi – o comunque di trasferire dei segnali da e verso essi – che risultano troppo lontani da una RSU per poter comunicare direttamente con essa.

Nel campo della ricerca su applicazioni VANET, la letteratura offre molte soluzioni che consentono di mettere in comunicazione due o più nodi che sono reciprocamente impossibilitati a comunicare a causa della loro distanza. In generale tali strategie risultano molto valide ma come è già stato evidenziato nel paragrafo 2.2.1 queste risultano nel complesso piuttosto difficili da implementare e possono talvolta risultare anche eccessivamente sofisticate per le necessità relativamente semplici sopra esposte. Si vuole offrire in questa sede una semplice soluzione di compromesso per tali esigenze in grado di sfruttare i concetti introdotti in questo capitolo e di integrarsi facilmente all'interno del sistema sviluppato.

Nelle normali condizioni di utilizzo i singoli veicoli raccolgono informazioni sull'ambiente circostante e comunicano tali dati ad altri loro pari solo dopo essere stati inclusi all'interno di un gruppo[8] e quindi solo dopo che essi o dei nodi sufficientemente vicini hanno ricevuto un segnale da una RSU contenente la richiesta di procedere alla formazione di un plotone per quella direzione. Se si desidera avere la possibilità di consegnare tale richiesta anche a quei veicoli che si trovano ancora molto lontani da essa è possibile ricorrere a dei nodi intermediari denominati *marker* in grado di rimbalzare il segnale della RSU e di ripeterlo verso altri dispositivi non ancora in grado di percepirlo direttamente. Sfruttando uno o più marker è possibile quindi anticipare la formazione dei gruppi e aumentare la portata delle unità infrastrutturali ove previsto. Come conseguenza, il flusso dei dati che verrà raccolto da una RSU i cui segnali di beaconing vengono estesi per mezzo di tali intermediari è relativo a un tratto stradale molto più esteso e pertanto risulterà più accurato, completo e affidabile a seconda dei diversi casi d'uso possibili.

Al fine di individuare in modo semplice ed efficace quale nodo può essere assunto al ruolo di marker viene considerata la sua posizione relativa rispetto alla RSU stessa e nei confronti di altri veicoli vicini eventualmente già attivati come marker. Ogni volta che un nodo – indipendentemente dal suo stato corrente – riceve un beacon da una RSU può guadagnare il ruolo di marker se

[8]Nel paragrafo 2.2.3 è stato proposto uno strumento di tipo proattivo che prevede la comunicazione tra nodi anche in stato idle. Tale strategia è comunque finalizzata alla semplice mappatura dei nodi circostanti per facilitare le successive operazioni di group formation e non è da intendersi come raccolta di informazioni sulle condizioni del traffico.

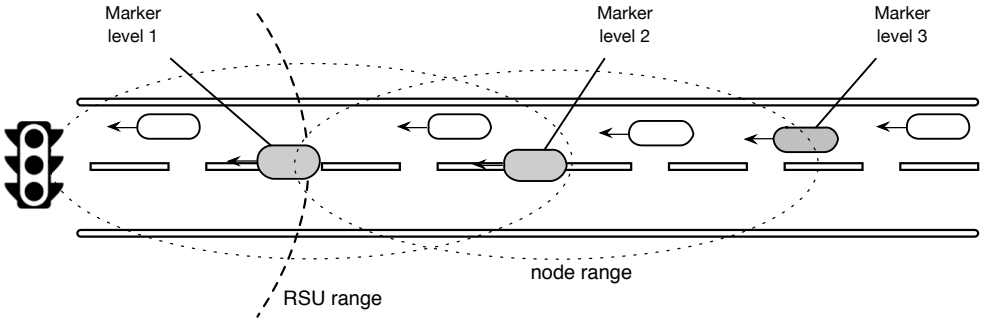

Figura 2.4: Estensione del range RSU mediante nodi marker (in grigio). Si assegna un livello a ciascun in base alla crescente distanza con la RSU.

esso non è già ricoperto da altri veicoli sufficientemente vicini e se la distanza dalla RSU supera una certa soglia prestabilita. Fintanto che mantiene il proprio ruolo, un marker effettua il broadcast ad intervalli regolari dello stesso messaggio ricevuto in origine dalla RSU e può quindi attivare altri nodi che non hanno potuto ricevere il beacon originale. Per estendere ulteriormente la portata dei messaggi è possibile ricorrere a più nodi marker attivi contemporaneamente per una stessa direzione, come rappresentato nella Figura 2.4, dove viene schematizzata la strategia appena esposta. Nel rettilineo stilizzato sono appunto presenti più nodi marker (in grigio) che ritrasmettono il segnale della RSU fino a tre hop di distanza. Per distinguere i diversi ripetitori in una direzione, viene ad essi assegnato un livello crescente in proporzione alla distanza crescente dall'incrocio. Un marker di primo livello è tale se riesce a comunicare direttamente con la RSU che lo ha attivato, mentre un ripetitore di secondo livello viene attivato da un marker di primo livello e così via fino a quando non si raggiunge un livello massimo o un tetto di distanza limite dalla RSU.

Occorre tuttavia ricordare che un marker non è un'unità infrastrutturale fissa ma è a tutti gli effetti un utente della strada e come tale può spostarsi in maniera non sempre predicibile e andare incontro a malfunzionamenti e disconnessioni; quello dei marker è quindi un ennesimo meccanismo di tipo *best-effort* la cui resa è fortemente dipendente dalle caratteristiche contingenti del traffico e dell'ambiente circostante. Per tali motivi e in particolari modo

per evitare che il sistema degeneri in un semplice quanto indesiderato flooding di messaggi provenienti dalla RSU da parte di un insieme incontrollato di ripetitori, si rende necessario porre dei limiti ben precisi alla loro creazione e stabilire in quali condizioni essi devono cessare il proprio operato. Appare anzitutto evidente come sia del tutto inutile disporre di marker troppo vicini tra loro in quanto i loro raggi di portata andrebbero a sovrapporsi e non offrirebbero alcun guadagno in termini di estensione del campo operativo delle RSU. Si stabilisce quindi che due nodi ripetitori non possono avvicinarsi oltre a un limite prestabilito pena la rimozione di uno di essi dal ruolo di marker. Nonostante non sia prevista alcuna forma di interazione tra ripetitori di una stessa direzione è quindi fondamentale che ciascuno di essi mantenga continuamente aggiornata una lista di altri nodi marker in quella direzione comprensiva delle relative posizioni. Tale richiesta può essere facilmente soddisfatta sfruttando gli stessi messaggi broadcast che ciascun ripetitore produce a intervalli costanti.

Capitolo 3

Architettura del sistema

In questo capitolo si vogliono presentare e discutere approfonditamente i dettagli implementativi e le scelte adottate per realizzare concretamente il sistema di Intelligent Transportation – da qui in avanti abbreviato semplicemente in ITS – presentato nel precedente capitolo. Verrà posta particolare attenzione all'organizzazione architetturale impiegata, alla logica di comportamento dei singoli protocolli, ai messaggi che essi scambiano tra loro e alle modalità di funzionamento dei componenti basilari del sistema.

Prima di esplorare nel dettaglio i diversi aspetti architetturali della soluzione realizzata, verranno brevemente esposti e motivati i principali strumenti adottati per lo sviluppo e gli ambienti di simulazione impiegati per l'elaborazione e il collaudo dei sorgenti prodotti. Si rimanda la lettura al capitolo 5 per qualsiasi informazione relativa alle modalità di esecuzione e raccolta dei risultati sperimentali o agli specifici parametri di simulazione utilizzati nei diversi test.

3.1 Strumenti e ambienti di simulazione

Lo sviluppo di nuove tecnologie software e hardware ricorre sempre più spesso all'utilizzo di appositi pacchetti dediti alla simulazione degli specifici scenari di utilizzo e dei casi d'uso previsti per un particolare dominio applicativo. Tali strumenti permettono infatti di studiare nel dettaglio il comporta-

mento e la risposta del sistema in base a molteplici stimoli in ingresso. A tale scopo il progettista, dopo aver definito e codificato una versione sperimentale della logica di comportamento dei vari elementi costituivi dell'applicazione in esame, ne effettua il deploy in un ambiente software che si occupa di simulare in maniera più o meno realistica lo scenario reale per il quale il sistema è stato concepito.

Questa modalità di progettazione si contrappone nettamente a quella più tradizionale che prevede una diretta sperimentazione nel reale ambiente di utilizzo e – specialmente nel caso di sviluppo di applicativi hardware – risulta considerevolmente più costosa in termini di tempo e risorse necessarie. Nonostante l'emergere di soluzioni relativamente economiche e accessibili a un'utenza sempre più vasta [Sad11; SK10], la progettazione di sistemi complessi, destinati a un pubblico vasto e a scenari di utilizzo differenziati non può prescindere dall'utilizzo di specifici ambienti di simulazione. Esistono inoltre casi nei quali la sperimentazione per mezzo di strumenti virtuali è l'unica opzione inizialmente possibile: nel campo delle reti tra sistemi mobili ad esempio è richiesta non solo la codifica degli specifici protocolli, algoritmi e framework che costituiscono il fondamento principale della tecnologia che si intende studiare, ma anche l'installazione di quest'ultima su un vasto numero di dispositivi – spesso eterogenei e dislocati nello spazio – per poter ottenere un riscontro significativo del loro comportamento. È evidente come già il semplice costo per l'acquisto o per la produzione dell'hardware necessario a tale sperimentazione risulti proibitivo per la maggior parte dei progettisti; anche il tempo necessario per la raccolta dei risultati nei diversi test, la ricodifica dei sorgenti in caso di errori o modifiche, il successivo deploy sui molteplici dispositivi e la nuova messa a punto dell'ambiente di sperimentazione comportano un ciclo di sviluppo estremamente lento e dispendioso.

Per i motivi appena esposti l'implementazione e il testing dell'architettura di ITS introdotta nel capitolo precedente sono stati realizzati all'interno di un sistema di simulazione di rete. Esistono numerosissime soluzioni software in grado di fornire un ambiente virtuale talvolta anche molto complesso per la sperimentazione di tecnologie di comunicazione, applicazioni distribuite, protocolli di routing o di trasporto e molto altro. Alcune soluzioni proprietarie

– seppure piuttosto costose – raggiungono un elevato livello di sofisticazione e garantiscono ottime performance per le esigenze più professionali; tra queste è possibile annoverare [13b] oppure [13g]. Esiste inoltre un numeroso insieme di applicazioni open source, spesso anche piuttosto avanzate, che consentono comunque di ottenere ottimi risultati senza costi aggiuntivi e sono supportate da vaste comunità di sviluppatori e utenti che ne garantiscono un continuo sviluppo e contribuiscono in molti casi a fornire una curva di apprendimento meno ripida rispetto alle loro controparti proprietarie; alcuni tra gli strumenti più diffusi in questo campo sono ns-2 [11], ns-3 [13c], OMNet++ [13f] oppure gns3 [13a]. Un ulteriore vantaggio dato dalle soluzioni di tipo open source deriva dalla possibilità di studiare e modificare direttamente il codice sorgente del pacchetto di simulazione o dei suoi moduli accessori: questo consente agli utenti più avanzati di comprendere appieno il funzionamento intimo del nucleo di simulazione e della logica implementata per i vari stack protocollari e – soprattutto – permette di intervenire in prima persona sui sorgenti sia per implementare nuove funzionalità e modelli sperimentali, sia per personalizzare il comportamento del pacchetto secondo le proprie particolari esigenze.

3.1.1 NS-3

Per questo progetto di tesi si è scelto di utilizzare ns-3.18 come principale strumento di simulazione di rete e piattaforma per lo sviluppo dei vari componenti e dei protocolli di comunicazione. In generale l'architettura dettagliata di ns3 e dei suoi numerosissimi componenti è decisamente complessa e una sua esposizione esaustiva esula dagli scopi di questo testo. Considerando che il progetto realizzato utilizza un sottoinsieme adeguatamente ristretto delle molteplici funzionalità offerte dal pacchetto è sufficiente limitarsi in questa sede a una veloce presentazione della sua organizzazione complessiva. Nel seguito del capitolo si avrà cura di approfondire ove necessario eventuali caratteristiche utili alla completa comprensione degli argomenti trattati. Per una descrizione completa delle specifiche di ns-3 è possibile fare riferimento alla sua documentazione ufficiale [13e] e alle guide disponibili per i suoi

numerosi moduli [13d].

Ns-3 è un simulatore ad eventi discreti realizzato in C++ e finalizzato allo studio, allo sviluppo e alla messa a punto di soluzioni e protocolli di comunicazione tra nodi e apparecchiature di rete. Mediante esso è possibile definire le proprietà e le interfacce dei nodi presenti nell'ambiente, definire una topologia di rete che mette in comunicazione tali dispositivi, installare e configurare gli opportuni stack protocollari che si intendono utilizzare per fornire ai nodi i diversi servizi previsti dal modello ISO/OSI [Tan02] e infine aggiungere un eventuale insieme di applicazioni che generano traffico all'interno della rete e processano i dati trasportati per mezzo di pacchetti. Per realizzare tutte queste (e molte altre) funzionalità vengono create e connesse tra loro numerose classi di oggetti che ripropongono il comportamento delle principali entità fisiche presenti nel mondo reale (nodi fissi o in movimento, mezzi per la trasmissione dei segnali, dispositivi di rete, edifici, ...), codificano gli algoritmi e i protocolli presenti in letteratura (stack internet, routing, applicazioni di rete, ...) e simulano in maniera sufficientemente realistica l'effetto di alcuni fenomeni fisici di interesse (segnali elettrici ed elettromagnetici, propagazione, errori, consumi energetici, ...); un utente può limitarsi ad utilizzare i diversi moduli messi a disposizione dal pacchetto software, modificare i sorgenti di alcuni di essi secondo le proprie necessità oppure crearne di completamente nuovi.

Una visione ad alto livello dell'organizzazione complessiva di ns-3 è data in Figura 3.1, dove è possibile osservare i principali livelli architetturali e una descrizione dei servizi offerti da questi ultimi agli strati superiori della gerarchia. In figura è anche presente una schematizzazione dei concetti maggiormente utilizzati all'interno della piattaforma e di come questi si traducono nelle classi di oggetti più importanti usate per realizzare concretamente il trasferimento di un pacchetto di dati tra applicazioni che eseguono su una coppia di nodi. La classe di oggetti fondamentale in ns-3 è appunto definita come `Node` e rappresenta l'astrazione di un generico dispositivo collocato all'interno di una rete, ovvero un qualsiasi tipo di elaboratore – mobile o con posizione fissa – al quale vengono successivamente aggiunti ulteriori oggetti che ne stabiliscono le proprietà (posizione, velocità, autonomia, ... etc) e il

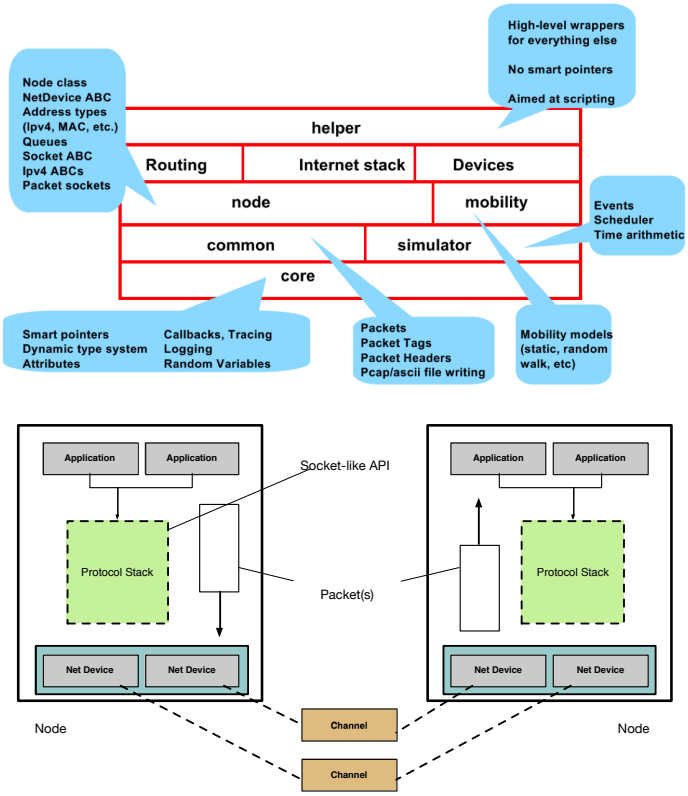

Figura 3.1: Struttura di alto livello di ns-3 e principali astrazioni utilizzate per realizzare lo scambio di pacchetti tra due nodi. Le applicazioni in ns-3 comunicano attraverso pacchetti processati da uno stack protocollare e trasmessi verso un canale fisico per mezzo di uno o più dispositivi di rete.

comportamento (device utilizzati, protocolli di comunicazione, applicazioni, modelli di mobilità, . . . etc). L'utente definisce pertanto uno scenario virtuale creando un nodo per ogni agente di rete che intende simulare e aggrega ad esso i molteplici oggetti messi a disposizione da ns-3, eventualmente personalizzandoli manualmente oppure servendosi di sottoclassi. Nel nostro caso, ogni veicolo presente nell'ambiente verrà rappresentato da un oggetto `Node` al quale verranno aggregati i diversi componenti che realizzano l'architettura del sistema sviluppato (vedi il paragrafo 3.2), i principali oggetti previsti normalmente in ns-3 per applicare una traccia di mobilità (vedi paragrafo 3.1.2) e infine quelli necessari per creare una connessione senza fili verso altri suoi pari.

Altre astrazioni particolarmente importanti in ns-3 sono quelle di *canale* e *dispositivo* di rete: esse infatti realizzano concretamente la trasmissione di pacchetti tra più nodi, simulando di fatto il livello più basso del modello ISO/OSI, ovvero quello fisico. Come si può osservare in Figura 3.1, il componente `Channel` è responsabile per il trasferimento dei singoli bit che compongono le trame di dati scambiate tra due interfacce di rete (`NetDevice`). Esso rappresenta quindi mezzi fisici di tipo point-to-point (tipicamente dei cavi di rete) o wired (canali 802.11, Bluetooth, WiMax, LTE, ... etc) e permette oltretutto di simulare dei disturbi nelle trasmissioni, ritardi, effetti di propagazione, sovrapposizione e collisione tra più trame. La classe di oggetti `NetDevice` imita il comportamento delle interfacce di rete comunemente usate per accedere al canale fisico e immettere o ricevere i segnali elettrici utilizzati per codificare la trasmissione. All'interno del progetto di tesi è stata utilizzata la sottoclasse `WifiNetDevice` per disporre dei servizi di un'interfaccia di rete wireless. Per dettagli tecnici e precisi sul motore di simulazione utilizzato da ns-3 per realizzare lo stack protocollare previsto dallo standard 802.11 è possibile consultare Lacage e Henderson [LH06].

3.1.2 SUMO

Un qualsiasi progetto di ricerca in ambito VANET non può prescindere dall'utilizzo di uno strumento in grado di simulare la mobilità di un insieme di veicoli all'interno di un reticolo stradale: le caratteristiche e la qualità delle trasmissioni wireless tra più nodi dipendono fortemente dalla posizione che essi assumono in ogni preciso istante di simulazione ed è quindi indispensabile poter conoscere nel dettaglio l'intero percorso seguito da qualsiasi veicolo simulato nell'ambiente. Questa informazione viene detta *traccia di mobilità* e consiste in una formale descrizione delle coordinate spaziali e della velocità assunte in ogni momento dai nodi presenti in una simulazione. All'interno di una traccia, il tragitto seguito da un qualsiasi veicolo viene strutturato come una collezione ordinata di *waypoint*, ciascuno dei quali specifica la sua posizione in un particolare istante di tempo, il vettore relativo alla sua velocità istantanea e altre eventuali informazioni accessorie (orientamento,

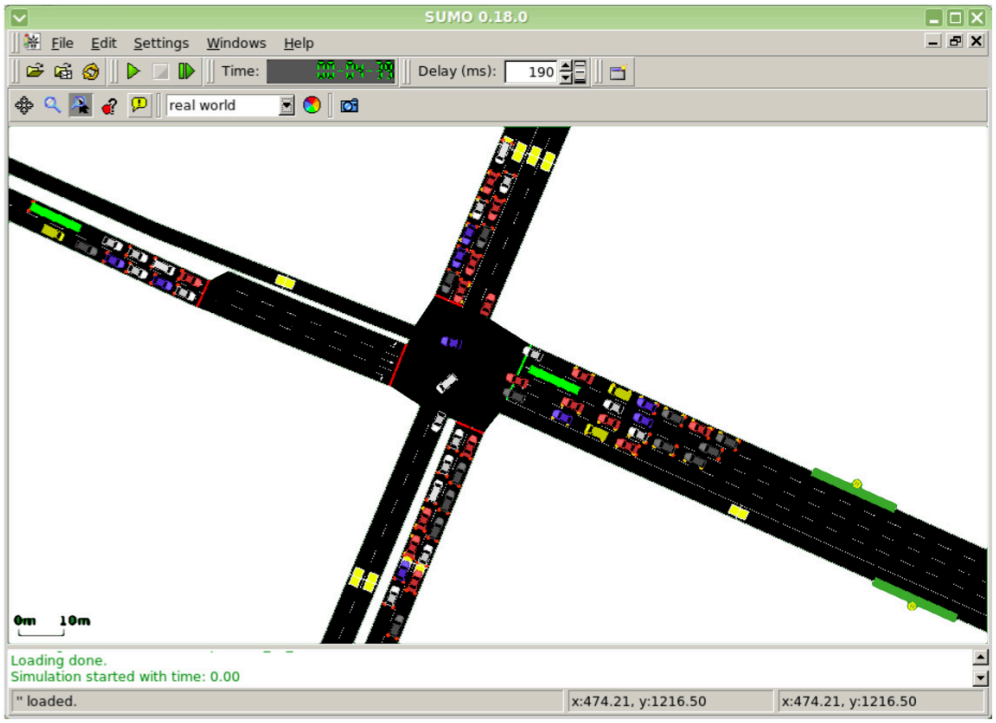

Figura 3.2: SUMO è il simulatore di traffico veicolare utilizzato per disporre di tracce di mobilità per i singoli veicoli in ns-3.

accelerazione, . . . etc). Sfruttando l'interpolazione di due waypoint adiacenti nel tempo è sempre possibile risalire all'esatta posizione di ciascun nodo in qualsiasi istante di tempo simulato.

Il semplice ricorso a ns-3 o a un qualsiasi altro simulatore di rete non è in genere sufficiente per ottenere delle tracce di mobilità verosimili in grado di rappresentare un flusso realistico di veicoli che si spostano da un punto all'altro di una mappa topografica. I modelli di mobilità standard inclusi nel pacchetto risultano infatti molto semplici e per nulla funzionali agli scopi del progetto in esame. Per rispondere a tale esigenza si è fatto ricorso a SUMO [Kra+12], un simulatore urbano open-source ad eventi discreti in grado di imitare in maniera estremamente realistica il comportamento di diversi tipi di automezzi che si spostano all'interno di una rete stradale; quest'ultima può essere interamente costruita dall'utente oppure ottenuta a partire da mappe realmente esistenti. Grazie al supporto nativo agli strumenti di rilevazione a

induzione (induction loop), a molteplici strumenti per il geoposizionamento e a svariati formati professionali, SUMO si è rivelata una soluzione particolarmente utile ed efficiente per ottenere e convertire alcune tracce di mobilità messe a disposizione dai progetti COLOMBO e iTetris. Per un approfondimento sulle diverse metodologie di simulazione dei veicoli su percorsi stradali virtualizzati e un confronto tra i software di simulazione maggiormente diffusi in ambito accademico è possibile consultare [HFB09].

Le simulazioni urbane utilizzate come riferimento per i vari test illustrati nel capitolo 5 sono state costruite sfruttando delle rilevazioni reali effettuate in aree urbane strategiche delle città europee di Norimberga e Bologna. Le zone coinvolte nel campionamento sono normalmente note per soffrire di problemi di congestione e riguardano un totale di 24 ore per ciascuno scenario. Selezionando di volta in volta un opportuno intervallo temporale da cui estrarre una traccia di mobilità compatibile con ns-3 si può disporre di tutte le diverse tipologie di traffico che è possibile incontrare durante un'intera giornata: da un flusso estremamente denso e rallentato relativo alle principali ore di punta a quello ridotto e scorrevole tipico delle ore notturne. Per ulteriori dettagli sugli scenari utilizzati come ambienti di test e sui parametri di simulazione in essi utilizzati si consultino i paragrafi 5.1 e 5.2.

3.2 Architettura generale

La realizzazione di un sistema software complesso richiede necessariamente un attento sforzo di analisi per poter individuare quali sono le caratteristiche fondamentali del progetto, le soluzioni più efficaci e meno costose per realizzarlo e le modalità con cui organizzarne l'architettura complessiva. Siccome i requisiti possono cambiare durante la stesura del codice oppure altri imprevisti potrebbero costringere a delle modifiche in fasi più avanzate, è particolarmente importante strutturare l'organizzazione del sistema nella maniera più modulare possibile, dividendo nettamente i compiti e le responsabilità dei diversi componenti e individuando le modalità più semplici per organizzarli in un'architettura funzionale e semplice.

La struttura di ns-3 viene sicuramente in aiuto a tale proposito se si considera che i maggiori sforzi durante la sua progettazione sono stati orientati verso la definizione di un sistema completamente modulare [LH06, p. 2]. Viene infatti privilegiato l'utilizzo di callback e object aggregation per evitare forti interdipendenze tra i componenti e permettere che questi possano essere aggiunti, rimossi o modificati persino durante il tempo di esecuzione. Il design di ns-3 ha fortemente influenzato anche l'architettura complessiva del sistema in esame: anche se particolarmente avanzato, il comportamento di un nodo di rete che comunica e scambia informazioni con altri suoi pari può essere sempre definito in termini di un complesso automa a stati finiti. L'insieme dei protocolli sviluppati ed esposti in questo capitolo si potrebbe quindi implementare come un unico componente monolitico che riceve in ingresso i messaggi ricevuti dagli altri nodi della rete, effettua eventuali cambiamenti di stato e produce all'occorrenza dei segnali in uscita. Una simile strategia guadagnerebbe sicuramente in termini di prestazioni e risulta infatti l'approccio principalmente utilizzato quando si ha l'esigenza di sintetizzare una logica complessa in un unico componente hardware compatto ed efficiente. Risulta tuttavia evidente come questa scelta possa rivelarsi del tutto controproducente nel momento in cui si desidera semplicemente arrivare a definire un prototipo del sistema da poter studiare e perfezionare: il punto di interesse principale di questo lavoro di tesi non è infatti l'ottenimento di un prodotto hardware o software finito e pronto per l'implementazione e l'utilizzo in larga scala ma piuttosto la realizzazione, lo studio e l'analisi di quell'insieme di protocolli di comunicazione tra veicoli introdotti nel capitolo precedente. Anche se la prospettiva di una futura sperimentazione reale è comunque tenuta in considerazione, occorre precisare che le soluzioni di seguito esposte sono state concepite in vista di una loro simulazione nello specifico ambiente di simulazione fornito da ns-3 e non sono in alcun modo da intendersi come una proposta di implementazione per un eventuale prototipo reale.

Piuttosto che produrre un unico macro-componente per l'intero protocollo di gestione del ciclo di vita di un nodo, dalla formazione di un gruppo allo scambio di informazioni con il leader fino alla chiusura del gruppo stesso, si è preferito suddividere tale logica in tanti componenti più semplici e indipen-

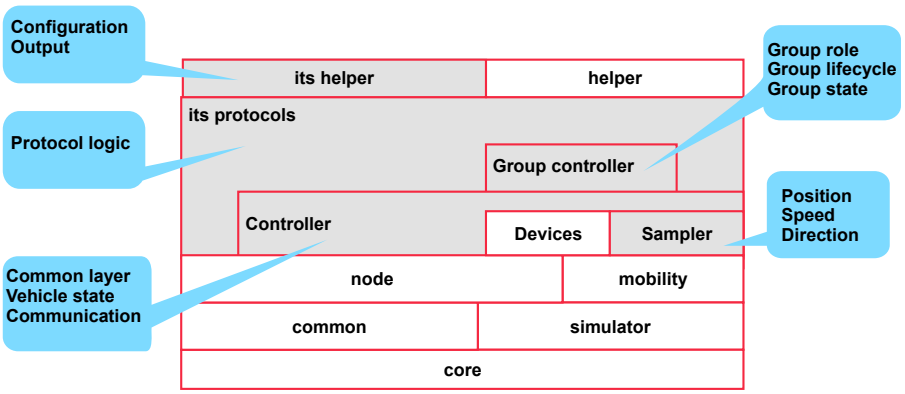

Figura 3.3: Organizzazione ad alto livello del modulo ITS sviluppato (riquadri in grigio) e integrazione con i componenti standard di ns-3.

denti, ciascuno di essi incaricato di occuparsi di un preciso aspetto funzionale del sistema. Questi moduli a granularità più fine vengono definiti *comportamenti* e derivano tutti da un'unica classe astratta di base `Behaviour` che ne formalizza l'interfaccia e le caratteristiche comuni. È quindi previsto un comportamento esclusivamente dedicato alla formazione dei gruppi, uno che definisce la logica dei nodi marker, uno che descrive il protocollo utilizzato dai nodi membri di un gruppo e così via. Tali classi di oggetti sono pensate per poter funzionare in modo completamente indipendente l'una dalle altre e per poter essere installate e attivate secondo necessità: se si intende concentrare lo studio sulla risposta dei singoli nodi al protocollo di group formation è possibile attivare esclusivamente il comportamento che realizza tale logica; se si vuole invece eseguire una simulazione dell'intero codice senza che vengano però utilizzati nodi marker si può disattivare il relativo comportamento e mantenere inalterati gli altri. Questa scelta permette quindi di concentrarsi sullo sviluppo e sul test di ciascun componente in maniera modulare e offre una notevole flessibilità ed espansibilità al sistema, potendo in qualsiasi momento abilitare e/o riconfigurare un qualsiasi set di protocolli o crearne dei nuovi partendo da zero. I diversi comportamenti sviluppati verranno descritti dettagliatamente nel capitolo 4, dopo che saranno stati chiariti gli aspetti riguardanti i componenti architetturali di più basso livello, indispensabili per il corretto funzionamento dei behaviour.

Confrontando le figure 3.1 e 3.3 si può osservare in che modo il modulo ITS si inserisce all'interno della struttura principale di ns-3 e come i suoi componenti vengono disposti e si interfacciano tra loro per svolgere le funzionalità indicate. Ciascuno dei blocchi evidenziati in Figura 3.3 verrà in seguito ampiamente presentato e argomentato. Per poter meglio cogliere l'organizzazione complessiva dell'architettura del progetto si noti come l'insieme che racchiude tutte le logiche dei protocolli sviluppati (*its protocols*) sia appoggiato su tre componenti fondamentali che forniscono servizi di importanza critica per l'intero sistema. `ItsController` è il controllore principale del sistema sviluppato e si preoccupa di fornire un livello di funzionalità di base utile a tutti gli strati superiori, nascondendo buona parte delle complessità di ns-3 per offrire una sorta di API comune ai componenti superiori. In particolare esso realizza le primitive per lo scambio e la ricezione dei messaggi e per il recupero delle informazioni di stato del veicolo. `ItsGroupController` è un secondo controllore esclusivamente dedicato alla gestione delle informazioni e delle operazioni che riguardano l'eventuale gruppo al quale un nodo può appartenere. `NodeSampler` infine è un componente di fondamentale importanza per il sistema in quanto si interfaccia al layer di mobilità di ns-3 per realizzare un continuo campionamento dei valori di posizione e velocità del veicolo introducendo degli opportuni errori casuali per simulare realisticamente la non perfetta precisione e accuratezza dei sensori che verrebbero utilizzati per recuperare tali informazioni.

La Figura 3.4 schematizza quanto esposto finora ed espone tutti i componenti che realizzano nel complesso la soluzione proposta. La classe `Node` presente in ns-3 costituisce il punto di partenza principale dell'architettura in quanto utilizzata per rappresentare sia le autovetture – indipendentemente dalla particolare tipologia di veicolo di cui si vuole disporre[1] – sia le unità infrastrutturali fisse. Queste ultime, trattandosi di nodi totalmente privi di mobilità e con un insieme di funzionalità radicalmente differente rispetto ai veicoli, sono caratterizzate da una struttura dei componenti semplificata: il controllore di gruppo e il campionatore di posizione, direzione e velocità sono del tutto assenti in quanto non necessari, mentre l'insieme dei comportamenti

[1] vedi par. 2.3

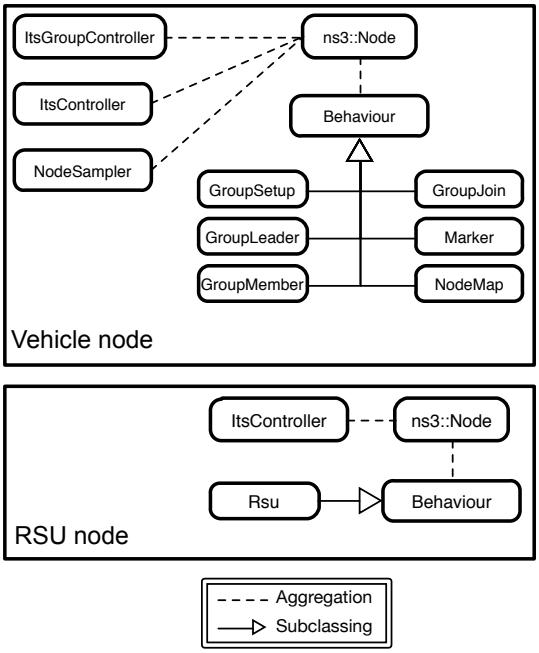

Figura 3.4: Architettura complessiva del modulo ITS. Ciascun componente viene aggregato alla classe `Node` e può interagire con gli altri mediante uso di callback o dipendenze dirette. La classe `Behaviour` viene ereditata da tutti gli oggetti responsabili delle varie logiche di protocollo. I nodi che rappresentano le singole RSU prevedono un'organizzazione semplificata rispetto ai nodi mobili.

installati prevede la sola classe `BehaviourRsu` che ne descrive autonomamente la logica. I normali nodi dispongono invece di tutti i componenti presentati in figura ad eccezione degli shadow vehicle che non dispongono di alcun modulo oltre al semplice oggetto *Node*: questa particolare classe vuole infatti simulare quei veicoli che non possiedono un sistema di comunicazione ITS e non possono pertanto partecipare ai protocolli di seguito descritti.

3.2.1 Controllore

L'interazione tra i diversi moduli raffigurati in Figura 3.3 avviene normalmente dall'alto verso il basso: le varie logiche di protocollo non comunicano direttamente tra loro essendo idealmente indipendenti ma fanno riferimento ai due controllori sottostanti per ottenere le informazioni e i servizi di cui hanno bisogno; anche il controllore di gruppo si rivolge quasi esclusivamente

al controllore principale, mentre `NodeSampler` rimane un componente piuttosto isolato dagli altri e si limita ad utilizzare il modulo di mobilità di ns-3 per svolgere il proprio compito.

Nonostante sia preferibile seguire l'organizzazione verticale appena esposta, si potrebbe occasionalmente incorrere in situazioni *cross-layer* dove è necessario che le logiche di più alto livello possano accedere a servizi e informazioni presenti a diversi strati più bassi dell'architettura senza mediazione da parte dei componenti intermediari. Per favorire queste eventualità ed evitare che l'interfaccia offerta dal controllore risulti troppo restrittiva e rigida, `ItsController` mette a disposizione un puntatore al nodo ns-3 sul quale esso è in esecuzione. Tale oggetto consente di sfruttare dei meccanismi di object aggregation per risalire ai molteplici moduli dell'architettura di ns-3 o agli altri componenti del progetto stesso.

Vista comunque l'importanza centrale che il controllore riveste per l'intera organizzazione dell'architettura in esame, verranno di seguito elencate e discusse le primitive e l'interfaccia messe a disposizione dei livelli superiori.

Setup e configurazione

Per poter configurare correttamente un nodo come entità del sistema ITS sviluppato occorre seguire una serie di passaggi che vanno dalla creazione del nodo stesso, all'installazione del controllore fino alla configurazione delle singole logiche di protocollo. Tutti questi passaggi sfruttano le funzionalità di aggregazione oggetti e impostazione degli attributi fornite dal pacchetto ns-3 [13e, p.39]. Per *attributo* di un oggetto si intende un qualsiasi tipo di variabile che rappresenta una sua proprietà e che si vuole poter visualizzare e manipolare dall'esterno. Ns-3 consente infatti di leggere e impostare il valore di un qualsiasi attributo in svariati modi: è possibile stabilirne un valore di default, modificarlo e leggerlo a run-time oppure parametrizzarlo mediante linea di comando o file di configurazione esterno. Questi strumenti vengono utilizzati per quasi tutti gli oggetti realizzati all'interno del progetto e consentono di trattare diversi valori chiave di quantità e tempo come parametri che è possibile variare da simulazione a simulazione senza ricompilare l'intero

Tabella 3.1: Attributi per la classe `ItsController`.

Attributo	Tipo	Default	Descrizione
`Type`	enum.	-	Tipologia di nodo che si intende rappresentare (RSU, veicolo medium, veicolo full).
`DirectionTolerance`	double	8.0	Angolo di tolleranza per il cono di conformità di direzione per questo nodo.

insieme dei sorgenti. Nel seguito verranno elencati per ciascun componente i diversi attributi che consentono di modificarne il comportamento.

Per installare il controllore ITS in un oggetto `Node` appena creato è possibile utilizzare il metodo di ns-3 `CreateObjectWithAttributes` che consente di creare e ottenere un riferimento a un nuovo oggetto di tipo `ItsController` e di specificare un valore per uno o più attributi previsti per quella classe di oggetti. Tra gli attributi disponibili per il controllore (vedi tabella) è fondamentale specificare un valore per l'attributo `Type` che specifica su quale tipo di entità (RSU o veicolo, vedi di seguito) si sta installando il componente appena creato. Per completare la procedura, una volta instanziato e configurato il componente `ItsController` è necessario aggregarlo a un oggetto `Node` usando la primitiva di ns-3 `AggregateObject`; in seguito a tale operazione il controllore rileverà l'avvenuta installazione e imposterà adeguatamente i componenti base previsti per l'architettura.

Terminata la procedura di installazione, il controllore risulterà funzionante ma privo di qualsiasi tipo di logica. Per inserire un comportamento all'interno di un nodo provvisto di controllore ITS si utilizza la primitiva `SubscribeBehaviour` e si fornisce come parametro un oggetto della classe `Behaviour` appena creato.

Attivazione e spegnimento dei nodi

Un nodo ITS è detto *attivo* quando è in grado di comunicare e ricevere messaggi da altri nodi attivi presenti nel proprio raggio di copertura. Nell'intervallo di tempo compreso tra l'inizio e la fine della simulazione, un nodo

segue il percorso definito dalla traccia di mobilità in uso, indipendentemente dal fatto di essere attivo o spento. Attraverso due semplici primitive (rispettivamente `Activate` e `Deactivate`) è possibile quindi avviare o disinnescare un veicolo in qualsiasi momento all'interno del tempo simulato. Il metodo `IsActive` permette inoltre di sapere se il nodo sottostante è attualmente attivo.

Viene adottata una convenzione secondo la quale i nodi che corrispondono a unità infrastrutturali vengono automaticamente attivati dal controllore al momento della loro creazione. Al contrario, i veicoli entrano ed escono dal sistema in istanti definiti all'interno di una traccia di mobilità e questi in generale non corrispondono necessariamente al tempo di inizio e di fine dell'intera simulazione. Siccome ns-3 non supporta la creazione e la distruzione di nodi a simulazione avviata, si utilizza la primitiva `Activate` per avviare le singole autovetture nell'istante in cui entrano nell'ambiente. Per convenzione quindi, un nodo veicolo – dal momento in cui viene istanziato – viene mantenuto disattivato fino all'intervento dell'utente.

Classi di nodi

Nel paragrafo 2.3 sono state presentate le tre classi di veicoli che sono state previste per rappresentare veicoli dotati di strumenti di comunicazione e sensori all'avanguardia (full vehicle), utenti che utilizzano dispositivi handheld con prestazioni più contenute (medium vehicle) e autovetture tradizionali sprovviste di ogni sistema di comunicazione (shadow vehicle). È stato inoltre introdotto l'oggetto ns-3 `Node` che aggrega tutti i molteplici componenti che ne formalizzano il comportamento e le proprietà nel corso della simulazione. Si ricordi tuttavia che tale oggetto viene utilizzato per rappresentare qualsiasi entità che partecipa ai protocolli sviluppati, sia essa un'autovettura in movimento o una RSU fissa posizionata in un incrocio stradale. Si presenta pertanto l'esigenza di garantire alle logiche di protocollo la capacità di poter distinguere la *tipologia* di un generico oggetto `Node` e comprendere quindi se esso è rappresentativo di una RSU o di un veicolo.

Il metodo `GetNodeType` offerto dal controllore restituisce un intero enu-

merato che descrive la tipologia di nodo sul quale esso è stato installato. I valori previsti per tale enumerato sono:

`NT_RSU` indica un dispositivo fisso di infrastruttura. Come mostrato nella Figura 3.4 questa tipologia è caratterizzata da un'organizzazione dei componenti ridotta e semplificata rispetto alla controparte mobile pur disponendo della stessa interfaccia a livello di controllore;

`NT_VEHICLE` indica un dispositivo installato all'interno di un'autovettura. Per distinguere tra le diverse classi di veicoli previste, questo valore viene ulteriormente suddiviso come segue:

`NT_FULL` rappresenta la classe di veicoli full;
`NT_MEDIUM` rappresenta la classe di veicoli medium;

`NT_ALL` racchiude tutti i valori precedenti e indica pertanto una qualsiasi entità in grado di interagire con i protocolli ITS realizzati.

I veicoli shadow sono caratterizzati dal fatto di non disporre di alcun componente che consenta loro di comunicare. Per questo motivo essi risultano del tutto invisibili agli altri nodi e non è quindi necessario includere un valore per poterli distinguere.

Invio di messaggi

Le logiche di protocollo installate sopra un controllore fanno affidamento alle primitive di invio e ricezione dei messaggi per realizzare lo scambio di informazioni da un nodo verso uno o più pari. La modalità più versatile e maggiormente utilizzata per produrre e spedire un messaggio consiste nell'utilizzare il metodo `Send` messo a disposizione dal controllore. Tramite esso è possibile specificare una tipologia di nodi destinatari (si selezioni `NT_ALL` per non imporre alcun filtro), il contenuto informativo del pacchetto e il tipo di protocollo previsto per la sua elaborazione. Il metodo `SendTo` permette invece di specificare un preciso nodo come unico destinatario del messaggio (unicast). È prevista un'ulteriore primitiva `SendLoopback` la quale non invia un pacchetto verso la rete esterna ma lo inoltra verso altre logiche all'interno dello stesso nodo per supportarne la comunicazione diretta ove necessario.

Vista l'importanza che questo argomento riveste all'interno del sistema, le modalità con cui i messaggi vengono indirizzati, trasferiti e processati dalle singole logiche di protocollo verranno approfondite nel paragrafo seguente.

3.2.2 Scambio di messaggi

Le primitive più importanti messe a disposizione dal controllore sono senza dubbio quelle per l'invio e la ricezione dei messaggi. Per comprendere appieno il loro funzionamento è necessario tuttavia chiarire le modalità con cui ogni singolo pacchetto di informazioni viene trasferito dalla particolare logica che lo ha generato (mittente) al componente adeguato per la sua ricezione (destinatario). Siccome è preferibile che i diversi comportamenti installati sopra a un controllore restino il più possibile indipendenti tra loro, è necessario predisporre un meccanismo che consenta a ciascun mittente di specificare per ogni messaggio in uscita una tipologia di protocollo a cui si riferisce, senza che sia necessario conoscere a priori lo specifico componente di comportamento designato per riceverlo. In questo modo si evita non solo di introdurre una dipendenza diretta tra i diversi behaviour ma si garantisce anche una maggiore flessibilità ed estensibilità dell'intera architettura dal momento in cui è possibile avere più logiche locali a uno stesso nodo come destinatari indipendenti di un unico messaggio.

Il sistema definisce in maniera univoca i quattro diversi protocolli previsti per il sistema sviluppato attraverso il tipo di dato enumerato `ProtocolId` (PID) che può assumere i valori riassunti in Tabella 3.2:

A ciascun messaggio inviato per mezzo del controllore viene quindi assegnato uno dei precedenti valori per indicare a che tipo di protocollo il messaggio appartiene. Al momento della ricezione di un messaggio, un qualsiasi nodo ricevente analizzerà il contenuto dell'header del pacchetto, estrarrà tale valore e lo confronterà con una tabella nella quale viene indicata per ogni behaviour iscritto al controllore una lista di PID al quale tale logica è interessata. Il controllore si preoccupa quindi di consegnare una copia[2] di un

[2]Si sceglie di inviare una copia del pacchetto in ingresso piuttosto che un suo riferimento per scongiurare side effects tra i componenti destinatari.

Tabella 3.2: Valori dell'enumerato `ProtocolId`.

Valore	Descrizione
`PID_GROUP_SETUP`	Protocollo per la formazione dei gruppi ed elezione del nodo leader
`PID_GROUP_MANAGEMENT`	Protocollo per la gestione del gruppo all'interno del suo ciclo di vita
`PID_HEARTBEAT`	Messaggi CAM tra veicoli per la creazione di una mappa di nodi dinamica durante l'uso di protocolli proattivi
`PID_MARKER`	Protocollo di gestione per nodi marker

pacchetto appena ricevuto a quei comportamenti che hanno espressamente richiesto di ricevere informazioni relative a quello specifico PID. A tale scopo la classe base astratta `Behaviour` prevede il metodo virtuale `Receive` attraverso il quale il controllore comunica l'avvenuta ricezione di un messaggio d'interesse. La procedura di *dispatch* dei pacchetti in ingresso non è tenuta a seguire un ordine preciso nella consegna ai diversi behaviour in quanto indipendenti.

Questo meccanismo viene schematizzato nella Figura 3.5 dove vengono raffigurati un nodo mittente α e uno ricevente β. Il behaviour `A` all'interno del primo nodo chiede al suo controllore l'invio di un pacchetto destinato a qualsiasi tipo di entità (`NT_ALL`) relativamente al protocollo `pid1`. Il controllore di α processa tale richiesta e invia il pacchetto sfruttando l'interfaccia di rete installata per quel nodo dopo aver arricchito il contenuto del messaggio con un ulteriore header che specifica alcune informazioni utili sul mittente (vedi nel seguito). In seguito alla ricezione del pacchetto, il controllore di β esamina la lista delle sottoscrizioni dei comportamenti in suo possesso e scopre che entrambi i behaviour `B` e `C` sono interessati al protocollo `pid1` e quindi inoltra tale messaggio ad entrambi.

È importante sottolineare che esiste una relazione uno a molti tra un comportamento mittente e relativi destinatari e tra un nodo mittente e riceventi: un pacchetto viene inviato da uno e un solo nodo ad opera di un solo specifico comportamento; un pacchetto viene ricevuto da zero, uno o più nodi e – localmente a ciascuno di questi – può essere processato da zero, uno

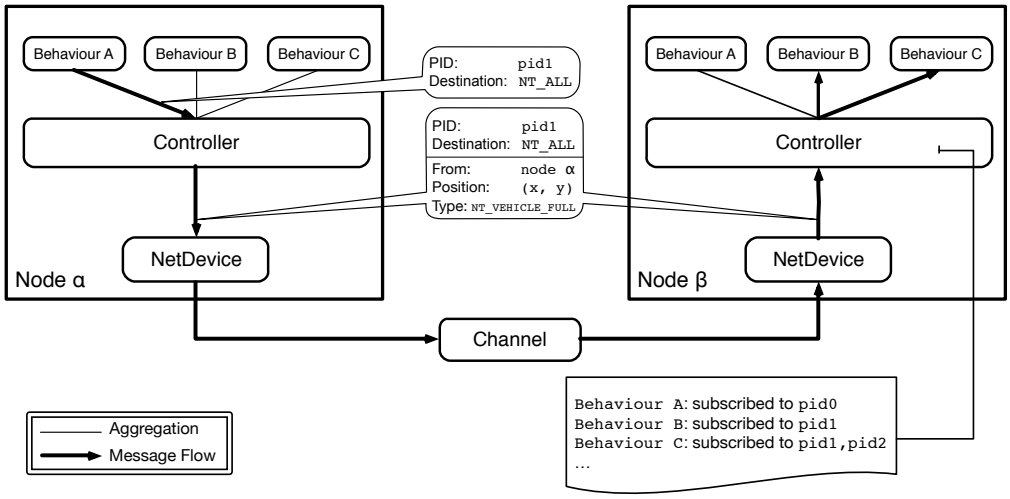

Figura 3.5: Invio e instradamento di un messaggio ITS da un nodo α a un nodo β.

o più comportamenti diversi. Questo meccanismo consente un forte disaccoppiamento tra mittente e destinatario e permette di organizzare protocolli molto articolati (in particolare quello di gestione dei gruppi) in più logiche indipendenti che svolgono un insieme ristretto di funzionalità complementari in sinergia con le altre.

Header

Un pacchetto di dati trasferito in una rete di nodi connessi viene codificato in ns-3 dalla classe `Packet` la quale realizza di fatto un wrapper di un buffer di byte che descrive il contenuto *raw* di un pacchetto. Per evitare di dover compilare e interpretare ogni messaggio servendosi unicamente di una mappa di byte, ns-3 mette a disposizione dei meccanismi di più alto livello per organizzare e recuperare informazioni all'interno dei singoli pacchetti. In particolare si è fatto un uso estensivo delle funzionalità offerte dalla classe astratta `Header` che permette di leggere e scrivere un header per un pacchetto sfruttando la semplice astrazione di oggetto tipica dei linguaggi object-oriented. Per definire un formato di header personalizzato è possibile estendere la classe `Header` in base alle proprie esigente.

Un'ulteriore comodità è data dalla possibilità di poter aggiungere più

oggetti header all'interno di un singolo pacchetto ns-3, potendo così simulare quanto avviene normalmente in uno stack protocollare, dove ciascun layer di servizio aggiunge il proprio header per descrivere il payload incapsulato. Nel sistema sviluppato si è scelto di sfruttare questa funzionalità per organizzare le informazioni di un pacchetto ITS nei due principali livelli di architettura: quello delle logiche di protocollo e quello di controllore.

A livello più alto, ogni behaviour arricchisce ciascun messaggio con una serie di informazioni relative allo specifico protocollo per cui è stato concepito. Tali dati vengono passati al controllore mediante le primitive di scambio messaggi ma risultano del tutto opachi a quest'ultimo, non essendo esso in grado di interpretarli. Per ciascuno dei protocolli descritti nella Tabella 3.2 viene realizzata un'apposita estensione della classe `Header` che verrà descritta in sede di discussione dei protocolli stessi.

A livello di controllore, ad ogni messaggio in uscita viene automaticamente aggiunto l'header `ItsHeader` che fornisce un'importante lista di informazioni relative al mittente e al suo stato, unita ad alcune specifiche sul tipo di destinatario desiderato; questi dati sono di norma estremamente utili anche alle singole logiche di protocollo e restano pertanto visibili ai livelli più alti del sistema, contrariamente a quanto avviene nel modello ISO/OSI nel quale è previsto un completo isolamento tra i layer verticali. I campi previsti per l'header ITS a livello di controllore vengono riassunti nella Tabella 3.3:

La lista delle informazioni incluse per il mittente include un valore intero identificativo univoco per il nodo sorgente, il suo `NodeType` e la sua posizione e direzione attuali. L'id univoco per ciascun nodo (`NodeId`) viene ottenuto riutilizzando il concetto di identificativo intero previsto in ns-3 e assegnato automaticamente ad ogni istanza della classe `Node`; al di là di tale semplificazione, tale valore è del tutto analogo a quello di indirizzo MAC per le interfacce di rete. Le informazioni sulla posizione e direzione del mittente – nel caso di nodi corrispondenti a veicoli mobili – riflettono il campionamento più recente a disposizione di quest'ultimo e sono pertanto soggette ad errori e *staleness*.

Le informazioni circa il tipo di sorgente e destinatario permettono sia di identificare la classe di nodo che ha inviato un messaggio, sia di effettuare un

Tabella 3.3: Campi previsti per `ItsHeader`.

Campo	Tipo di dato	Descrizione
`ProtocolId`	enumerato	Identificativo del protocollo relativo al messaggio (Tabella 3.2)
`SourceId`	intero	Identificativo univoco del nodo mittente
`SourceType`	enumerato	Identificativo del tipo di nodo mittente (Paragrafo 3.2.1)
`SourcePosition`	`Vector2D`	Posizione del nodo mittente al momento dell'invio del messaggio
`SourceDirection`	virgola mobile	Direzione del nodo mittente al momento dell'invio del messaggio
`DestinationId`	intero	Identificativo univoco del nodo destinatario (facoltativo)
`DestinationType`	enumerato	Identificativo del tipo di nodi destinatari (facoltativo)

multicast in base al valore specificato in `DestinationType`. Sfruttando questo campo è quindi possibile destinare un pacchetto ai soli veicoli escludendo le RSU, oppure confinarlo ai soli veicoli di tipo full presenti nelle vicinanze e così via. È importante comunque notare che le trame di dati trasmesse all'interno di un canale fisico vengono sempre ricevute da tutti i nodi ns-3 in grado di riceverle, indipendentemente dai valori specificati all'interno dell'header. Le funzionalità di multicast e filtraggio di messaggi descritte di seguito vengono quindi realizzate esclusivamente a livello di controllore.

Il campo `DestinationId` consente infine di realizzare una trasmissione *unicast* verso uno specifico nodo. Messaggi di questo tipo vengono inviati mediante la primitiva `SendTo` ed è possibile specificare il valore chiave `ID_ALL` per indicare che il messaggio non è esclusivo di un particolare nodo e può essere inviato come broadcast o multicast. In nessun caso vi è ovviamente alcuna garanzia di avvenuta ricezione da parte del nodo o della classe di nodi indicati come mittenti.

Filtraggio dei messaggi

L'header appena descritto viene incluso autonomamente dal controllore mittente per ogni messaggio scambiato e processato dal ricevente al momen-

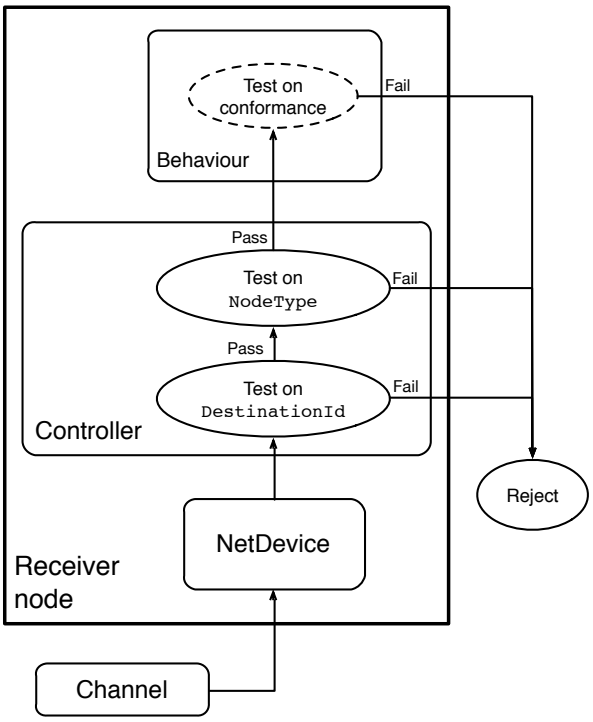

Figura 3.6: Test eseguiti su un pacchetto in ricezione. I due test effettuati dal controllore verificano se è stato specificato un mittente (`DestinationId`) o una classe di nodo(`NodeType`) come filtri di destinazione. Un singolo behaviour può effettuare un eventuale test di conformità sulla direzione.

to della ricezione. Le informazioni in esso contenute vengono consegnate anche alle logiche superiori, in particolare per poterle informare sull'identità e le caratteristiche del mittente, utili allo svolgimento dei diversi protocolli realizzati.

Prima di consegnare un messaggio agli opportuni Behaviour registrati, il controllore svolge una serie di test per verificare se il pacchetto appena ricevuto è effettivamente destinato a quel nodo. Questi test consistono in un controllo sul `NodeId` destinatario – se specificato dalla sorgente – e in un filtro relativo al `NodeType`. Se entrambi i test vengono superati il messaggio è considerato valido ed è inoltrato alle opportune logiche, in caso contrario esso viene scartato. Questi passaggi vengono riassunti nella Figura 3.6.

Un'ulteriore tipologia di filtraggio può essere effettuata sulla conformità dei valori di direzione del nodo mittente e destinatario. Questo test può esse-

re richiesto da una logica di protocollo, la quale decide autonomamente come comportarsi in caso di successo o fallimento del check. Il controllore mette a disposizione il metodo `IsConformantDirection` che accetta in ingresso un valore di direzione sul quale effettuare il controllo (in questo caso viene passato il valore `SourceDirection` letto dall'header) e restituisce un booleano contenente l'esito del test. Come è stato definito nel paragrafo 2.2.2, due direzioni sono conformanti solo se l'angolo compreso tra esse non supera un valore di tolleranza massima. Per ciascun nodo, la tolleranza alla conformità è data dall'attributo `DirectionTolerance` del modulo controllore. Sfruttando questo meccanismo è quindi possibile effettuare una nuova tipologia di messaggi multicast i cui destinatari risultano essere soltanto quei nodi che condividono la stessa direzione di marcia del nodo mittente.

3.3 Modulo di campionamento

Quasi tutte le logiche di protocollo implementate per il sistema utilizzano in maniera estensiva le informazioni di posizione, velocità e direzione introdotte nel capitolo 2 e prodotte al livello più basso dell'architettura dal componente `NodeSampler`. Compito di quest'ultimo è quello di effettuare una serie di campionamenti ad intervalli regolari nel tempo, introdurre degli errori casuali per simulare un qualsiasi livello di imprecisione negli strumenti di rilevazione e di effettuare una media dei campionamenti più recenti per restituire valori meno sensibili ai disturbi e ai cambiamenti di direzione poco marcati.

Al momento del singolo campionamento, i valori ottenuti dal modulo di `MobilityModel` di ns-3 consistono nei vettori di posizione e velocità. La posizione corrente del nodo viene alterata sommandovi un vettore di errore caratterizzato da una direzione con distribuzione uniforme nell'intero angolo giro e da un modulo con distribuzione normale (gaussiana) a valore medio pari a zero e un raggio massimo configurabile. L'errore sulla velocità viene invece ricavato ruotando semplicemente il vettore di velocità di un angolo distribuito normalmente e con valore medio pari a zero; un valore di varianza molto basso garantisce che gli *outcome* della distribuzione si mantengano suf-

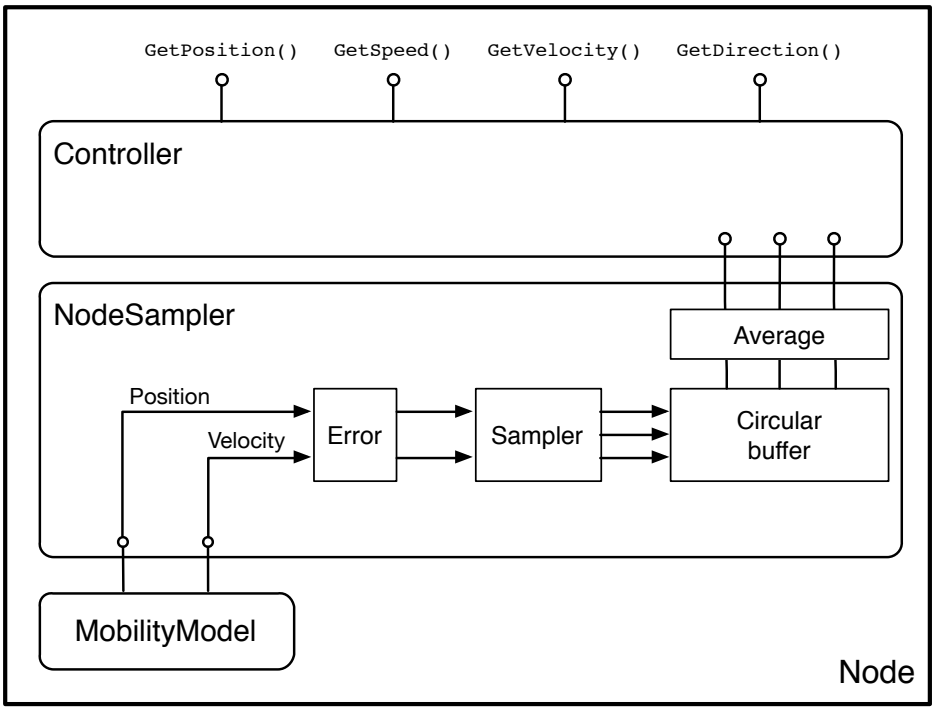

Figura 3.7: Schema di funzionamento del componente `NodeSampler`: i valori di posizione e velocità ottenuti da `MobilityModel` vengono perturbati con degli errori, viene calcolata la direzione del nodo e memorizzata all'interno dei buffer. Una media dei valori presenti in un dato momento nel buffer fornisce al controllore lo stato relativo a posizione, velocità (modulo e vettore) e direzione correnti.

ficientemente prossimi allo zero. Si è scelto di mantenere inalterato il modulo della velocità in quanto lo studio degli effetti di tale perturbazione è di scarso interesse, considerato che questa grandezza non influenza in alcun modo le logiche degli algoritmi realizzati e l'errore verrebbe quindi semplicemente trasferito alle RSU. In letteratura esistono dei modelli di errore estremamente più sofisticati [Ran94] che consentono di simulare in maniera molto più realistica le imprecisioni dovute ai sensori di geolocalizzazione più utilizzati. L'adozione di tali tecniche avrebbe tuttavia introdotto una complessità eccessiva e i risultati forniti dal semplice modello appena descritto sono apparsi adeguati per gli scopi del presente lavoro di tesi.

Terminata l'introduzione di errori, i valori di posizione e velocità vengono ulteriormente processati prima della loro memorizzazione. In questa fase –

attraverso semplici calcoli trigonometrici – si procede all'estrazione dell'informazione relativa alla direzione a partire dal vettore di velocità appena ottenuto. Viene inoltre verificato che lo spostamento compiuto dal nodo tra il campionamento attuale e quello precedente sia superiore in modulo a una soglia minima prefissata. Nel caso di nodo fermo o con velocità prossima allo zero si avrebbero infatti dei campionamenti molto vicini nello spazio e il loro contributo risulterebbe poco utile. Un vettore di velocità nullo non consentirebbe inoltre di calcolare la direzione corrente del nodo. Questo accorgimento consente quindi di mantenere una distanza spaziale minima tra campionamenti successivi. Come conseguenza, negli intervalli di tempo in cui il veicolo si mantiene fermo, le informazioni relative a posizione, direzione e velocità non vengono aggiornate ed esso mantiene lo stato più recente da lui memorizzato.

Terminati tutti i calcoli e le verifiche sul campione corrente, questo viene memorizzato all'interno di un buffer circolare di dimensione configurabile. Quando il modulo è a regime, l'inserimento di un valore recente comporta lo scarto di quello più vecchio. Le informazioni messe a disposizione da `NodeSampler` ai livelli superiori consistono in una media aritmetica[3] di tutti i valori precedentemente memorizzati. La dimensione del buffer determina quindi in maniera indiretta quanto il modulo risulta sensibile ai cambiamenti di stato sulla guida del veicolo e – soprattutto – con quale ritardo essi vengono avvertiti ai livelli più alti dell'architettura. Una memoria di grandi dimensioni si comporta inoltre come una sorta di filtro sui disturbi di breve durata in quanto questi vengono successivamente mediati tra più campioni.

Il percorso che le informazioni di posizione e velocità seguono all'interno del componente `NodeSampler` e le operazioni ad esse applicate vengono riassunte e illustrate nella Figura 3.7 dove viene anche riportata l'interfaccia offerta dal controllore alle logiche di protocollo. Occorre ricordare che i nodi di tipo `NT_RSU` non sono dotati di mobilità e non prevedono pertanto il modulo per il campionamento di tali valori. In tali casi il controllore è consapevole della diversa organizzazione adottata per il nodo e provvede personalmente a fornire dei valori nulli per velocità e direzione e a leggere

[3]Le grandezze angolari vengono ovviamente calcolate come medie circolari.

Tabella 3.4: Attributi previsti per il modulo `NodeSampler`.

Attributo	Tipo	Default	Descrizione
`Resolution`	intero	1000	Intervallo di tempo (ms) tra due campionamenti successivi
`Quantity`	intero	5	Numero di campioni memorizzabili all'interno del buffer circolare
`PositionRadius`	double	20	Raggio di errore massimo (m) per il campionamento della posizione
`PositionVariance`	double	1.8	Valore della varianza per la distribuzione degli errori di posizione
`DirectionVariance`	double	0.002	Valore della varianza per la distribuzione degli errori di direzione
`MovementThreshold`	double	2	Distanza minima (m) tra due campionamenti successivi

direttamente dall'oggetto `MobilityModel` la sua posizione, senza aggiungervi alcun errore.

Il comportamento del modulo `NodeSampler` è progettato per offrire la massima flessibilità sui diversi aspetti delle operazioni di campionamento sopra descritte. Sia la generazione degli errori sia il comportamento del buffer circolare sono interamente parametrizzabili attraverso i molteplici attributi predisposti dalla classe ed elencati nella Tabella 3.4.

Avendo presentato l'architettura complessiva del sistema realizzato, i suoi principali componenti e le meccaniche adottate per la comunicazione tra i suoi partecipanti è ora possibile procedere alla descrizione dettagliata delle logiche poste al livello superiore: i protocolli che realizzano concretamente il comportamento atteso del sistema.

Capitolo 4

Design e implementazione dei protocolli ITS

Nel precedente capitolo è stata presentata l'architettura generale del sistema sviluppato e ci si è soffermato sui suoi componenti principali. Scopo di tali moduli è quello di fornire servizi essenziali alle logiche di protocollo poste immediatamente sopra ad essi, fornendo loro informazioni di stato sul veicolo e primitive essenziali per lo scambio dei messaggi e la gestione del nodo nel quale si trovano ad operare. Verranno in questo capitolo esaminati i vari *comportamenti* sviluppati per il sistema, ovvero quelle logiche idealmente indipendenti tra loro introdotte nel paragrafo 3.2 le quali realizzano concretamente quei protocolli di ITS che sono stati presentati nel corso del capitolo 2 e che costituiscono pertanto il "core business" della soluzione presentata in questo testo.

4.1 Protocolli di formazione dei gruppi

I protocolli che sono stati realizzati per primi sono quelli responsabili della formazione dei gruppi a partire da singoli nodi in stato di quiescenza (idle). Per formazione di un gruppo (*group formation*) si intende quell'attività finalizzata alla creazione di un nuovo plotone ad opera dei veicoli che transitano in una precisa direzione, individuando quali nodi risultano compatibili con

la definizione di gruppo data al paragrafo 2.2 e quale tra loro viene eletto al ruolo di leader. Superata questa fase preliminare, il plotone inizia la raccolta dei dati di interesse sulla composizione del traffico e prosegue nel suo ciclo di vita seguendo le logiche illustrate nel paragrafo 4.3.

In generale, i nodi che partecipano alla formazione di un gruppo non sono isolati e nel corso delle loro comunicazioni possono ricevere messaggi provenienti da nodi già membri di altri gruppi oppure da veicoli che stanno intraprendendo la formazione di un diverso plotone. Nel primo caso, la completa separazione tra i messaggi è garantita dal campo `ProtocolId` presente nell'header principale e indicatore del protocollo a cui il pacchetto si riferisce. Dalla Tabella 3.2 si evince che la formazione dei gruppi e la successiva gestione del loro ciclo di vita si appoggiano su protocolli diversi e pertanto qualsiasi interferenza tra i relativi messaggi viene scongiurata senza sforzi aggiuntivi. È tuttavia necessario prevedere un sistema in grado di evitare sovrapposizioni tra due o più insiemi di veicoli impegnati in uno stesso intervallo di tempo nella formazione di gruppi separati, visto che tali nodi comunicano scambiandosi messaggi relativi allo stesso protocollo (PID_GROUP_SETUP). Tale meccanismo è inoltre estremamente utile per identificare i diversi gruppi attivi nel corso della simulazione, permette alle RSU di distinguere e organizzare i diversi contributi ottenuti da essi e si pone come base per qualsiasi estensione al sistema che intenda ricorrere alla comunicazione tra gruppi adiacenti per realizzare uno scambio di dati tra nodi distanti, come suggerito al paragrafo 6.3.

Nel rispondere a tale esigenza occorre notare che ogni gruppo è presieduto da un unico leader ed è creato per mezzo e in funzione di una specifica unità infrastrutturale alla quale vengono forniti i dati raccolti e integrati dai vari membri. Sembra pertanto ragionevole predisporre come identificatore univoco di un gruppo un numero intero che individui sia l'RSU a cui esso fa riferimento sia il nodo eletto come leader. Tale intero viene denominato `GroupId` (nel seguito abbreviato in GID) ed è composto dalla concatenazione dei due `NodeId` relativi rispettivamente alla RSU di riferimento e al nodo responsabile del plotone. Questa scelta risolve inoltre a priori qualsiasi problema di duplicazione dei nodi leader visto che, per come è stato definito

l'identificatore, un nodo può essere gestore di un solo gruppo e uno stesso gruppo non può essere guidato da più di un nodo.

Durante la procedura di formazione dei gruppi, quando cioè non sono ancora stati individuati dei gestori definitivi per i nodi appena attivati, i protocolli di group formation non possono ovviamente utilizzare un GID per coordinare la loro attività. Questo tuttavia non costituisce un problema in quanto è sufficiente utilizzare le sole informazioni relative alla direzione di riferimento e alla precisa RSU che ha sollevato la richiesta di setup. Tutti i messaggi usati dalle logiche di formazione includono tale coppia di valori all'interno dei loro header. Un'unico messaggio beacon emesso dalla RSU può quindi dare luogo alla creazione di più gruppi relativi a una stessa direzione: i protocolli sono infatti progettati per indurre i veicoli a partizionarsi tra loro nell'eventualità che la loro disposizione complessiva risulti impossibile da contenere all'interno di un unico plotone. Al termine della procedura, ogni partizione assumerà l'identità di gruppo e disporrà di un GID autonomo.

L'idea principale dalla quale le logiche di group formation realizzate traggono ispirazione si basa su due semplici princìpi. Il primo sancisce che per evitare un eccessivo consumo di risorse energetiche e computazionali, specialmente nel caso di veicoli che sfruttano dispositivi con autonomia limitata, i nodi dovrebbero essere mantenuti attivi (cioè in comunicazione) soltanto negli istanti di tempo in cui si trovano sufficientemente vicini alle unità infrastrutturali; questa osservazione è anche motivata dal fatto che i dati raccolti in posizioni eccessivamente distanti dai punti notevoli risulterebbero con ogni probabilità di scarso interesse.

Il secondo principio aiuta nell'identificazione del nodo leader per un gruppo e stabilisce che il veicolo ideale per tale ruolo è quello che riesce a comunicare con il maggior numero di suoi pari, limitatamente al suo range operativo e ai nodi che risultano possedere una direzione conformante a quella prevista dalla RSU. Adottando questa strategia, il ruolo di leader viene "spinto" verso il centro del gruppo stesso e permette di ridurre il numero di plotoni che devono essere formati per "coprire" i nodi presenti in una data direzione.

Nel paragrafo 2.2.3 sono state anticipate due diverse modalità messe a punto per la formazione dei gruppi: una reattiva e una proattiva. La pri-

ma segue fedelmente i due princìpi appena esposti ed opera ipotizzando una conoscenza nulla sull'ambiente circostante da parte dei nodi in stato idle che vengono attivati a seguito della ricezione di un beacon da una RSU. La seconda ammette una semplice forma di comunicazione tra nodi in stato idle basata su semplici messaggi heartbeat[1] tra nodi adiacenti finalizzata alla creazione di una mappa di nodi distribuita per facilitare e accelerare la procedura di elezione del leader. Ciascuno di questi due protocolli verrà esposto nei successivi paragrafi, mentre nel capitolo 5 verranno spesso commentati e messi a confronto i risultati ottenuti da entrambe le strategie.

Il primo passo per la formazione di un gruppo consiste quindi nell'attivare i nodi che si stanno avvicinando a una qualsiasi RSU in base alla direzione che questi stanno seguendo. Ciò viene realizzato dalle RSU stesse mediante un costante invio di segnali *beacon* ad intervalli regolari ed equidistanziati. Ricordando che ogni RSU è solitamente incaricata di monitorare un insieme di più strade che si incontrano in un punto, sarà necessario che ciascuno di questi messaggi contenga una lista dei valori di direzione sulla base dei quali avviare la formazione dei gruppi. Ogni nodo in stato idle che riceve un beacon verifica se la sua direzione corrente è conforme a uno dei valori in esso indicati; se nessuno dei valori risulta compatibile il messaggio è semplicemente ignorato, in caso contrario il nodo si attiva e dà inizio alla successiva fase di elezione del gruppo, come mostrato in Figura 4.1.

Tabella 4.1: Formato dei messaggi di Group Setup Request inviati dalle RSU.

Campo	Valore	
`ProtocolId`	`PID_GROUP_SETUP`	
`MessageType`	`MT_GROUP_SETUP_REQ`	
Campo	Tipo	Descrizione
`RSUId`	`NodeId`	Identificatore della RSU che richiede la formazione del gruppo.
`Directions`	lista di double	Lista dei valori di direzione alle quali la RSU è interessata.

[1] Vedi il paragrafo 4.2.

Riassumendo, l'*attivazione diretta* di un nodo si verifica in seguito alla ricezione di un beacon da una RSU contenente una lista delle direzioni d'interesse e soltanto dopo che una tra queste è stata verificata come conformante alla direzione corrente di quel nodo. Si ha invece una *attivazione indiretta* quando un nodo in stato idle viene risvegliato da un veicolo già attivo a lui conforme. Entrambe le strategie prevedono modalità differenti di attivazione indiretta che verranno approfondite di seguito. Il modulo che realizza una logica di tipo reattivo è denominato `BehaviourGroupSetupReactive`, mentre quello di tipo proattivo viene definito da oggetti di tipo `BehaviourGroupSetupProactive`; entrambi derivano dalla classe `Behaviour` che formalizza un'interfaccia comune a tutti i moduli di protocollo e realizza alcuni servizi utili ad essi.

4.1.1 Strategia reattiva

Il secondo dei due princìpi adottati per la formazione dei gruppi stabilisce che nella selezione di un nodo leader per un gruppo si devono preferire quelli che racchiudono il più alto numero di nodi all'interno del loro raggio di comunicazione. Tuttavia, nel momento in cui questi vengono attivati a seguito della ricezione di un beacon proveniente da una RSU, non è disponibile alcuna conoscenza di quali e quanti nodi risultano presenti nel range di un nodo, né tantomeno esiste la possibilità di sapere quali tra questi possiedono una direzione conforme a quelle indicate dalla RSU.

Per sopperire a tale mancanza si prevede che nel momento in cui un qualsiasi nodo viene attivato in maniera diretta, esso provveda ad inviare un messaggio broadcast contenente la propria identità e la propria intenzione a formare un gruppo nella direzione specificata. Per un certo periodo di tempo tale nodo rimane in ascolto di eventuali altre risposte relative alla stessa combinazione di RSU e direzione: in caso di ricezione egli è di fatto informato della presenza di un altro nodo nel suo raggio operativo che intende partecipare alla formazione di un gruppo verso lo stesso senso di marcia. Tenendo traccia del numero di risposte pervenute mediante un semplice contatore si ottiene una stima dei veicoli che tale nodo riuscirebbe a controllare se venisse

eletto al ruolo di leader.

Terminato il periodo di tempo dedicato all'ascolto delle risposte, i nodi attivi si contendono tale qualifica mediante una sorta di "asta" nella quale un nodo può proporsi come leader inviando un messaggio broadcast di offerta (bid) e specificando in esso il numero di veicoli percepiti. Il nodo che avrà ottenuto il valore più alto (o uno qualsiasi tra questi in caso di parità) concluderà definitivamente il protocollo di elezione inviando un messaggio che certifica la sua elezione al ruolo di coordinatore del gruppo.

L'intero protocollo di group formation può quindi essere suddiviso in due fasi: in una prima fase i nodi vengono attivati e raccolgono informazioni su quali e quanti nodi conformanti sono presenti nelle vicinanze, nella seconda si procede a stabilire quale nodo è in grado di gestire il maggior numero di veicoli tra quelli appena attivati. Il vincitore viene eletto come leader e conclude il protocollo di formazione, avviando di conseguenza quelli relativi alla gestione del ciclo di vita mostrati nel paragrafo 4.3.

Fase 1: attivazione dei nodi

Nell'esporre la prima delle due fasi sopra introdotte si farà riferimento alla Figura 4.1 dove vengono raffigurati alcuni nodi inizialmente inattivi che si avvicinano a una RSU seguendo una delle tre direzioni di interesse. Nella prima parte della figura, la RSU invia un messaggio beacon che viene ricevuto dai nodi `a` e `f`. Quest'ultimo ha una direzione che non è conforme a nessuna tra quelle specificate nella lista e pertanto ignora il messaggio. Il nodo `a` è invece conforme alla direzione -160° e viene quindi attivato in maniera diretta. Nella Figura 4.1b, il nodo reagisce all'attivazione mediante l'invio di un messaggio di risposta (MT_GROUP_SETUP_RESPONSE) che testimonia la propria disponibilità a formare un gruppo per la combinazione RSU e direzione in esso specificata. Il formato dei messaggi di risposta viene definito nella Tabella 4.2.

Alla ricezione di un messaggio di risposta, un nodo inattivo può attivarsi in maniera indiretta a patto che la propria direzione sia conforme a quella indicata nel campo `Direction`. Se il test di conformità ha successo e il nodo

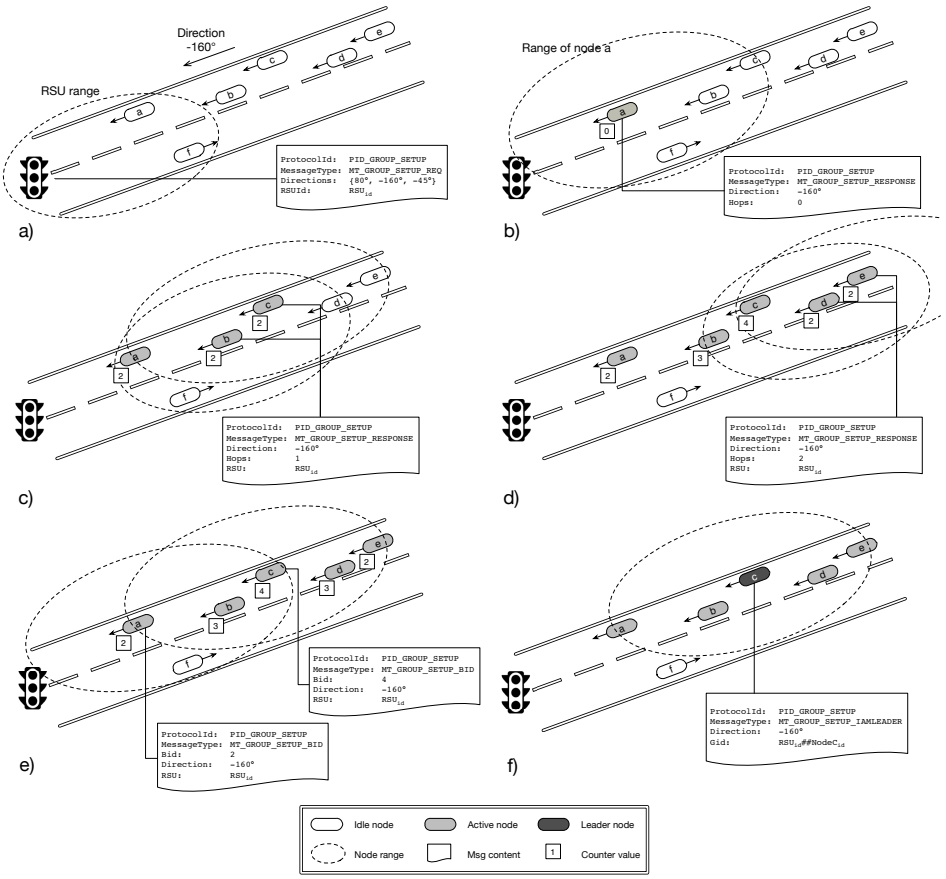

Figura 4.1: Schematizzazione del processo di formazione dei gruppi con strategia reattiva. L'RSU invia dei beacon periodici che attivano i nodi nelle direzioni specificate (a). Ogni nodo attivato invia un messaggio di risposta per informare i vicini della propria partecipazione al protocollo (b). Eventuali nodi inattivi che percepiscono tale messaggio vengono attivati indirettamente e rispondono a loro volta purché non venga raggiunto il limite massimo di due hop (c, d). Segue una fase di contesa del ruolo di leader mediante asta sul numero dei nodi percepiti (e). Il nodo che ha inviato l'offerta (bid) più elevata viene eletto come leader (f).

Tabella 4.2: Formato dei messaggi di Group Setup Response inviati dai nodi a seguito della loro attivazione.

Campo	Valore	
`ProtocolId`	`PID_GROUP_SETUP`	
`MessageType`	`MT_GROUP_SETUP_RESPONSE`	
Campo	Tipo	Descrizione
`RSUId`	`NodeId`	Identificatore della RSU che richiede la formazione del gruppo.
`Direction`	double	Direzione per la quale si è scelto di formare un gruppo.
`Hops`	intero	Numero di salti di attivazione compiuti a partire dalla richiesta originale.

viene attivato si dice che la richiesta di formazione del gruppo ha subito un salto di attivazione (hop), ovvero tale veicolo è stato attivato non da un beacon proveniente dalla RSU ma da un messaggio di risposta di un suo pari. In simili casi, il contatore che si occupa di tenere traccia dei nodi rilevati viene inizializzato a 1.

Per evitare che il sistema di attivazione indiretta possa procedere a cascata, provocando il risveglio di nodi anche molto lontani dalla RSU (flooding), si impone un limite massimo al numero di salti di attivazione ai quali la richiesta può andare incontro, sfruttando un concetto simile a quello di TTL [Tan02] previsto per il protocollo IP: ogni messaggio di response indica in un campo denominato `Hops` il numero di salti di attivazione che sono stati effettuati fino a questo momento. Se un nodo idle riceve una risposta con un valore di hop superiore al limite massimo prestabilito[2] il messaggio viene scartato e la catena di attivazioni indirette viene pertanto interrotta. Nella figura (parti b, c, d) si nota come il nodo `a` imposta tale valore a 0 nel suo messaggio di risposta in quanto attivato direttamente dalla RSU. I nodi `b` e `c` vengono attivati indirettamente da `a` e presentano un valore pari a 1 e così via per i restanti nodi `d` ed `e`.

Con una semplice variazione del valore massimo ammesso per i salti di attivazione è possibile alterare il comportamento del sistema per fare in modo

[2]Il numero massimo di salti è controllato dall'attributo `MaxHops` del modulo `BehaviourGroupSetupReactive`.

che esso "sparga" una richiesta di attivazione fino a dei nodi molto distanti oppure imporre che questa resti confinata in un'area circoscritta. In ogni caso, si tenga presente che un salto di attivazione può avvenire soltanto nel caso in cui i veicoli investiti dal segnale risultino avere una direzione conformante a quella stabilita inizialmente dai nodi attivati per via diretta dalla RSU. Negli scenari d'uso reali inoltre, a meno di rettilinei particolarmente lunghi, la conformazione del reticolo stradale non prevede quasi mai dei lunghi tratti a direzione uniforme a partire dagli incroci semaforici; ne consegue che è la particolare topologia dell'ambiente che circoscrive in primo luogo la capacità di una richiesta di formazione di allontanarsi dalla RSU che la origina, indipendentemente dal limite che viene manualmente imposto ai salti di attivazione.

Un ulteriore parametro che influenza il comportamento del protocollo durante la prima fase è la durata dell'intervallo temporale nel quale i nodi raccolgono passivamente le richieste provenienti da altri loro pari. Superato questo lasso di tempo la fase 1 viene completata e i nodi tornano a comunicare per stabilire chi tra loro verrà eletto come coordinatore del gruppo. L'evento che segna il passaggio dalla prima alla seconda fase coincide con l'invio del primo messaggio di offerta (*bid*), ovvero quando uno tra i nodi attivati termina l'attesa di ulteriori messaggi di richiesta e segnala la propria candidatura indicando quanti nodi esso è riuscito a individuare.

Per evitare che i nodi che vengono investiti per primi dalla richiesta di formazione del gruppo risultino avvantaggiati rispetto a quelli più distanti, si stabilisce che la durata di tale intervallo temporale sia scelta casualmente da ogni nodo mediante una distribuzione uniforme tra un valore minimo e massimo. Questa scelta aiuta inoltre a scongiurare che più nodi si sovrappongano tra loro e inviino entrambi un messaggio in istanti di tempo talmente ravvicinati da provocare una collisione tra le trame. L'utilizzo di intervalli e ritardi casuali ricorre numerosissime volte nelle varie logiche di protocollo; esso permette infatti di alleviare con poco sforzo i numerosi problemi derivanti dalle collisioni tra pacchetti inviati in istanti troppo vicini. Relativamente alla determinazione dell'intervallo di attesa per l'invio del primo messaggio di offerta, gli attributi che ne controllano il tempo minimo e massimo so-

no rispettivamente `TimeBidMin` e `TimeBidMax`; nei diversi test effettuati si è scelto di utilizzare dei valori che vanno da 150ms a 250ms.

Riassumendo, immediatamente dopo aver inviato il proprio messaggio di risposta, ogni nodo innesca un timer di durata casuale scelta secondo le modalità appena esposte e inizia l'ascolto di ulteriori risposte. Il primo tra i veicoli attivati che esaurisce il tempo di attesa passiva procede con la seconda fase del protocollo inviando la propria candidatura. Tutti i nodi che ricevono tale messaggio terminano la prima fase, indipendentemente dal tempo di attesa residuo.

Fase 2: elezione del group leader

Il primo nodo attivo che invia un messaggio di bid segna l'inizio della seconda fase e risveglia tutti i veicoli nel suo range operativo dichiarando loro il numero di nodi che esso ha rilevato durante la prima fase. Si avvia pertanto una sorta di asta tra i nodi nella quale chi tra loro ha contato un numero di veicoli maggiore rispetto a quelli dichiarati nell'ultimo messaggio di bid può "rilanciare" l'offerta inviando un'ulteriore offerta. Per fare in modo che i vari contatori rimangano stabili durante questa procedura, ogni nodo che passa dalla prima alla seconda fase diventa insensibile a ulteriori messaggi di risposta e mantiene pertanto inalterato il proprio conteggio. Quando il veicolo che dispone del contatore più alto avrà inviato la propria offerta nessun nodo nelle vicinanze sarà in grado di rilanciare; esaurito un breve periodo di timeout, l'autore dell'offerta più alta dichiarerà la propria vittoria con un messaggio broadcast (MT_GROUP_SETUP_IAMLEADER) e assumerà il ruolo di leader per il gruppo, terminando con successo il protocollo di formazione.

In maniera analoga a quanto già descritto per la fase 1, immediatamente dopo l'invio di un messaggio di bid il mittente innesca un timer di durata casuale scelta uniformemente tra un valore minimo e massimo, entrambi parametrizzabili rispettivamente mediante gli attributi `TimeLeaderMin` e `TimeLeaderMax`. Terminato tale periodo, se l'offerta precedentemente trasmessa non è stata superata, esso assume il controllo del gruppo e invia un

Tabella 4.3: Formato dei messaggi di offerta (bid).

Campo	Valore	
`ProtocolId`	`PID_GROUP_SETUP`	
`MessageType`	`MT_GROUP_SETUP_BID`	
Campo	Tipo	Descrizione
`RSUId`	`NodeId`	Identificatore della RSU che richiede la formazione del gruppo.
`Direction`	double	Direzione per la quale si è scelto di formare un gruppo.
`Bid`	intero	Entità dell'offerta, ovvero il numero di nodi che il mittente ha registrato durante la prima fase.

ultimo messaggio conclusivo (IAmLeader) per informare i nodi vicini della sua avvenuta elezione e per comunicare l'identità del gruppo appena formato. La struttura dati `Group` contenuta nel messaggio conclusivo include sia la direzione conformante scelta per il gruppo, sia l'identificatore univoco per il plotone (`GroupId`). I nodi che ricevono tale pacchetto hanno quindi conferma definitiva dell'identità della RSU alla quale faranno riferimento, dell'id del leader per il gruppo e del valore di direzione che occorre mantenere.

Durante il periodo di attesa che precede l'invio del messaggio conclusivo di leadership, può accadere che un nodo che ha già presentato la propria candidatura – o che è comunque in procinto di inviare la propria – riceva un messaggio di bid contenente un valore superiore al proprio contatore. Ogni nodo che vede la propria offerta (indipendentemente dal fatto che questa sia stata già piazzata o sia ancora in attesa) superata da un altro veicolo deve immediatamente annullare il timer che precede l'invio del messaggio di leadership e attendere in maniera passiva che un altro nodo completi il protocollo dichiarandosi leader del gruppo. Normalmente un nodo che non ha ancora piazzato un'offerta recede dall'asta anche se riceve un messaggio di bid pari al proprio contatore; in ogni caso, nella remota eventualità che si verifichino delle parità tra due offerte, si stabilisce per convenzione che il vincitore è il nodo con l'identificatore (`NodeId`) più alto tra i contendenti.

È chiaramente preferibile evitare che la seconda fase del protocollo di elezione degeneri in un flooding di messaggi bid. Si vuole cioè impedire l'invio

Tabella 4.4: Formato dei messaggi di leadership (IAmLeader).

Campo	Valore
`ProtocolId`	`PID_GROUP_SETUP`
`MessageType`	`MT_GROUP_SETUP_IAMLEADER`

Campo	Tipo	Descrizione
`Group`	`Group`	Identificatore e direzione del gruppo formato.

eccessivo e incontrollato di candidature che porterebbe la maggior parte dei nodi attivi a comunicare agli altri il valore del proprio contatore interno. Per regolamentare ed ottimizzare questa fase è stato stabilito che ciascun nodo attivo ha a disposizione un solo messaggio di bid; questo può essere inviato esclusivamente in due occasioni: allo scadere del tempo di attesa che separa la prima e la seconda fase, oppure per rilanciare un'offerta più bassa di quella in proprio possesso.

Per scongiurare l'affollamento e l'eventuale sovrapposizione di più offerte in risposta ad un bid, ciascun nodo – analogamente a quanto avviene nella prima fase, precedentemente all'invio del messaggio di risposta – deve attendere un breve intervallo di tempo casuale limitato dall'attributo `TimeResponseMax` prima di poter trasmettere la propria controfferta. L'intenzione di un nodo ad effettuare un'offerta viene in ogni caso inibita nel momento in cui esso riceve un messaggio di bid contenente un valore pari o superiore a quello del proprio contatore.

Le regole appena formalizzate prevedono un'unica eccezione nel caso in cui un nodo attivo non rilevi alcun veicolo nel proprio raggio operativo: in questa eventualità l'elezione del leader per il gruppo è banale e non è necessario ricorrere all'invio della propria candidatura. In sintesi, se un nodo termina la prima fase e dispone di un conteggio dei veicoli pari a zero, esso attende comunque i due intervalli di tempo previsti (attesa per il bidding e attesa per la leadership) terminati i quali potrà procedere all'invio del messaggio `IAmLeader` senza aver prima comunicato la propria offerta.

Un esempio completo del protocollo appena descritto è dato dal diagramma di sequenza in Figura 4.2, dove vengono rappresentati i messaggi scam-

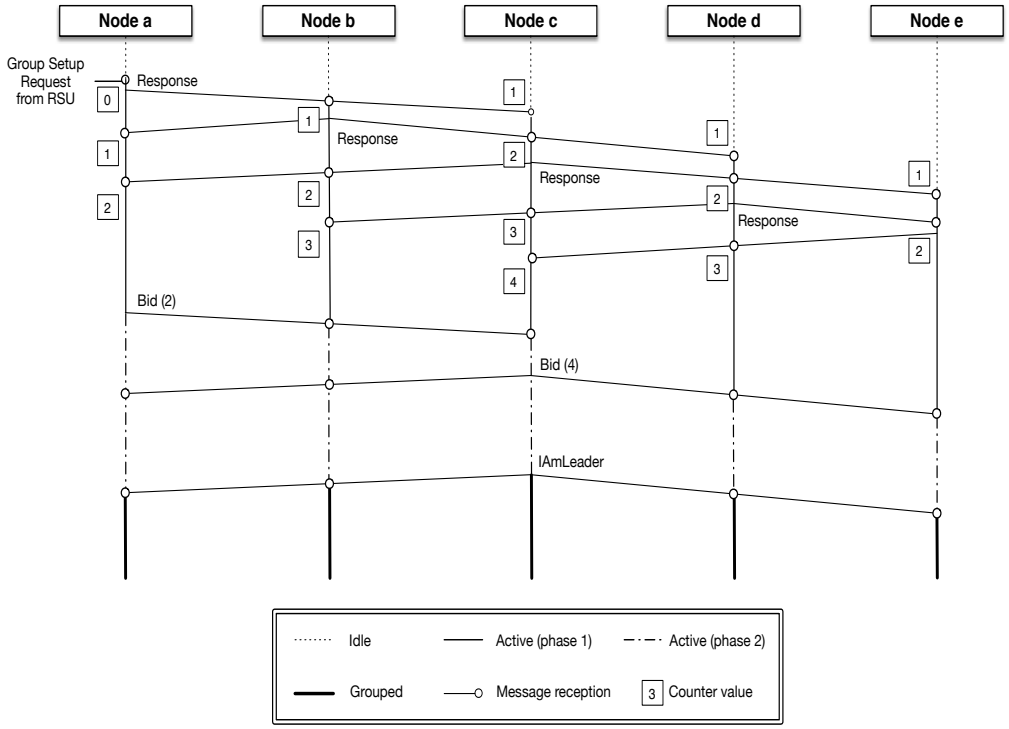

Figura 4.2: Diagramma di sequenza dei messaggi scambiati tra i nodi di Figura 4.1 durante la formazione del gruppo.

biati dagli stessi nodi della Figura 4.1 nel corso delle due fasi sopra descritte. Nella Tabella 4.5 vengono elencati i numerosi attributi messi a disposizione dal modulo `BehaviourGroupSetupReactive` per consentire al progettista di personalizzarne il comportamento.

4.1.2 Strategia proattiva

In condizioni di densità di traffico molto elevate, il protocollo di formazione dei gruppi appena esposto presenta alcuni svantaggi che hanno spinto a considerare lo sviluppo di una sua versione alternativa al fine di tentare di ottenere migliori performance negli scenari particolarmente intensi dal punto di vista del numero di nodi che si avvicinano in media alla stessa RSU. Le principali dissimilitudini tra le due strategie risiedono nelle diverse ipotesi dalle quali esse traggono ispirazione e nelle differenti modalità di scelta degli

Tabella 4.5: Attributi previsti per il modulo `BehaviourGroupSetupReactive`.

Attributo	Tipo	Default	Descrizione
`Enabled`	booleano	True	Determina se il comportamento è attivo o meno durante la simulazione.
`TimeResponseMax`	intero	15	Intervallo massimo di attesa (ms) prima dell'invio di messaggi `Response` o `Bid` (in caso di controfferta).
`TimeBidMin`	intero	150	Intervallo minimo di attesa (ms) prima del passaggio dalla prima alla seconda fase.
`TimeBidMax`	intero	250	Intervallo massimo di attesa (ms) prima del passaggio dalla prima alla seconda fase.
`TimeLeaderMin`	intero	100	Intervallo minimo di attesa (ms) prima dell'invio del messaggio di leadership.
`TimeLeaderMax`	intero	200	Intervallo massimo di attesa (ms) prima dell'invio del messaggio di leadership.
`MaxHops`	intero	2	Numero massimo di salti di attivazione consentiti.

istanti di tempo per l'invio dei messaggi. Questi due punti verranno trattati nel dettaglio prima di darne una descrizione precisa e completa.

Conoscenza pregressa dei nodi circostanti

Lo svantaggio principale dato dalla strategia reattiva è l'elevato numero di messaggi di risposta che vengono inviati in un intervallo di tempo relativamente breve. Per poter misurare con quanti nodi ciascun veicolo è in grado di comunicare, ognuno di essi deve infatti inviare un pacchetto response per informare i suoi vicini della propria presenza. Questo comportamento, nel caso di densità particolarmente elevate, può portare ad un'esplosione di messaggi tale da compromettere seriamente l'esito del protocollo di formazione a causa delle numerose sovrapposizioni di trame a livello fisico. Tali collisioni possono inoltre investire numerosi nodi impegnati in altri protocolli, diminuendo così le performance dell'intero sistema in un'area piuttosto vasta.

Per tentare di alleviare queste difficoltà, il protocollo di formazione di tipo proattivo parte dal presupposto che i nodi in avvicinamento alla RSU conoscano a priori i diversi veicoli presenti nelle vicinanze e il numero di nodi visti da ciascuno di essi. Sotto tale ipotesi, l'elezione del leader per il gruppo diventa un'operazione estremamente più semplice in quanto ogni nodo – nel momento stesso in cui viene attivato – è in grado di sapere se esso è o meno il candidato ideale per tale ruolo. In caso affermativo esso può dichiararsi come leader dopo aver atteso un breve intervallo di tempo; se un nodo invece conosce un candidato migliore spedisce a quest'ultimo un messaggio di invito per attivarlo. Alla ricezione di tale messaggio un nodo verifica personalmente se esso è conforme alla direzione prevista per il gruppo e in caso affermativo attende un certo quantitativo di tempo prima di concludere la procedura di group formation dichiarandosi leader – se non viene nel frattempo preceduto da altri veicoli.

La facoltà di disporre di una conoscenza dei nodi presenti nelle vicinanze anche nei momenti in cui questi risultano inattivi è possibile soltanto se si ammette di poter consentire uno scambio di messaggi tra nodi in stato idle, ipotesi precedentemente esclusa dal primo dei due princìpi esposti all'inizio del paragrafo 4.1. Per poter utilizzare una strategia proattiva si deve quindi poter "rilassare" tale vincolo e istruire ogni veicolo presente nella simulazione – ad ovvia eccezione di quelli di tipo shadow – affinché esso invii a intervalli regolari e opportunamente ampi dei messaggi denominati *heartbeat* al fine di informare periodicamente i nodi vicini della propria presenza e di altre sue proprietà utili (direzione, posizione, nodi conosciuti, ... etc). In questo modo ciascun veicolo è in grado di costruire e mantenere aggiornata una propria *mappa dei nodi* circostanti e verificare in ogni momento le loro caratteristiche.

Il servizio di mappatura dei nodi in prossimità è offerto dal componente `BehaviourNodeMap`; il suo funzionamento e i dettagli sulle modalità con cui esso realizza le funzionalità sopra accennate verranno approfonditi nel paragrafo 4.2. Per il momento si assumerà di potere utilizzare la struttura dati fondamentale da esso messa a disposizione – `NodeMap` – la quale consiste in una collezione di nodi conosciuti e delle loro relative proprietà, tra cui posizione, velocità e numero di veicoli da loro rilevati. Ovviamente tali

informazioni costituiscono una stima necessariamente imperfetta dello stato reale dell'ambiente in quanto la loro freschezza è in proporzione diretta alla frequenza con cui vengono ricevuti i messaggi di heartbeat di ciascun nodo. Nondimeno esse risultano di grande aiuto per lo svolgimento del protocollo esposto di seguito.

Scelta dinamica degli intervalli temporali

Una seconda differenza tra le due strategie di formazione dei gruppi è data dalla modalità con la quale vengono scelti gli intervalli di attesa nelle varie fasi del processo di elezione. Nella logica di tipo reattiva tali intervalli vengono sempre scelti in maniera uniforme tra un valore di minimo e uno di massimo, indipendentemente dalle particolari condizioni contingenti dei veicoli che partecipano al protocollo. In generale questa scelta garantisce comunque una buona resa a fronte di un costo implementativo molto contenuto; esiste comunque l'eventualità che degli outcome particolarmente sfortunati portino a un'elezione non ottimale ma questa è occasionalmente tollerabile se si considera che lo scopo ultimo delle diverse logiche implementate non è quello di arrivare a una perfetta conoscenza delle condizioni di traffico dell'ambiente ma di ottenerne un'euristica utilizzabile da applicazioni di più alto livello.

Nel momento in cui un elevato numero di veicoli si trova a dover scegliere un istante per l'invio di un messaggio attraverso la stessa procedura uniforme, si possono verificare frequenti collisioni tra le trame dei pacchetti, visto che la probabilità che due o più nodi scelgano dei valori temporali troppo poco distanti tra loro cresce velocemente con l'aumentare dei partecipanti. Siccome il protocollo di tipo proattivo si propone come alternativa maggiormente scalabile in condizioni di elevata densità si è scelto di ricorrere a un metodo differente per l'ottenimento di periodi di attesa casuali.

Come si capirà meglio nel seguito, l'intervallo di attesa maggiormente significativo per l'elezione proattiva è quello che precede l'invio del messaggio di leadership spedito dal candidato migliore per dichiararsi coordinatore del gruppo in via definitiva. Nel seguito verrà spiegato come la densità netta

percepita da ciascun nodo vene utilizzata come indice della sua qualità come potenziale coordinatore del gruppo. In base a tale parametro si calcola il valore di ritardo utilizzando la seguente espressione:

$$Delay = \frac{\texttt{DelayBase}}{cdensity^{\texttt{WaitExponent}}} \pm rnd$$

dove `DelayBase` e `WaitExponent` indicano i valori dei rispettivi attributi definiti nella Tabella 4.7, *cdensity* corrisponde al valore di densità netta definita di seguito e *rnd* indica un breve valore scelto casualmente con distribuzione uniforme tra zero e `DelaySkew`. La Figura 4.3 illustra i valori di tale espressione in funzione del valore di densità locale al nodo che ne fa uso.

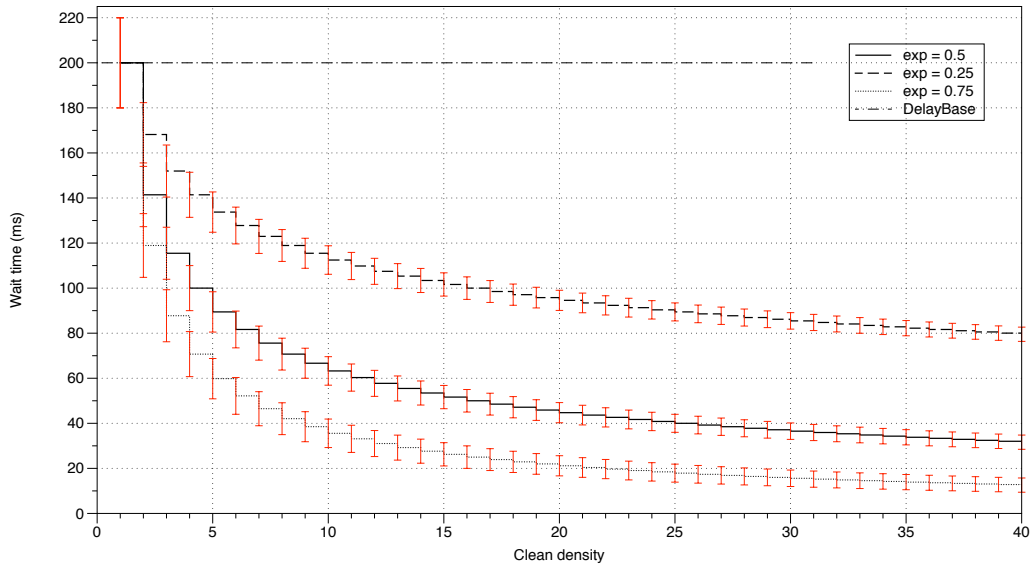

Figura 4.3: Distribuzione degli intervalli di attesa in base al valore della densità netta del nodo. Vengono rappresentate tre curve rappresentative di diversi valori dell'attributo `WaitExponent`; il valore dell'attributo `DelayBase` viene assunto pari a 200. Vengono rappresentate in rosso le barre relative allo skew aggiunto o sottratto dal valore di base.

Per comprendere meglio tale espressione si consideri che un nodo risulta tanto più avvantaggiato nell'elezione al ruolo di leader quanto più breve è l'intervallo di tempo che deve attendere prima di proclamarsi tale a seguito della ricezione di un invito. Per incentivare l'elezione dei nodi con un valore di *cdensity* più alto (cioè che vedono più nodi all'interno del loro raggio) si

pone tale variabile come denominatore del valore di partenza `DelayBase` e si aumenta il grado dell'espressione con un esponente configurabile in modo che la curva tenda asintoticamente a zero. L'esponente `WaitExponent` deve essere ovviamente maggiore di zero affinché il ritardo diminuisca al crescere della densità percepita dal nodo. Viene infine aggiunto un piccolo skew casuale per evitare che due nodi che dispongono dello stesso valore di *cdensity* calcolino lo stesso valore di ritardo e sincronizzino le loro trasmissioni.

Procedura di elezione proattiva

L'intera procedura di formazione proattiva si svolge in un'unica fase e risulta concettualmente più semplice se confrontata con quella di tipo reattivo, grazie sopratutto alle forti ipotesi di partenza che sono state assunte. L'attivazione diretta dei nodi da parte dei messaggi beacon provenienti dalle RSU avviene analogamente a quanto esposto in precedenza: ogni ricevente verifica la propria conformità rispetto alle direzioni specificate e in caso affermativo si attiva e dà inizio alla procedura di elezione. A differenza di quanto avviene nella strategia reattiva, nel protocollo proattivo i nodi non reagiscono all'attivazione con un messaggio di risposta ma effettuano una scansione della `NodeMap` locale alla ricerca di un candidato ottimale.

Il parametro che viene utilizzato da ciascun nodo come indicatore di preferenza al ruolo di group leader è la cosiddetta *densità netta*[3] (clean density), ovvero il numero di veicoli non raggruppati che un nodo vede all'interno del proprio raggio tali per cui la loro direzione attuale risulta conformante a quella scelta nel processo di group formation. Ad esempio, se un veicolo viene attivato per la formazione di un gruppo relativamente alla direzione di valore 30° esso conterà nella sua mappa locale il numero di nodi a lui conosciuti che non fanno già parte di un gruppo e stanno procedendo in una direzione conformante a 30°. Se a seguito di tale ricerca un nodo attivo scopre di essere a conoscenza di un veicolo con una densità netta maggiore della propria esso procederà ad inviargli un messaggio unicast di *invito* per comunicargli la propria preferenza.

[3]Consultare il paragrafo 4.2 per un confronto fra le diverse tipologie di densità definite dal sistema.

Tabella 4.6: Formato dei messaggi di invito.

Campo	Valore	
`ProtocolId`	`PID_GROUP_SETUP`	
`MessageType`	`MT_GROUP_SETUP_INVITE`	
Campo	Tipo	Descrizione
`RSUId`	`NodeId`	Identificatore della RSU che richiede la formazione del gruppo.
`Direction`	double	Direzione per la quale si è scelto di formare un gruppo.

Nel caso scelga di spedire un messaggio di invito, il mittente delega di fatto la creazione del gruppo al suo candidato e rimane passivamente in attesa di un messaggio di leadership che annunci il completamento della procedura, indipendentemente dallo specifico nodo mittente che assumerà quindi il ruolo di leader. Per scongiurare situazioni di stallo i mittenti dei messaggi di invito prevedono in ogni caso un periodo di timeout terminato il quale procederanno essi stessi a prendere il controllo del gruppo effettuando il broadcast del pacchetto di leadership. Tale intervallo è di durata pari al valore dell'attributo `Timeout` più o meno un tempo di *skew* limitato in modulo dall'attributo `DelaySkew`.

Alla ricezione di un messaggio di invito un nodo procede ad attivarsi – se non lo è già – sempre verificando che la sua direzione sia conformante con quella specificata originariamente dalla RSU. Prima di poter assumere il ruolo di group leader, un nodo attivato mediante invito deve attendere per un periodo di tempo scelto in maniera inversamente proporzionale al suo valore di densità netta. È infatti assai frequente che all'interno di una stessa procedura di formazione più nodi decidano di inviare molteplici messaggi di invito (mai più di uno a testa) verso diversi candidati; ciascuno di questi potrebbe decidere di assumere il comando dando luogo a un'inefficiente frammentazione del gruppo. Per favorire l'elezione del leader migliore si sceglie di garantire un tempo di attesa più breve ai candidati più promettenti, ovvero a quelli che conoscono un maggior numero di potenziali membri per il gruppo, come illustrato nella Figura 4.3.

Il primo nodo che invia un messaggio di leadership (vedi la Tabella 4.4)

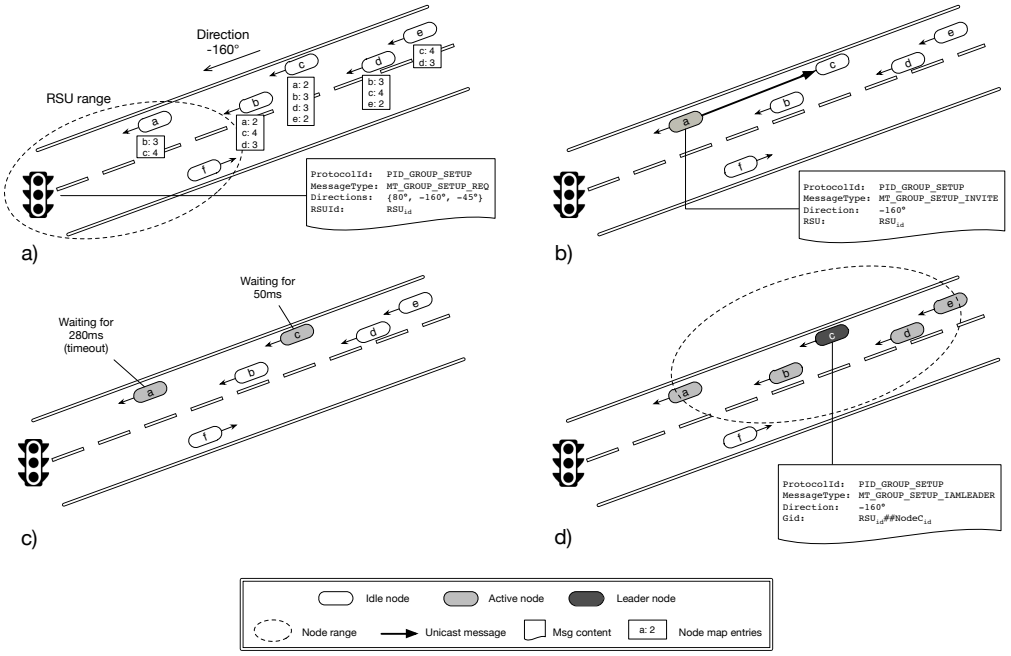

Figura 4.4: Schematizzazione del processo di formazione dei gruppi con strategia proattiva. L'RSU invia dei beacon periodici che attivano i nodi nelle direzioni specificate (a). Ogni nodo attivato verifica la conoscenza di un valido candidato all'interno della propria nodemap ed eventualmente invia un messaggio di invite esclusivamente al nodo scelto (b). Il primo nodo che termina il proprio periodo di attesa o esaurisce il timeout (c) assume il ruolo di leader inviando un messaggio broadcast (d).

Tabella 4.7: Attributi previsti per il modulo `BehaviourGroupSetupProactive`.

Attributo	Tipo	Default	Descrizione
`Enabled`	booleano	True	Determina se il comportamento è attivo o meno durante la simulazione.
`DelaySkew`	intero	20	Valore massimo (ms) che viene aggiunto o sottratto come *skew* agli intervalli temporali scelti casualmente.
`DelayBase`	intero	200	Intervallo base di attesa (ms) per l'espressione di calcolo del tempo di attesa in caso di invitation.
`WaitExponent`	double	0.5	Esponente per l'espressione di calcolo del tempo di attesa in caso di invitation.
`Timeout`	intero	250	Periodo di timeout (ms) successivo all'invio del messaggio di invitation.

assume il ruolo di leader per il gruppo e informa tutti i nodi nelle vicinanze della sua elezione, anche eventuali veicoli ancora inattivi. I rimanenti nodi attivi che ricevono tale comunicazione annullano qualsiasi tipo di attesa o timeout e cedono il controllo del plotone al mittente.

Un esempio dell'intera procedura di elezione proattiva viene schematizzato nella Figura 4.4, mentre la Tabella 4.7 riassume gli attributi messi a disposizione dal modulo `BehaviourGroupSetupProactive`.

4.2 Mappatura dei nodi circostanti

Nell'architettura realizzata si presenta in due occasioni la necessità di gestire e mantenere aggiornata una struttura dati locale ai singoli nodi che permetta loro di conoscere quanti e quali veicoli sono presenti in un dato momento all'interno del proprio range operativo. La prima occasione si manifesta nella gestione del normale ciclo di vita di un gruppo esposta nel paragrafo 4.3. In particolare, il controllore di un plotone deve necessariamente disporre di una mappa dettagliata dei vari membri del gruppo per poter svolgere correttamente le proprie attività e integrare i dati a sua disposizione. In secondo luogo, è stato ampiamente discusso nel paragrafo 4.1.2 come il modulo per la formazione proattiva dei gruppi necessiti di conoscere

la densità netta dei nodi conosciuti per procedere rapidamente all'elezione del group leader.

Il modulo del sistema che permette di disporre di tali informazioni è il componente `BehaviourNodeMap`, il quale si occupa di organizzarle in una struttura dati denominata `NodeMap` e costituita da una lista di elementi di tipo `NodeInfo` ognuno dei quali mantiene i dati più recenti disponibili per un particolare nodo. Questo modulo è piuttosto atipico se confrontato con le altre logiche di protocollo per il fatto che esso offre delle funzionalità agli altri comportamenti e non è pertanto perfettamente isolato. Esso è in grado di svolgere il proprio operato in maniera del tutto indipendente ma quei behaviour che ne fanno uso necessitano di un suo riferimento per recuperare la mappa dei nodi quando necessario.

Tabella 4.8: Informazioni disponibili all'interno della struttura `NodeInfo`.

Attributo	Tipo	Descrizione
`NodeId`	`NodeId`	Identificatore univoco del nodo.
`GroupId`	`GroupId`	Eventuale identificatore del gruppo al quale il nodo appartiene attualmente.
`Speed`	double	Velocità attuale del nodo.
`Direction`	double	Direzione attuale del nodo.
`Position`	`Vector2D`	Posizione attuale del nodo.
`LastSeen`	`Time`	Istante temporale nel quale sono state pervenute le informazioni più recenti per il nodo.
`Density`	`Density`	Valori di densità (assoluta, netta, conformante) relativi al nodo.

Come è possibile notare osservando la Tabella 4.8, le informazioni di stato per ciascuna entry disponibili all'interno di questa struttura dati comprendono l'identificatore univoco del nodo in questione, la sua posizione, velocità e direzione corrente, l'ultimo istante temporale nel quale i dati a disposizione sono stati aggiornati, l'identificatore dell'eventuale gruppo al quale il nodo può fare attualmente riferimento e tre diversi valori di densità contenuti nella struttura `Density`. Per ogni nodo infatti, viene tenuta traccia della sua *densità assoluta*, ovvero il numero complessivo di nodi che esso vede all'interno del proprio raggio di comunicazione, la *densità conformante*, cioè quanti

tra questi dispongono di una direzione conformante al nodo stesso e infine la *densità netta* (o *clean density*), ovvero quanti nodi tra quelli conformanti non appartengono momentaneamente ad alcun gruppo (cioè sono in stato idle).

Per recuperare le informazioni relative ai diversi nodi presenti nelle vicinanze il modulo `BehaviourNodeMap` sfrutta due diverse fonti: in primo luogo esso analizza costantemente tutto il traffico in entrata al nodo su cui risiede e in base ad esso ricostruisce e tiene traccia in maniera passiva delle informazioni disponibili per diversi mittenti incontrati. Per integrare questo processo – specialmente nei periodi di tempo in cui il nodo è inattivo – esso ricorre attivamente al protocollo PID_HEARTBEAT per inviare costantemente dei messaggi CAM denominati *heartbeat* (MT_HEARTBEAT_INFO) finalizzati unicamente alla divulgazione delle proprie informazioni di stato ai nodi circostanti. Ciascuna di queste due modalità di recupero dati verrà di seguito analizzata nel dettaglio.

Ogniqualvolta il componente `BehaviourNodeMap` rilevi l'esistenza di un qualsiasi nodo nelle vicinanze e riesca ad ottenere una serie di dati sul suo conto, esso aggiorna automaticamente il campo `LastSeen` per l'entry corrispondente, in modo da tenere traccia dell'ultimo istante in cui la presenza di un veicolo è stata confermata. Se a partire da tale momento non pervengono più informazioni relative a quel nodo entro un certo periodo di timeout, questi viene rimosso dalla nodemap. L'intervallo che regola il timeout tollerato per ciascun nodo è regolabile attraverso l'attributo `Timeout`.

4.2.1 Analisi del traffico in ingresso

Nel paragrafo 3.2.2 è stato spiegato come ogni messaggio scambiato tra due nodi ITS qualunque contiene sempre un header generale (`ItsHeader`) che trasporta numerose informazioni sull'autore del pacchetto. In particolare, queste ultime sono elencate nella Tabella 3.3 e comprendono indicazioni sulla posizione, direzione e identità del nodo mittente. Per sfruttare questa forma di conoscenza implicita il modulo `BehaviourNodeMap` dichiara al controllore di essere interessato alla ricezione di *qualsiasi* messaggio pervenuto al nodo sottostante, indipendentemente dal suo PID. Al momento della ricezione di

un pacchetto – in maniera del tutto trasparente alla logica alla quale esso è realmente destinato – vengono analizzati i diversi campi del suo header principale per ricavare ogni genere di informazione utile sul nodo che lo ha emesso.

Ulteriori informazioni sullo stato dei mittenti si possono ottenere considerando il `ProtocolId` di ogni messaggio scambiato. Gli header relativi ad alcuni protocolli (in particolare per i messaggi di tipo MT_GROUP_BEACON, vedi paragrafo 4.3) contengono infatti diversi campi utili per dedurre alcune proprietà importanti relative ai nodi vicini. Ad esempio, se si scopre che un veicolo ha appena inviato un messaggio di tipo MT_GROUP_SETUP_IAMLEADER si saprà con certezza che esso è il leader di un gruppo appena formato e si potranno quindi estrarre le informazioni sul relativo `GroupId` dall'header del pacchetto.

Questa forma di conoscenza implicita sui nodi circostanti è particolarmente utile in quanto ottenibile in maniera gratuita, semplicemente ascoltando le conversazioni in corso. In particolare, essa garantisce dei risultati ottimali alla presenza di nodi relativamente vicini a una RSU e già attivati per un particolare gruppo, visto che i diversi membri dei plotoni scambiano periodicamente numerosi messaggi per il mantenimento del ciclo di vita del gruppo (vedi paragrafo 4.3). Tuttavia, per poter ottenere informazioni anche su nodi che non sono ancora stati attivati è necessario prevedere il protocollo ausiliario illustrato nel paragrafo seguente.

4.2.2 Scambio di heartbeat

Un nodo inattivo, in accordo con le linee guida delineate nel paragrafo 4.1, rimane normalmente silente e resta quindi impossibile da rilevare mediante la tecnica passiva appena esposta. Per ovviare a questo problema si deve necessariamente poter rilassare il principio che stabilisce che i nodi inattivi non debbono in alcun caso generare traffico di rete e tollerare l'invio sporadico di semplici messaggi detti *heartbeat* e analoghi al principio dei *Cooperative Awareness Message* frequentemente adottati in letteratura e recentemente standardizzati in [ETS13].

Nel presente sistema non si farà ricorso a un particolare standard ufficiale sull'uso dei CAM ma se ne utilizzerà un semplice formato ad-hoc illustrato nella Tabella 4.9. I campi presenti al suo interno sono complementari a quelli dell'header principale (`ItsHeader`) e permettono complessivamente di ottenere tutte le informazioni utili su un nodo, anche nei periodi di tempo in cui questo resta inattivo. È inoltre previsto un messaggio heartbeat apposito (MT_HEARTBEAT_LEAVE) per segnalare l'intenzione di un nodo di essere rimosso dalle mappe dei suoi vicini, ad esempio per segnalare un suo imminente spegnimento.

Tabella 4.9: Formato dei messaggi heartbeat.

Campo	Valore	
`ProtocolId`	`PID_HEARTBEAT`	
`MessageType`	`MT_HEARTBEAT_INFO`	
Campo	Tipo	Descrizione
`Speed`	double	Velocità attuale del nodo.
`Density`	`Density`	Valori di densità (assoluta, netta, conformante) relativi al nodo.

Il modulo `BehaviourNodeMap` adotta una strategia intelligente per ridurre al minimo il ricorso ai messaggi CAM: quando possibile si considerano le informazioni ottenute per via indiretta mediante ascolto del traffico in ingresso. In linea di principio si prevede l'invio di messaggi heartbeat ad intervalli regolari sufficientemente distanti tra loro per non comportare un eccessivo costo in termini di pacchetti distribuiti. Il periodo di tempo che intercorre tra due invii successivi viene perturbato con un leggero skew casuale per evitare che due o più nodi possano inavvertitamente sincronizzarsi e cancellarsi vicendevolmente le relative trame di dati. Se il modulo rileva che il proprio controllore ha effettuato l'invio di un qualsiasi messaggio relativo a un altro protocollo, l'invio dell'heartbeat successivo viene ritardato di un intero periodo. In questo modo è possibile evitare l'invio di pacchetti CAM per quei nodi che sono comunque impegnati in una conversazione e che quindi forniscono informazioni circa la propria presenza in via indiretta. Non appena il nodo

resta silenzioso per un tempo eccessivamente lungo, il protocollo di heartbeat riprende ad operare normalmente.

Tabella 4.10: Attributi previsti per il modulo `BehaviourNodeMap`.

Attributo	Tipo	Default	Descrizione
`Enabled`	booleano	True	Determina se il comportamento è attivo o meno durante la simulazione.
`RsuSilence`	booleano	False	Impedisce l'invio di messaggi heartbeat se viene rilevata la presenza di una RSU.
`Timeout`	intero	4100	Timeout (ms) massimo superato il quale un nodo viene rimosso dalla nodemap se non sono pervenute informazioni sul suo stato.
`TimeCheck`	intero	500	Intervallo di tempo (ms) che separa due controlli successivi della nodemap.
`TimeHeartbeat`	intero	2000	Intervallo di tempo (ms) che separa l'invio di due heartbeat successivi.

Il componente appena descritto garantisce una buona versatilità al sistema in quanto può essere utilizzato da qualsiasi logica aggiuntiva e il protocollo heartbeat può essere esteso per includere ulteriori informazioni oltre a quelle attualmente previste. Il suo utilizzo relativo ai nodi inattivi deve però essere considerato con attenzione: l'invio costante nel tempo di CAM comporta un traffico di rete non sempre giustificato e le trame di tali pacchetti potrebbero interferire con le normali attività dei nodi circostanti a causa dell'elevata probabilità che hanno di collidere con le altre trame trasmesse, specialmente in ambienti ad elevata densità. Il leggero ma costante traffico generato per mezzo del protocollo heartbeat può inoltre contribuire al consumo energetico di quei veicoli che sfruttano dispositivi con autonomia limitata per partecipare al sistema[4]. Nel corso dell'analisi dei test effettuati verrà spesso esaminato l'effetto che i messaggi CAM comportano sulla performance complessiva del sistema realizzato.

[4]Si veda la definizione di mid-class vehicle al paragrafo 2.3

4.3 Protocolli di gestione del ciclo di vita

Nel corso del paragrafo 2.2.3 sono state ampiamente discusse le funzionalità che un gruppo deve realizzare per informare adeguatamente l'RSU a cui fa riferimento dello stato del traffico per quella direzione, così come viene complessivamente percepito per mezzo dei suoi membri. Ciascun componente del plotone deve pertanto raccogliere quante più informazioni possibili sul suo stato e su quello dei nodi circostanti e inviarle periodicamente al proprio leader che provvederà a sintetizzarle e a spedire un resoconto *fused* dei dati a sua disposizione alla propria RSU.

Compito secondario ma altrettanto importante del leader di un gruppo è quello di monitorare costantemente il proprio stato e quello dei propri nodi membri: occorre tenere sempre presente che i partecipanti di un plotone – incluso il leader stesso – corrispondono a veicoli dotati di una propria mobilità. Anche se i processi di group formation sopra descritti si concludono normalmente in un tempo relativamente breve (500 - 800ms) tale per cui – con le dovute approssimazioni – i nodi che vi partecipano possono essere considerati pressoché fermi, il ciclo di vita di un gruppo può durare da alcuni secondi a un paio di minuti; nel corso di questo lasso di tempo i vari nodi che lo compongono possono andare incontro a disconnessioni, guasti, cambiamenti di direzione e allontanamenti reciproci. Persino il leader è suscettibile delle stesse condizioni che possono portare un qualsiasi nodo ad uscire dal gruppo. Il sistema prevede inoltre che eventuali nodi inattivi che si trovino a transitare nelle vicinanze di un nodo leader possano decidere autonomamente di entrare a far parte del relativo gruppo, senza dover ricorrere ai protocolli di formazione descritti all'inizio del presente capitolo.

Un gruppo è quindi un'entità che subisce continue trasformazioni e i suoi membri possono da esso uscire e rientrare in qualsiasi momento. Spetta dunque al suo coordinatore controllare periodicamente quanti e quali nodi vi stanno partecipando e ottenere da questi le informazioni utili per lo svolgimento del suo compito. I diversi messaggi scambiati tra leader e membri che consentono la gestione del protocollo di gestione dei nodi di un gruppo rientrano all'interno del PID rappresentato dal valore PID_GROUP_MANAGEMENT

e comprendono un totale di sei diversi tipi di messaggi che verranno presentati nei paragrafi successivi.

La logica che definisce il comportamento di un nodo leader è contenuta nel modulo `BehaviourGroupLeader`, mentre quella relativa ad un generico membro di un gruppo viene formalizzata dal modulo `BehaviourGroupMember`. Questi comportamenti risultano normalmente inattivi all'interno del sistema e vengono attivati automaticamente nel momento in cui un nodo entra a far parte di un gruppo per poi tornare inerti al termine del suo ciclo di vita. Uno dei compiti principali del componente `ItsGroupController` è quello di occuparsi dell'attivazione e della sospensione dei comportamenti responsabili della gestione del ciclo di vita di un gruppo esposti di seguito.

4.3.1 Esplorazione dei nodi membri

La procedura con la quale il leader effettua periodicamente una scansione completa dei membri del proprio gruppo viene chiamata *probe* e ha il duplice scopo di recuperare i vari dati riguardanti lo stato dei singoli nodi e di verificare quanti tra loro risultano ancora presenti. Al termine di ogni esplorazione, il leader calcola una sintesi di tutti i dati raccolti e la invia alla propria RSU di riferimento.

La procedura di probe ha inizio nel momento in cui il leader del gruppo invia un messaggio MT_GROUP_BEACON e resta in attesa delle varie risposte dei propri membri entro un tempo limite da lui prefissato (*probe time*). Ogni nodo che risponde al segnale di beacon nei tempi previsti conferma la propria presenza e allo stesso tempo consegna le informazioni più recenti in proprio possesso. Sia i messaggi di richiesta spediti dal leader, sia quelli di risposta inviati dai diversi membri utilizzano lo stesso formato di messaggi illustrato nella Tabella 4.11.

Il ricevente di tali messaggi è quindi in grado di ottenere una nutrita serie di informazioni sul mittente. Ogni partecipante del gruppo – incluso il leader stesso – riutilizza il componente `BehaviourNodeMap` e delega automaticamente ad esso la costruzione e la gestione della `NodeMap` comprensiva dei diversi membri. Siccome i messaggi di group beacon vengono scambiati

Tabella 4.11: Formato dei messaggi di beacon scambiati tra membri del gruppo.

Campo	Valore	
`ProtocolId`	`PID_GROUP_MANAGEMENT`	
`MessageType`	`MT_GROUP_BEACON`	
Campo	Tipo	Descrizione
`Group`	`Group`	Identificatore e direzione del gruppo.
`Interval`	intero	Periodo di durata complessiva (ms) della procedura di probe entro il quale il leader attende le risposte dei membri.
`NumNodes`	intero	Numero di nodi attualmente noti al mittente.
`Speed`	double	Velocità attuale del mittente.
`Density`	`Density`	Valori di densità relativi al mittente.

molto più frequentemente rispetto al timeout previsto per il protocollo di heartbeat, tale componente ricorrerà esclusivamente all'analisi del traffico in ingresso e uscita per tenere traccia dei diversi veicoli, in accordo con quanto esposto nel paragrafo 4.2.1.

Una volta terminato il tempo stabilito per l'esplorazione dei diversi nodi, il leader verifica se l'RSU relativa al gruppo è presente nella nodemap[5] e in caso affermativo provvede a calcolare una media delle informazioni appena raccolte e a includerle in un messaggio MT_GROUP_INFO. Raccogliendo informazioni dai diversi leader attivi nelle varie direzioni di interesse, ogni RSU è in grado in questo modo di monitorare l'andamento nel tempo dei vari parametri utili a comprendere le condizioni attuali del flusso di veicoli in avvicinamento.

Le esatte tempistiche utilizzate per scandire e regolare la fase di esplorazione devono essere considerate attentamente al fine di scongiurare una scarsa resa dell'algoritmo e di conseguenza dell'intero sistema. Specialmente negli scenari caratterizzati da un'elevata densità, i gruppi tendono ad essere composti da numerosi membri i quali si trovano a loro volta a dovere operare

[5]Il messaggio di group info viene inviato solo se la relativa RSU è all'interno del raggio del leader. A seguito dei salti di attivazione o del protocollo di marker può infatti verificarsi la formazione di un gruppo in cui alcuni o tutti i membri risiedono all'esterno del campo di azione dell'RSU.

Tabella 4.12: Formato dei messaggi di info prodotti dal leader di un gruppo al termine della procedura di esplorazione dei nodi.

Campo	Valore	
`ProtocolId`	`PID_GROUP_MANAGEMENT`	
`MessageType`	`MT_GROUP_INFO`	
Campo	Tipo	Descrizione
`Group`	`Group`	Identificatore e direzione del gruppo.
`GroupStats`	`GroupStatistics`	Struttura dati contenente le informazioni di stato destinate alla RSU.
`IsDisposed`	booleano	Flag che indica se il gruppo è stato chiuso in seguito all'ultima iterazione.

in un ambiente molto "rumoroso" ovvero dove esistono molti altri nodi nelle vicinanze che partecipano in conversazioni all'interno di altri gruppi. Le immediate vicinanze di una qualsiasi RSU risultano essere delle aree particolarmente critiche non solo per il potenziale sovraffollamento dei veicoli ma anche per l'elevato numero di gruppi che questi comportano e la conseguente quantità di trame di dati che vengono continuamente scambiate, prevalentemente a causa del protocollo di raccolta dati utilizzato dai molteplici membri dei gruppi.

Per queste motivazioni, è di fondamentale importanza che ogni nodo adotti alcune strategie per tentare di minimizzare la possibilità di incorrere in interferenze tra pacchetti trasmessi simultaneamente e cercare comunque di mantenere adeguate performance per il protocollo ammettendo comunque la perdita di alcuni dati. Una prima semplice precauzione utilizzata è quella di evitare di rimuovere un nodo dalla lista dei membri conosciuti a seguito di una sola mancata risposta al probe del leader: l'attributo `MaxProbeFails` del modulo leader permette di tollerare il fatto che un nodo possa per qualche motivo non riuscire a rispondere a un certo numero di esplorazioni; se il membro in questione continua a non rispondere per un numero di iterazioni superiore al valore specificato, esso viene in ogni caso rimosso.

Una seconda importante strategia prevede di dimensionare in maniera dinamica il probe time, ovvero il tempo limite concesso all'insieme dei nodi di

un gruppo per rispondere a un segnale beacon trasmesso dal leader. L'idea è quella di concedere un intervallo di tempo tanto più ampio quanto maggiore è il numero di nodi presenti nel gruppo. A seguito della ricezione di un messaggio beacon proveniente dal leader, ciascuno di essi è infatti tenuto a scegliere un istante di tempo casuale con distribuzione uniforme tra zero e il valore specificato nel campo `Interval` dell'header pervenuto. Se il numero di mittenti è particolarmente numeroso, dilatare tale intervallo diminuisce la probabilità che i diversi pacchetti si sovrappongano cancellandosi a vicenda. Nel momento in cui un leader intende dare inizio a un'iterazione di probe, esso calcola valore per l'intervallo seguendo l'espressione sotto riportata:

$$Interval = ProbeMax - \frac{ProbeMax - ProbeMin}{(n+1)^{0.25}}$$

dove *ProbeMax* e *ProbeMin* indicano i rispettivi attributi di `BehaviourGroupLeader` che controllano il tempo massimo e minimo per il probe time e dove n corrisponde al numero di membri conosciuti al leader.

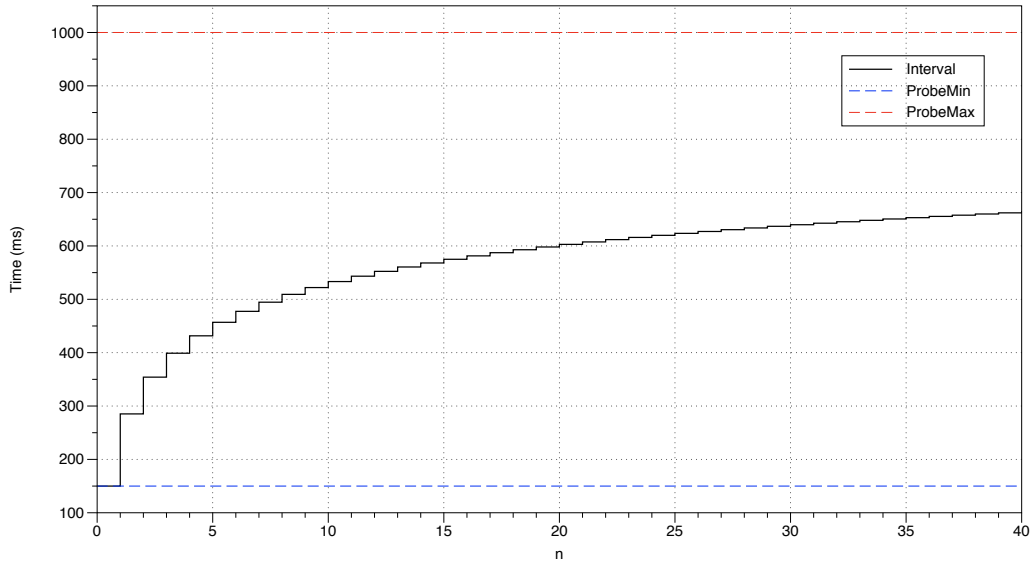

Figura 4.5: Dilatazione dell'intervallo di probe al crescere del numero di membri presenti in un gruppo (n). Si assumono dei valori di 150ms e 1000ms rispettivamente per gli attributi `TimeProbeMin` e `TimeProbeMax`.

La Figura 4.5 mostra l'andamento dell'intervallo di probe mano a mano

che i membri all'interno del gruppo crescono di numero. Il risultato si mantiene sempre asintoticamente lontano dal valore scelto come massimo in quanto si decide di prevedere un periodo di pausa tra due iterazioni di esplorazione consecutive.

Un ultimo accorgimento relativo alle temporizzazioni del protocollo riguarda l'intervallo di tempo che separa la fine di un'iterazione dall'inizio di quella successiva: per evitare che diverse procedure di esplorazione tendano a sovrapporsi si sceglie di applicare a tale intervallo uno skew scelto casualmente e delimitato in modulo dall'attributo `TimeProbeSkew`. Avendo precedentemente definito il valore *Interval* come l'intervallo di durata di una generica iterazione di esplorazione, il periodo di tempo durante il quale sia il leader del gruppo sia i relativi membri attenderanno l'inizio di quella successiva è ottenuto mediante la semplice espressione seguente:

$$Wait = ProbeMax - Interval \pm rnd$$

nella quale *rnd* indica un valore casuale di skew scelto mediante distribuzione uniforme tra zero e `TimeProbeSkew`.

Dal punto di vista dei singoli membri del gruppo la procedura di esplorazione si svolge in maniera elementare: essi provvedono a rispondere al messaggio di probe inviato dal loro leader con un messaggio beacon[6] contenente le informazioni di stato più recenti in loro possesso. Come è già stato accennato, ogni nodo risponde scegliendo un istante di invio casuale uniformemente distribuito all'interno dell'intervallo temporale specificato dal gestore. La risposta può avvenire soltanto se il nodo ha verificato di possedere una direzione ancora conformante con quella prevista per il gruppo a cui appartiene. Nel caso il veicolo inizi a seguire una rotta inappropriata esso si disattiverà autonomamente, evitando di rispondere al proprio leader.

Ogni volta che viene ricevuto un messaggio di probe proveniente dal proprio leader, ciascun membro del gruppo innesca un timer di durata configurabile via attributo `LeaderTimeout`; se allo scadere del tempo fissato non

[6]Si sceglie di inviare il messaggio beacon di risposta in modalità broadcast per fare in modo che anche i membri del gruppo possano costruire senza costi aggiuntivi una mappa parziale degli altri loro pari.

Tabella 4.13: Attributi previsti per il modulo `BehaviourGroupLeader`.

Attributo	Tipo	Default	Descrizione
`Enabled`	booleano	True	Determina se il comportamento è attivo o meno durante la simulazione.
`MergeGroup`	booleano	True	Stabilisce se al leader è permesso unirsi (in qualità di membro) a un altro gruppo se il proprio risulta vuoto.
`TimeProbeMax`	intero	1000	Periodo massimo (ms) per la durata della procedura di esplorazione dei membri.
`TimeProbeMin`	intero	150	Periodo minimo (ms) per la durata della procedura di esplorazione dei membri.
`TimeProbeSkew`	intero	50	Valore massimo (ms) che viene aggiunto o sottratto come *skew* al tempo di attesa tra due esplorazioni consecutive.
`MaxProbeFails`	intero	1	Massimo numero di iterazioni di esplorazione che un nodo membro può fallire prima di venire espulso dal gruppo.

sono pervenute ulteriori comunicazioni dal gestore del gruppo, esso considera persa la comunicazione col proprio leader e ritorna a uno stato di completa inattività.

Tabella 4.14: Attributi previsti per il modulo `BehaviourGroupMember`.

Attributo	Tipo	Default	Descrizione
`Enabled`	booleano	True	Determina se il comportamento è attivo o meno durante la simulazione.
`QuickDispose`	booleano	True	Stabilisce se al nodo è permesso abbandonare il gruppo se rileva che la distanza dall'RSU di riferimento è in aumento.
`LeaderTimeout`	intero	2000	Periodo di timeout (ms) per il leader del gruppo.

Lo schema di sequenza in Figura 4.6 completa la trattazione del protocollo di esplorazione dei membri illustrando un esempio nel quale tre nodi appartenenti allo stesso gruppo comunicano tra loro durante due iterazioni di probe consecutive. Al termine della prima il leader del gruppo non vede l'RSU all'interno del proprio range e pertanto conclude la procedura senza

ulteriori segnali. Nel corso della seconda iterazione l'RSU viene avvistata a seguito della ricezione di un suo beacon; trovando l'RSU di riferimento all'interno del proprio raggio operativo, il leader provvede a consegnare ad essa le informazioni appena ottenute.

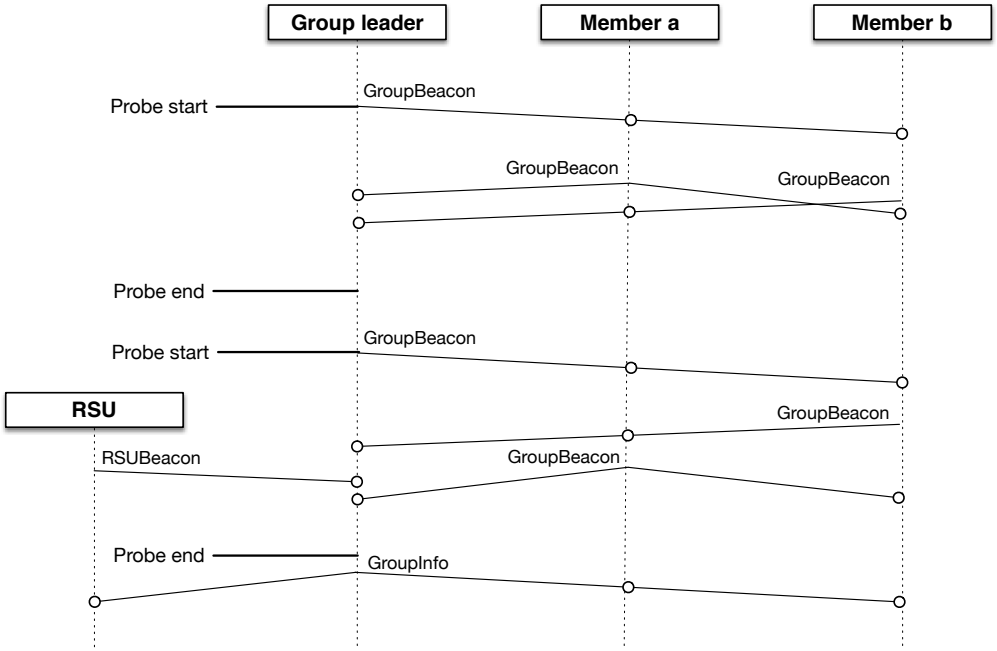

Figura 4.6: Diagramma di sequenza dei messaggi scambiati tra i nodi membri di un gruppo durante due procedure di esplorazione consecutive. Al termine della seconda iterazione, il leader provvede all'invio del messaggio di informazioni in quanto l'RSU di riferimento entra all'interno del suo range.

Le informazioni che ogni RSU riceve dai gruppi a cui fa riferimento sono rappresentate dalla struttura dati `GroupStatistics` e includono:

- Numero dei membri del gruppo
- Velocità media dei nodi (m/s)
- Velocità massima raggiunta all'interno del gruppo (m/s)
- Dimensione del diametro del gruppo (m)
- Densità conformante media percepita dai nodi

Il *diametro* di un gruppo consiste in una stima dell'estensione di un plotone di veicoli in un dato momento ed è definito come la distanza che separa il

membro più lontano dalla RSU di riferimento dal nodo più vicino; nei gruppi composti da un solo nodo il diametro assume sempre il valore zero.

4.3.2 Terminazione di un gruppo

Esistono due modalità con le quali il gruppo può cessare la propria attività: la prima viene detta *terminazione* (dispose) ed è il normale metodo con cui un gruppo conclude il proprio ciclo vitale quando i suoi servizi non sono più necessari; la seconda è sintomo di una qualche anomalia e si verifica quando – a causa di molteplici motivi – esso arriva a perdere tutti i suoi componenti. Esistono infatti diverse condizioni (discusse nel paragrafo 4.3.4) che possono portare un qualsiasi membro di un gruppo a distaccarsi da esso e a tornare inattivo.

In calce al paragrafo 2.2.3 è stata discussa la condizione che porta un qualsiasi gruppo a cessare la propria attività: nel momento in cui un gruppo è sufficientemente vicino alla RSU di riferimento i diversi veicoli inizieranno – se è loro possibile – ad abbandonare la direzione sulla quale il gruppo è stato fondato e percorreranno strade diverse, dividendosi. In queste situazioni non è possibile ottenere ulteriori informazioni dai componenti del plotone ed esso deve essere pertanto terminato.

Normalmente non è semplice assegnare una posizione precisa ad un gruppo, considerando che esso si compone di veicoli ciascuno dotato di una sua posizione e di un proprio pattern di mobilità. Esistono comunque diversi punti notevoli che possono essere utilizzati per comprendere con la giusta approssimazione la collocazione e la disposizione di un plotone; essi vengono illustrati nella Figura 4.7 e consistono nel nodo più vicino alla RSU, quello più lontano e quello leader. I primi due tra questi vengono chiamati punti di *frontiera* e vengono utilizzati per ricavare il diametro del gruppo.

Esistono buoni motivi per non considerare i nodi di frontiera come riferimento per la chiusura del gruppo. Se si terminasse un plotone non appena il nodo frontale (ovvero quello più vicino alla RSU) supera una soglia preimpostata si rischierebbe di cessare l'attività del gruppo troppo prematuramente; i membri risulterebbero infatti ancora relativamente lontani dalla RSU e pos-

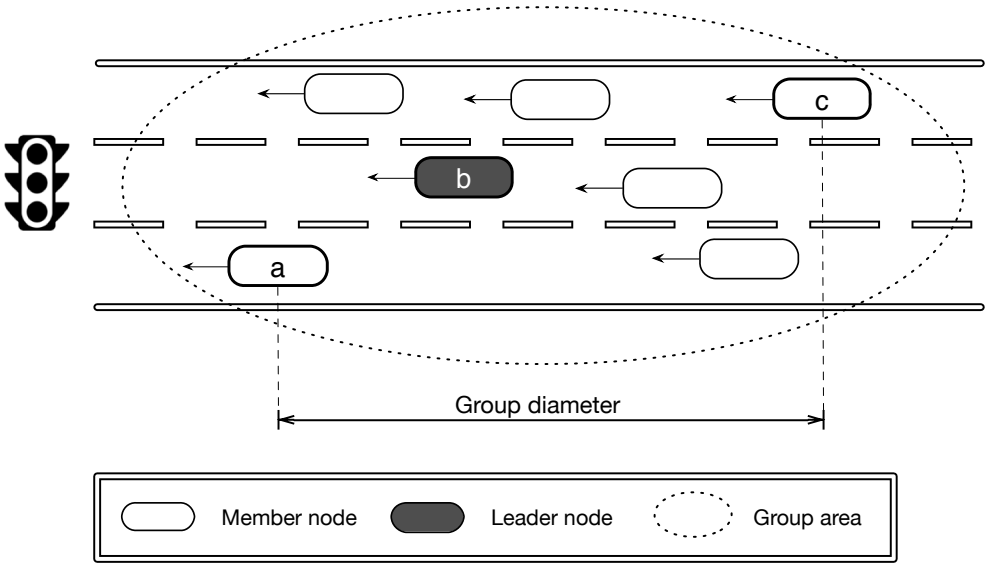

Figura 4.7: Posizioni notevoli all'interno di un gruppo: i nodi `a` e `c` individuano rispettivamente il punto più vicino e lontano del gruppo dalla RSU; dalla differenza di tali posizioni si ricava il diametro del gruppo. La posizione del nodo leader `b` viene utilizzata come punto di riferimento per la chiusura del gruppo.

sono contribuire ancora alla raccolta di informazioni utili per essa. Viceversa, chiudendo un gruppo nel momento in cui il suo ultimo nodo (quello più lontano) supera la soglia, si avrebbe che i restanti nodi potrebbero avere già superato l'incrocio e avere abbandonato il gruppo, specialmente il gestore.

Avendo progettato le diverse logiche di formazione in modo che esse eleggano al ruolo di leader quel nodo che risulta essere approssimativamente al centro del gruppo stesso, si è deciso di semplificare il problema e di considerare la posizione del leader come riferimento per determinare il momento in cui l'attività del plotone deve cessare. Il processo di chiusura di un gruppo viene quindi avviato dal suo leader il quale, se rileva che la propria posizione ha superato la soglia di valore configurabile attraverso l'attributo `SinkThreshold` previsto per il componente `ItsGroupController`, invierà uno speciale messaggio di informazioni (vedi Tabella 4.12) nel quale il flag `IsDisposed` risulterà abilitato. Siccome i pacchetti group info vengono sempre inviati come broadcast, sia l'RSU sia i membri del plotone capiscono che il gruppo ha appena completato il suo ultimo probe e ha cessato la propria

attività.

È infine previsto un semplice messaggio MT_GROUP_DISPOSE per forzare manualmente e in maniera del tutto asincrona[7] la terminazione di un gruppo. Attualmente non viene utilizzato da nessuna delle logiche realizzate e la sua implementazione è lasciata per eventuali estensioni del sistema. Se il leader di un gruppo riceve tale messaggio provvede a ritrasmetterlo ai propri membri per assicurarsi che esso venga ricevuto da tutti i componenti.

4.3.3 Annessione a un gruppo

A causa dell'elevata mobilità dei nodi nel corso della simulazione, è necessario prevedere delle tecniche aggiuntive mediante le quali i nodi possono entrare a fare parte di un gruppo quando questo è già formato e attivo. Il comportamento che si occupa di questa funzionalità è definito dal modulo `BehaviourGroupJoin` e il suo funzionamento risulta estremamente semplice.

Il componente di group join svolge la sua funzione nei momenti in cui un nodo è in stato di inattività e viene automaticamente disabilitato quando questi entra a fare parte di un plotone; la sua attivazione e spegnimento viene gestita in maniera trasparente dal controllore del gruppo. Durante i periodi in cui il comportamento è in funzione esso si limita a monitorare eventuali messaggi in ingresso appartenenti al protocollo PID_GROUP_MANAGEMENT alla ricerca di messaggi di beacon provenienti da un gruppo che abbia una direzione di riferimento conformante alla propria.

In questo modo, se un nodo inattivo capta un segnale di probe emesso da un leader[8] il cui gruppo si sposta seguendo una direzione compatibile, esso potrà inserirsi in tale plotone in qualità di membro. Per comunicare la propria annessione al gruppo il componente di join – quando vengono soddisfatte le condizioni appena esposte – provvede a rispondere al probe inviando un segnale di beacon con i propri dati, seguendo quindi lo stesso

[7] La normale terminazione di un gruppo operata dal leader a seguito dell'avvicinamento all'RSU avviene soltanto come atto conclusivo di un'operazione di probe; in questo senso essa può essere intesa come sincrona.

[8] I segnali di beacon provenienti dai membri vengono ignorati: per la definizione al par. 2.2.1 i nodi membri devono poter comunicare con il leader del gruppo.

comportamento messo in atto dai membri già presenti in un gruppo durante la procedura di esplorazione.

Il ritardo con cui il componente risponde ad un probe compatibile è scelto casualmente tra un intervallo minimo e massimo definito dagli attributi `JoinDelayMin` e `JoinDelayMax`.

Tabella 4.15: Attributi previsti per il modulo `BehaviourGroupJoin`.

Attributo	Tipo	Default	Descrizione
`Enabled`	booleano	True	Determina se il comportamento è attivo o meno durante la simulazione.
`JoinDelayMin`	intero	15	Intervallo minimo (ms) di attesa prima di unirsi a un gruppo compatibile.
`JoinDelayMax`	intero	80	Intervallo massimo (ms) di attesa prima di unirsi a un gruppo compatibile.

In maniera del tutto analoga alla procedura appena descritta, un leader di un gruppo *vuoto* (ovvero costituito dal solo gestore) può rinunciare al proprio ruolo e unirsi al leader di un gruppo percepito all'interno del proprio range. Questa funzionalità è particolarmente utile per evitare che in alcune particolari condizioni vengano a crearsi tanti gruppi vuoti per una direzione di marcia. Per evitare che due nodi leader possano decidere simultaneamente di cambiare gruppo e annullarsi a vicenda, si stabilisce un criterio in base al quale, data una coppia di leader ciascuno appartenente a un gruppo vuoto, esiste una sola configurazione risultante possibile come esito di una loro unione: un nodo gestore può subentrare in un gruppo esterno compatibile solo se il relativo leader possiede un identificativo (`NodeId`) maggiore del suo. La tecnica appena presentata è chiamata *group merging* e viene gestita dal componente `BehaviourGroupLeader`; la sua messa in atto è sempre condizionata dal valore dell'attributo `MergeGroup`.

4.3.4 Uscita da un gruppo

Un nodo attivo può uscire dal gruppo a cui appartiene quando questo viene chiuso per azione diretta del leader oppure quando non sono più verificate le condizioni di appartenenza al gruppo definite nel paragrafo 2.2.1 e

seguenti. Volendole riassumere, esse stabiliscono che un nodo può fare parte di un gruppo soltanto se esso è in grado di comunicare direttamente con il proprio leader e se la sua marcia mantiene una direzione conforme a quella prevista per il gruppo. Quando anche solo uno di questi due vincoli viene a mancare, il nodo deve necessariamente uscire dal gruppo. Si noti che il nodo responsabile del gruppo vede sempre soddisfatta la prima delle due condizioni.

Nell'eventualità che un nodo si trovi costretto a lasciare un gruppo si preferisce procedere a una sua uscita "silenziosa", cioè si evita la necessità di inviare un apposito messaggio per segnalare al leader la sua dipartita. Un veicolo che intende abbandonare un gruppo torna semplicemente a uno stato di inattività; dopo pochi secondi il leader provvederà in ogni caso a rimuoverlo dalla propria nodemap, a seguito dei ripetuti fallimenti riscontrati nella procedura di probe. Questa scelta comporta purtroppo una minore reattività del leader nel rilevare della perdita dei suoi membri ma è motivata dall'esigenza di ridurre il più possibile il traffico di messaggi tra i diversi componenti di un gruppo, sopratutto quando questi si trovano ad operare in aree molto dense di nodi. Se si ha l'esigenza di segnalare attivamente la dipartita di un veicolo da un gruppo è previsto l'utilizzo del semplice messaggio MT_GROUP_LEAVE il quale segnala ai riceventi l'intenzione del mittente di abbandonare il gruppo indicato.

Per coordinare meglio il comportamento dei veicoli, si stabilisce un ben preciso momento nel quale essi devono verificare la loro conformità alla direzione del gruppo. Tale istante viene fatto coincidere con l'inizio di un'itera-

Tabella 4.16: Formato dei messaggi di group handoff.

Campo		Valore
`ProtocolId`		`PID_GROUP_MANAGEMENT`
`MessageType`		`MT_GROUP_HANDOFF`
Campo	Tipo	Descrizione
`Group`	`Group`	Identificatore e direzione del gruppo.
`NodeMap`	`NodeMap`	Struttura dati contenente la mappa dei nodi del leader uscente.

zione della procedura di esplorazione dei nodi. Come è già stato accennato, ogni membro effettua un test sulla propria direzione alla ricezione del messaggio di probe inviato dal proprio leader e in caso di fallimento si disattiva e non risponde all'appello. Il gestore del plotone verifica invece la propria ammissibilità prima di inviare tale messaggio.

L'abbandono del gruppo da parte del leader è un evento piuttosto drammatico in quanto comporta la perdita dell'intero gruppo e di tutte le informazioni in esso raccolte. Per rispondere a questo problema viene implementato un protocollo di *handoff* mediante il quale il leader – prima di cessare definitivamente la propria attività – cede il controllo del gruppo ad un nodo membro scelto in base al suo valore di densità. Tale procedura si articola in due semplici passaggi e ha inizio con l'invio da parte del leader uscente di un messaggio di handoff destinato esclusivamente al nodo da lui scelto come successore. Compiuto questo primo passo, il mittente si disattiva e abbandona il gruppo senza attendere una conferma dal proprio potenziale successore per non aumentare eccessivamente il costo del protocollo.

Tabella 4.17: Formato dei messaggi di group switch.

Campo		Valore
`ProtocolId`		`PID_GROUP_MANAGEMENT`
`MessageType`		`MT_GROUP_HANDOFF`
Campo	Tipo	Descrizione
`Group`	`Group`	Identificatore e direzione del gruppo originale.
`NewGid`	`GroupId`	Nuovo identificatore per il gruppo.

Un messaggio di handoff contiene l'intera nodemap costruita fino a quel momento dal gestore originale e permette in questo modo al successore di riprendere la guida del gruppo senza perdite di dati. Per come è stato strutturato il sistema[9] , il cambiamento di un leader comporta necessariamente la modifica del GID del gruppo per fare sì che i membri e l'RSU di riferimento riconoscano il nuovo leader come tale. Per informare i nodi di un gruppo della necessità di effettuare il passaggio a un nuovo leader, il nuovo gestore del

[9]Vedi definizione di `GroupId` al paragrafo 4.1.

gruppo completa l'handoff con un messaggio di group switch il cui formato è riportato nella Tabella 4.17.

4.4 Protocollo Marker

Nel secondo capitolo è stata introdotta una particolare tipologia di nodi detti *marker* finalizzati ad estendere le funzionalità delle singole RSU anche in zone dell'ambiente lontane dalla loro naturale portata. L'idea alla base del loro semplice funzionamento consiste nel "rimbalzare" i segnali di beacon emessi da queste ultime in modo da riuscire ad attivare veicoli che si trovano a una considerevole distanza dalla RSU e avviare quindi con largo anticipo i protocolli di formazione dei gruppi.

Questa funzionalità è implementata dal componente `BehaviourMarker` in maniera del tutto trasparente alle altre logiche esposte in questo capitolo; un nodo può cioè assumere l'incarico di marcatore indipendentemente dagli altri ruoli che può avere già ottenuto durante la partecipazione agli altri protocolli.

Se un nodo guadagna il ruolo di marker a seguito della ricezione di un beacon da una RSU, esso controllerà quale tra le direzioni in esso riportate risulta conformante col proprio senso di marcia e assume tale valore come riferimento; tutti i segnali che ritrasmetterà a partire da quel momento includeranno esclusivamente tale valore come direzione consentita per la formazione dei gruppi. In questo modo i vari marcatori estendono la richiesta di group formation di una RSU esponendo soltanto la direzione che stanno percorrendo.

Nel corso del loro periodo di attività i nodi marker si limitano ad inviare un'unica tipologia di messaggio il cui formato è descritto nella Tabella 4.18. Questi pacchetti forniscono sostanzialmente lo stesso contenuto informativo dei beacon inviati dalle singole RSU (vedi la Tabella 4.1) ma permettono di essere ricevuti e processati unicamente dal componente `BehaviourMarker` prima di venire inoltrati alle altre logiche.

Indipendentemente dal fatto di avere assunto o meno il ruolo di marcatore, ogni volta che il componente marker di un nodo riceve un beacon inerente al proprio PID, esso produce un messaggio equivalente nel formato

Tabella 4.18: Formato dei messaggi beacon inviati dai nodi marker.

Campo	Valore	
`ProtocolId`	`PID_GROUP_MARKER`	
`MessageType`	`MT_MARKER_BEACON`	
Campo	Tipo	Descrizione
`SinkId`	`NodeId`	Identificatore della RSU di riferimento.
`SinkDirection`	double	Direzione di riferimento per la formazione dei gruppi.
`SinkPosition`	`Vector2D`	Posizione della RSU di riferimento.

MT_GROUP_SETUP_REQ previsto dal protocollo di group setup e lo spedisce via loopback locale al nodo alle logiche ad esso interessate. In questo modo, l'effetto di un messaggio spedito da un nodo marker risulterà del tutto analogo a quello ottenuto dall'invio di un beacon prodotto dalla RSU a cui esso fa riferimento.

4.4.1 Posizionamento dei nodi marker

Per scongiurare un'eccessiva quanto innecessaria proliferazione di nodi marcatori, nel paragrafo 2.4 sono state fornite tre condizioni affinché un qualsiasi nodo possa assumere il ruolo di marker e mantenerlo nel tempo:

1. La distanza di un nodo marker dalla propria RSU di riferimento deve essere compresa tra una soglia minima e una massima, rappresentate rispettivamente dagli attributi `SinkDistanceMin` e `SinkDistanceMax`.
2. La distanza reciproca tra due qualsiasi nodi marker deve essere superiore al valore controllato dall'attributo `MarkerDistanceMin`.
3. La direzione di un nodo marker deve essere conformante a quella originariamente fornita dalla RSU di riferimento.

Ogni volta che viene ricevuto un messaggio MT_GROUP_SETUP_REQ, sia esso proveniente da una RSU o da un veicolo marcatore, ciascun nodo verifica se queste condizioni sono soddisfatte: in caso affermativo sarà possibile

assumere (o mantenere) il ruolo di marcatore, se l'esito del test è negativo esso rinuncia a tale incarico e resta (o torna) inattivo.

La prima delle condizioni sopra elencate risulta molto semplice da verificare dal momento che ogni marker è sempre a conoscenza della posizione in cui si trova la propria RSU di riferimento. Per poter verificare la soddisfacibilità del secondo vincolo è necessario che ogni modulo mantenga una semplice mappa locale delle identità e delle posizioni degli altri nodi marcatori nelle proprie vicinanze, anche se esso attualmente non riveste il ruolo di marker. Per verificare la seconda condizione sarà quindi sufficiente scorrere ciascuna entry della lista, ricavare la distanza che separa i due nodi e confrontarla con il valore impostato per l'attributo `MarkerDistanceMin`.

Qualora una generica coppia di nodi attivi come marker scopra di non riuscire a soddisfare il secondo requisito è necessario disporre di un criterio per stabilire in maniera univoca chi dei due deve cedere il proprio ruolo, evitando così che entrambi si annullino vicendevolmente. Nell'implementazione realizzata si è scelto di mantenere attivo il marker che al momento del test risulta più distante dalla RSU.

Tabella 4.19: Attributi previsti per il modulo `BehaviourMarker`.

Attributo	Tipo	Default	Descrizione
`Enabled`	booleano	False	Determina se il comportamento è attivo o meno.
`TimeBeaconMin`	intero	1000	Periodo minimo (ms) di attesa tra invii successivi di messaggi beacon.
`TimeBeaconMax`	intero	2000	Periodo massimo (ms) di attesa tra invii successivi di messaggi beacon.
`SinkDistanceMin`	intero	50	Distanza minima (m) da mantenere tra il marker e l'RSU di riferimento.
`SinkDistanceMax`	intero	500	Distanza massima (m) da mantenere tra il marker e l'RSU di riferimento.
`MarkerDistanceMin`	intero	2000	Distanza minima (m) da mantenere tra il marker e un qualsiasi altro nodo marcatore.

Nel corso di questo capitolo sono state dunque presentate e discusse nel dettaglio quelle logiche (dette anche comportamenti) al livello più alto dell'architettura che realizzano concretamente i protocolli che ogni nodo dell'ambiente utilizza per interagire con altri suoi pari, per raggrupparsi e per raccogliere le informazioni di interesse per l'utente finale.

Le logiche realizzate si possono organizzare in quattro diverse aree di competenza: quella necessaria per la formazione dei gruppi di veicoli (`BehaviourGroupSetupReactive` e `BehaviourGroupSetupProactive`), quella che gestisce l'intero ciclo vitale di un gruppo e organizza la raccolta dei dati dai suoi nodi membri (`BehaviourGroupLeader`, `BehaviourGroupMember`, `BehaviourGroupJoin`), quella che raccoglie informazioni di stato sui nodi circostanti (`BehaviourNodeMap`) e infine quella per la propagazione dei segnali delle RSU (`BehaviourMarker`).

Nel capitolo successivo verranno proposti i risultati di alcuni test condotti al fine di verificare il corretto comportamento del sistema e delle logiche implementate, in modo da poterne studiare la risposta a fronte di alcuni scenari urbani realistici.

Capitolo 5

Risultati sperimentali

In questo capitolo verranno presentati gli scenari urbani utilizzati per sperimentare la risposta dell'architettura descritta fino a questo momento e saranno di seguito riportati e commentati i risultati dei test maggiormente significativi che sono stati svolti su di essi in modo da poter valutare il comportamento dei protocolli realizzati e la qualità dei dati prodotti da questi ultimi.

Si cercherà anzitutto di verificare che il sistema risponda adeguatamente agli stimoli offerti dai diversi ambienti predisposti. Verranno a tale scopo commentati alcuni studi condotti per individuare sotto quali limiti e condizioni è possibile avvalersi delle informazioni generate dai singoli gruppi di veicoli e si cercherà di capire se queste sono sufficienti per sintetizzare un'euristica soddisfacente per la determinazione dello stato reale del flusso di traffico nelle varie aree di interesse.

Nell'esporre i diversi risultati conseguiti dalle numerose simulazioni eseguite si porrà particolare attenzione nel confrontare le differenze in termini di costi e prestazioni offerte edalle due strategie di formazione a lungo discusse nel paragrafo 4.1.

Prima di procedere con l'analisi dei risultati verranno brevemente descritte le procedure necessarie per predisporre e configurare l'esecuzione di una simulazione ns-3 che utilizzi l'insieme dei moduli ITS prodotti.

5.1 Configurazione delle simulazioni

Per eseguire un'istanza di simulazione del sistema realizzato occorre predisporre e configurare quattro diversi elementi che contribuiscono a definire complessivamente il comportamento, i parametri utilizzati, il numero e la mobilità dei veicoli e la topologia dell'ambiente che si vuole utilizzare.

Script di simulazione

Lo script di simulazione è un listato di istruzioni in C++ che informano l'infrastruttura ns-3 su quali moduli devono essere impiegati e costruiscono la topologia dell'ambiente virtuale, definendone le proprietà e i comportamenti da seguire. Per caricare ed eseguire il modulo ITS descritto in questo testo è sufficiente utilizzare il file di sorgenti `itstracetest.cc` incluso tra i *deliverables* realizzati.

Il software messo a disposizione costruisce un ambiente ns-3 servendosi della traccia di mobilità[1] specificata all'interno del file di configurazione ITS e installa l'insieme dei moduli dell'architettura realizzata in ciascuno dei nodi in essa presenti. Terminata la fase di set-up, lo script procede ad avviare la simulazione secondo i parametri impostati dall'utente. Questi ultimi vengono forniti in input al programma mediante linea di comando; la lista dei parametri previsti dallo script indicato è descritta nella Tabella 5.1 sottostante.

Tutti i parametri predisposti per la riga di comando sono opzionali, fatta eccezione per `configFile`: quest'ultimo infatti specifica il nome del file xml contenente tutte le informazioni indispensabili per poter procedere alla simulazione.

File di configurazione ITS

Il file di configurazione xml per lo script ITS costituisce un input fondamentale per un'istanza di simulazione ITS in quanto fornisce ad esso le seguenti informazioni:

[1] Si veda il paragrafo 3.1.2.

Tabella 5.1: Parametri di comando per lo script di simulazione.

Parametro	Tipo	Descrizione
`configFile`	stringa	Nome del file .xml di configurazione per il modulo ITS
`animFile`	stringa	Nome del file di output NetAnim per la produzione di un'animazione della simulazione
`nodes`	intero	Numero di nodi da utilizzare nella simulazione. Se non specificato vengono utilizzati tutti quelli descritti dalla traccia di mobilità in input
`simTime`	intero	Durata in secondi della simulazione. Se non specificato la durata corrisponde a quella indicata nel file di configurazione
`SetupMode`	intero	Modalità di formazione dei gruppi utilizzata. 0 = Reattiva (default), 1 = Proattiva
`logFile`	stringa	Nome del file .log di output contenente un resoconto dettagliato della simulazione

- un insieme di veicoli da simulare e la relativa traccia di mobilità
- un insieme di nodi RSU con le relative proprietà
- alcuni parametri avanzati per la simulazione
- le opzioni di configurazione per il campionamento dei dati in real-mode

La lista dei veicoli da utilizzare all'interno della simulazione viene delimitata dal tag xml `<vehicles>` e richiede un attributo obbligatorio `tracefile` che punti al file di mobilità da utilizzare. All'interno del tag è previsto un elenco di entità denominate `<node>` ciascuna delle quali rappresenta una vettura che si intende inserire nell'ambiente e ne specifica l'id del relativo nodo ns-3 e gli istanti di tempo in cui questo deve essere attivato e disattivato (attributi `id`, `start` e `stop`). È possibile specificare un numero inferiore di veicoli rispetto a quelli definiti all'interno della traccia di mobilità utilizzata.

Compilare manualmente la lista completa di tutti i nodi previsti in una simulazione è indubbiamente un'operazione estremamente tediosa, motivo per cui è stato predisposto uno script Python `traceconvert.py` per generare automaticamente un file di configurazione xml a partire da un output di simulazione SUMO.

Una traccia di mobilità specifica il comportamento dei soli veicoli, ogni singola RSU deve quindi essere definita all'interno del tag `<infrastructure>` per mezzo dell'elemento `<rsu>` e dei suoi attributi `id`, `xpos` e `ypos`. Una lista di elementi `<direction>` descrive la lista dei valori di direzione monitorati da ciascuna RSU.

Mediante il tag facoltativo `<configuration>` è possibile intervenire su tre ulteriori parametri di configurazione: il suo attributo `ns3configfile` permette di puntare a un file testuale per la specifica dei valori degli attributi ns-3 (vedi sotto); l'elemento `<stop>` specifica l'istante finale della simulazione ma può essere sovrascritto mediante l'apposito parametro via riga di comando; il tag `<classes>` definisce la composizione della popolazione in termini di classi di veicoli. Gli attributi `full` e `medium` impostano la probabilità per ogni veicolo di appartenere alle rispettive classi descritte al paragrafo 2.3.

Traccia di mobilità SUMO

Una traccia di mobilità è un file testuale di estensione .tcl che dichiara un insieme di nodi e descrive in ogni istante i valori di posizione e velocità assunti da ciascuno di essi (vedi paragrafo 3.1.2). Ogni traccia viene ottenuta esportando l'output di una simulazione SUMO per mezzo del tool di utilità `traceExporter` messo a disposizione dal pacchetto stesso [Kra+12] e specificando il formato in uscita ns-2 (compatibile con ns-3).

File di configurazione ns-3

Anche se l'esatta topologia dell'ambiente e i componenti ns-3 da utilizzare nei diversi nodi vengono specificati dallo script di simulazione, esiste una lunga lista di parametri di sistema e di attributi per i vari moduli che possono essere configurati a piacimento senza bisogno di ricompilare il pacchetto utilizzato.

Ns-3 mette a disposizione la funzionalità `ConfigStore` per consentire di caricare queste diverse impostazioni da un file di testo e applicarle all'interno di uno script prima che il motore di simulazione venga avviato [13e]. Le preferenze possono essere specificate per tutti i nodi che utilizzano un particolare

componente (comando `default`), oppure per uno specifico percorso ns-3 (comando `value`) in modo da limitarne l'applicazione a un ristretto sottoinsieme di entità.

Il modulo ITS sviluppato definisce tre insiemi di nodi contenuti all'interno del componente `ItsNodeContainer`: quelli relativi alle RSU, ai veicoli di classe full e di classe medium. In questo modo è possibile specificare tale componente nel percorso di configurazione e accedere a uno dei tre contenitori per applicare un valore a tutti i nodi appartenenti a tale classe.

È importante ricordare che in caso di sovrapposizioni, i valori degli attributi configurati attraverso il file di configurazione hanno maggiore priorità rispetto a quelli specificati per mezzo della riga di comando.

5.2 Scenari urbani utilizzati

Durante le fasi di implementazione e messa a punto dell'architettura sono stati utilizzati diversi scenari per osservare il funzionamento dei sorgenti prodotti: da semplici rettilinei o incroci costruiti ad-hoc verso ambienti più complessi e sofisticati. Per sperimentare correttamente il comportamento del sistema è necessario avvalersi di tracce di mobilità contenenti reti stradali complete e adeguatamente complesse per la simulazione realistica di ambienti urbani con elevato indice di densità.

La Figura 5.1 mostra i due scenari principalmente utilizzati per produrre i numerosi test presentati in questo capitolo: "Andrea Costa" e "Pasubio", i quali prendono i loro nomi dalle due strade della città di Bologna da cui sono stati estratti. Entrambi sono stati messi a disposizione rispettivamente dai progetti iTetris e COLOMBO (vedi capitolo 1) e sono stati prodotti a seguito di rilevazioni su flussi reali di traffico effettuate nelle corrispettive zone della città. Nella figura è possibile osservare le direzioni prese in esame[2] e le caratteristiche dei tratti stradali simulati. Nel complesso, entrambe le tracce di mobilità rappresentano un totale di 48 ore di traffico veicolare. Per ovvi motivi non è possibile presentare in maniera compatta e leggibile i

[2]Nello scenario A. Costa la direzione sud-ovest non viene monitorata in quanto la relativa traccia di mobilità non registra il passaggio di alcun veicolo per tale tratto stradale.

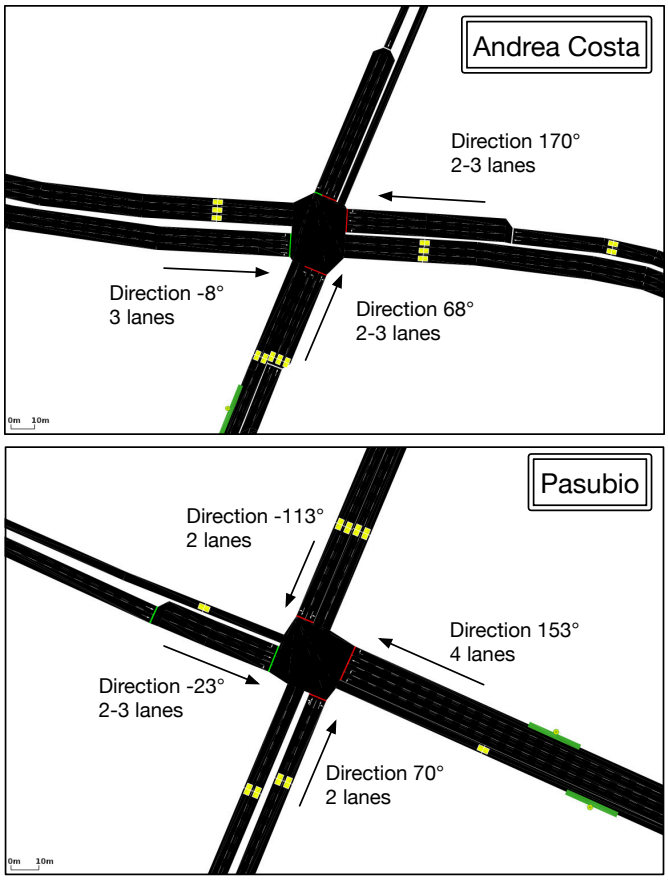

Figura 5.1: Rappresentazione topologica delle reti stradali Andrea Costa e Pasubio: i due principali scenari utilizzati per i test del sistema.

risultati derivati da una simile quantità di dati; nei successivi test vengono pertanto estratti degli intervalli temporali corrispondenti ai primi 30 minuti di simulazione da ciascuna delle due tracce utilizzate.

Le differenze principali tra le due tracce di mobilità utilizzate risiedono nella quantità di nodi simulati e nel numero di lane che costituiscono i tratti stradali che partono dall'incrocio in esame. Lo scenario A. Costa raggiunge un quantitativo di veicoli all'interno del raggio operativo della RSU che oscilla tra le 70 e le 110 unità; tra le direzioni monitorate, quella con valore -8° risulta la più trafficata e presenta frequenti congestioni. Lo scenario Pasubio presenta una densità decisamente elevata con un traffico a regime che varia da 80 a 140 veicoli al secondo e quattro direzioni caratterizzate

da un abbondante flusso di traffico; tra queste il tratto con direzione 153° è sicuramente degno di nota in quanto costituito da quattro corsie che presentano una congestione pressoché permanente all'interno dell'intervallo di simulazione considerato.

Nel corso della sperimentazione del progetto è stata frequentemente utilizzata a scopi di debug una traccia di mobilità più semplice ottenuta dal progetto COLOMBO e relativa a rilevazioni del traffico nei pressi di un incrocio della città di Norimberga. Tale scenario non presenta comunque elementi di interesse rispetto a quelli sopracitati e il traffico in essa rappresentato è decisamente meno intenso e poco impegnativo per il sistema. Per queste ragioni si è scelto di non utilizzare tale scenario nei test presentati nei paragrafi seguenti.

I parametri di sistema comunemente utilizzati per l'esecuzione dei test corrispondono ai valori predefiniti presentati nelle tabelle ai capitoli 3 e 4 che descrivono gli attributi dei vari componenti realizzati. Per quanto riguarda la configurazione dei numerosissimi parametri offerti dal pacchetto di simulazione ns-3 si è scelto per semplicità di adottare una scelta analoga a quella fornita di default dalla piattaforma iTetris, ovvero di utilizzare i valori predefiniti dallo standard 802.11b in modalità ad-hoc, con metodologia di trasmissione DSSS [Tan02] a 1 Mbps, ritardo di propagazione costante e modello logaritmico per la caduta del segnale. Come conseguenza delle precedenti impostazioni i nodi di classe full e le RSU hanno una portata approssimabile a 170 metri, mentre i nodi di classe media dispongono di circa 100 metri di range.

5.3 Risultati

Per verificare il corretto funzionamento del sistema occorre anzitutto accertarsi che esso sia in grado di fornire una stima apprezzabile della quantità di traffico proveniente dalle diverse direzioni monitorate. In particolare i dati forniti alla RSU in esame devono essere tali da permettere di risalire alla quantità di veicoli presenti in una data direzione e alla loro velocità media. In realtà è stato mostrato nel paragrafo 4.3.1 che l'architettura realizzata è

in grado di produrre anche altre informazioni relative alle densità medie percepite dai nodi del gruppo, alla velocità massima raggiunta e al diametro del gruppo. Tuttavia nel corso dei numerosi test effettuati e presentati in questo capitolo si è scelto di dare maggiore risalto ai dati riguardanti il numero di nodi rilevati e la loro velocità media essendo questi di maggiore criticità al fine di produrre una corretta comprensione della condizione del traffico che è necessario smaltire.

I test esposti nel seguito del presente capitolo si riferiscono ai risultati ottenuti utilizzando le configurazioni e gli scenari illustrati nel paragrafo 5.2. Siccome il sistema ricorre assai di frequente all'estrazione di numeri pseudo-casuali all'interno delle proprie logiche di protocollo è necessario predisporre per ogni diversa tipologia di test l'esecuzione di più iterazioni, fornendo allo script di simulazione gli stessi parametri ma un seme differente e procedendo in secondo luogo all'integrazione dei diversi risultati.

Ciascuno dei risultati di seguito esposti rappresenta quindi l'esito di un insieme di 30 iterazioni, ciascuna della durata di 30 minuti[3]. Normalmente, nel presentare i grafici riassuntivi, si sceglie di escludere i primi 100-200 secondi di simulazione per tagliare tutti i possibili effetti di bordo e presentare esclusivamente il comportamento del sistema a regime.

Nella maggior parte dei test svolti vengono messi a confronto i risultati ottenuti utilizzando una strategia di group formation di tipo reattiva e quelli ricavati utilizzando la controparte proattiva. Occorre infatti studiare e verificare se il ricorso a quest'ultimo protocollo comporta effettivamente un vantaggio in termini di qualità dei dati prodotti e capire eventualmente in quali circostanze esso si dimostra più vantaggioso rispetto alla strategia puramente reattiva.

5.3.1 Indicatori di prestazione

Per poter valutare correttamente le prestazioni della soluzione realizzata e svolgere i confronti tra i diversi scenari e protocolli è indispensabile disporre

[3]Nel seguito, se non esplicitamente espresso, tutti i valori temporali indicati si riferiscono al tempo di sistema simulato.

non solo dei risultati emessi dal sistema ma anche dei valori "ottimi" rispetto ai quali si sta tentando di ricavare una stima. Nel valutare i risultati relativi al numero di veicoli percepiti si confronteranno ad esempio i valori ottenuti dalla RSU di riferimento con il reale numero di veicoli presenti in quella direzione. I valori reali vengono estratti da una procedura detta *real-mode* che si occupa di campionare e registrare in ogni secondo tutti i dati di interesse per l'analisi in corso. Tale routine dispone ovviamente di una completa conoscenza dell'ambiente simulato e dello stato completo dei suoi partecipanti ma non influisce in alcun modo con il comportamento dell'architettura in esame (per questo motivo viene anche chiamata informalmente *god-mode*).

La procedura real-mode è programmata per monitorare lo stato dei veicoli che si trovano entro un raggio compreso tra 20 e 170 metri dalla RSU selezionata. Sotto i 20 metri i veicoli si avvicinano infatti al centro dell'incrocio e può risultare difficile distinguere l'esatta direzione a cui essi fanno riferimento; si ricorda inoltre che i gruppi sono impostati per terminare la loro attività superata tale distanza dalla RSU. La scelta di non tracciare i veicoli a distanze superiori a 170 metri dall'incrocio riflette il range massimo ottenibile dai veicoli di classe più elevata: oltre tale soglia essi non possono in ogni caso comunicare con l'RSU e non sarebbe quindi corretto includerli all'interno del conteggio.

I valori ottenuti mediante la modalità reale sono da considerarsi ideali soltanto in relazione al range monitorato e non in maniera assoluta: nonostante i nodi non riescano effettivamente a comunicare con l'RSU oltre il limite preimpostato, questi possono comunque fornire – una volta entrati nel range operativo – dei dati che comprendono il contributo di veicoli a distanza leggermente superiore alla soglia dei 170 metri. In simili casi la differenza tra i valori prelevati in real-mode e quelli ricavati dai protocolli è da intendersi a vantaggio del sistema, considerato che quest'ultimo riesce a coprire una distanza superiore al campo visivo della modalità reale.

Per contestualizzare meglio i grafici successivi si è scelto di riportare ove appropriato una curva che riporta la distanza del veicolo più lontano rilevato in ogni secondo dalla RSU di riferimento. Confrontando tale curva con il range costante usato in real-mode è possibile dare un'interpretazione più

significativa alle differenze tra i valori ideali e quelli effettivamente ottenuti.

Nello studiare e commentare il comportamento dell'architettura implementata, si vorrebbe poter estrarre da ogni simulazione un indice assoluto di performance che determini approssimativamente il livello di qualità dei dati raggiunto dal sistema in una particolare configurazione. In generale questo requisito non è semplice da soddisfare, considerando la quantità e l'eterogeneità delle informazioni prodotte e tenendo conto che è quasi impossibile disporre di valori ottimi con cui poterle confrontare. Per rispondere almeno in parte a questa esigenza è stato introdotto un indicatore denominato *delta* (Δ) il quale riporta una media della differenza tra il numero di nodi rilevati in ogni secondo dalla modalità reale e quelli individuati dal sistema per ciascuna direzione di interesse.

Data una qualsiasi delle direzioni monitorate, per il calcolo del relativo valore di delta viene utilizzata l'informazione sulla distanza del veicolo più lontano rilevato dalla RSU nella modalità seguente: in un generico secondo della simulazione, se tale valore è al di sotto del range in real-mode, la differenza tra i veicoli reali e quelli rilevati viene sommata in valore assoluto a un accumulatore, se invece i protocolli riescono a coprire un'area più estesa la differenza viene sommata mantenendone il segno. Al termine della simulazione, l'accumulatore viene diviso per il totale dei secondi simulati per ottenere il valore delta complessivo. È fondamentale sottolineare che le altre informazioni prodotte dai protocolli non vengono coinvolte nel calcolo di tale parametro ma risultano comunque utili per la comprensione delle reali condizioni del traffico, come verrà di seguito dimostrato nell'analisi dei risultati ottenuti.

In sintesi, il valore di delta relativo a una direzione misura lo scarto tra il numero dei veicoli rilevati dalla modalità reale e quelli percepiti dall'insieme dei protocolli realizzati. Un valore di delta positivo indica una prestazione inferiore rispetto a quella ottenibile mediante un ipotetico "radar" con visione perfetta tra le due soglie impostate. Il valore di delta decresce invece quando il sistema è in grado di rilevare dei nodi a distanza superiore rispetto al range utilizzato in real-mode.

I primi risultati sperimentali che verranno presentati riguardano una verifica della capacità del sistema di produrre un'euristica in grado di determinare le reali condizioni del traffico nei diversi scenari a disposizione; si cercherà inoltre di comprendere in quali modalità il fattore di penetrazione della tecnologia proposta incide sulla qualità del sistema stesso. Verrà infine condotto uno studio sull'effetto di differenti composizioni di popolazione al variare delle tre classi di veicoli enunciate nel paragrafo 2.3.

5.3.2 Comportamento del sistema

Il primo test presentato riguarda una verifica della capacità del sistema di stimare correttamente l'andamento delle caratteristiche del traffico in entrambi gli scenari utilizzati. Le figure 5.2 e 5.3 mostrano i risultati ottenuti rispettivamente nello scenario Andrea Costa e Pasubio. Ciascuno dei due grafici rappresenta l'andamento dei valori in due direzioni di interesse: le linee relative alle performance dei protocolli di grouping reattivi sono grafiche in rosso, quelle dei protocolli di tipo proattivo in blu. Le linee scure si riferiscono sempre ai valori ottenuti dalla procedura in real-mode.

A un primo impatto si nota immediatamente che i protocolli tendono in generale – specialmente nello scenario Pasubio – a sottostimare la reale quantità di nodi presenti nel tratto stradale, impressione che viene oltretutto confermata da valori di delta abbondantemente superiori allo zero. Questo è chiaramente del tutto prevedibile ed è imputabile a diversi fattori. Il più importante tra questi riguarda il fatto che l'attività di un gruppo viene sempre terminata nel momento in cui il leader raggiunge una distanza di 20 metri dalla RSU. Superato tale si ha un immediato crollo del numero di nodi rilevati dal sistema in quanto potrebbero venire disattivati anche dei nodi che si trovano ancora relativamente lontani dall'incrocio (e quindi ancora rilevati in real-mode).

Un altro fattore riguarda il rischio sempre presente che alcuni nodi escano anche solo temporaneamente da un gruppo a causa di un valore di direzione non conforme a quello previsto dalla RSU oppure per via di alcune iterazioni

di probe fallite a seguito di alcune perdite di pacchetti dovute a un elevato numero di collisioni.

Questi difetti possono in generale essere in parte corretti attraverso degli interventi al sistema finalizzati a perfezionare la gestione del ciclo di vita dei singoli gruppi ma possono anche essere sensibilmente attenuati da una logica aggiuntiva collocata internamente alle varie RSU. Compito di tale ipotetico modulo dovrebbe essere quello di raffinare i dati ottenuti direttamente dai singoli gruppi, effettuando una sorta di *smoothing* intelligente per eliminare variazioni improvvise o *glitch* e per migliorare le informazioni disponibili a seconda del loro contesto e soprattutto avendo a disposizione lo stato corrente dei segnali di traffico emessi dalla RSU. Si consideri ad esempio il caso in cui viene rilevata la chiusura di un gruppo dotato di una velocità media particolarmente bassa contestualmente all'emissione di un segnale di stop per quella direzione da parte dell'RSU. Una logica sufficientemente smart potrebbe facilmente intuire che in realtà i veicoli appartenenti al gruppo appena chiuso non possono avere oltrepassato l'incrocio, visto che la loro velocità era prossima allo zero e il semaforo stava segnalando ai veicoli di fermarsi. Viceversa, se la chiusura di un gruppo viene riscontrata quando le vetture hanno accesso all'incrocio è possibile rinforzare ancora per qualche secondo la loro presenza basandosi sulla velocità rilevata e sul tratto stradale che ancora devono percorrere prima di raggiungere l'RSU.

Nei due grafici inferiori posti in ciascuna delle figure 5.2 e 5.3 si tenta di simulare almeno in parte i risultati ottenibili mediante tale logica effettuando un'interpolazione dei dati ottenuti dalla RSU attraverso un polinomio di 90simo grado. Tale approssimazione è in ogni caso irrealistica in quanto non applica delle logiche basate sul contesto dei dati ricevuti e oltretutto sfrutta in ogni istante sia i campioni passati che quelli futuri. Essa consente comunque di osservare a colpo d'occhio quanto anche una semplice interpolazione riesca a migliorare la qualità (e la leggibilità) dei dati pervenuti.

Procedendo più concretamente all'analisi dei risultati si osserva innanzitutto l'andamento sinusoidale del numero dei veicoli presenti nelle diverse direzioni. Questo è chiaramente dovuto al fatto che in condizioni di traffico intenso i veicoli procedono dapprima ad accodarsi in attesa di poter accede-

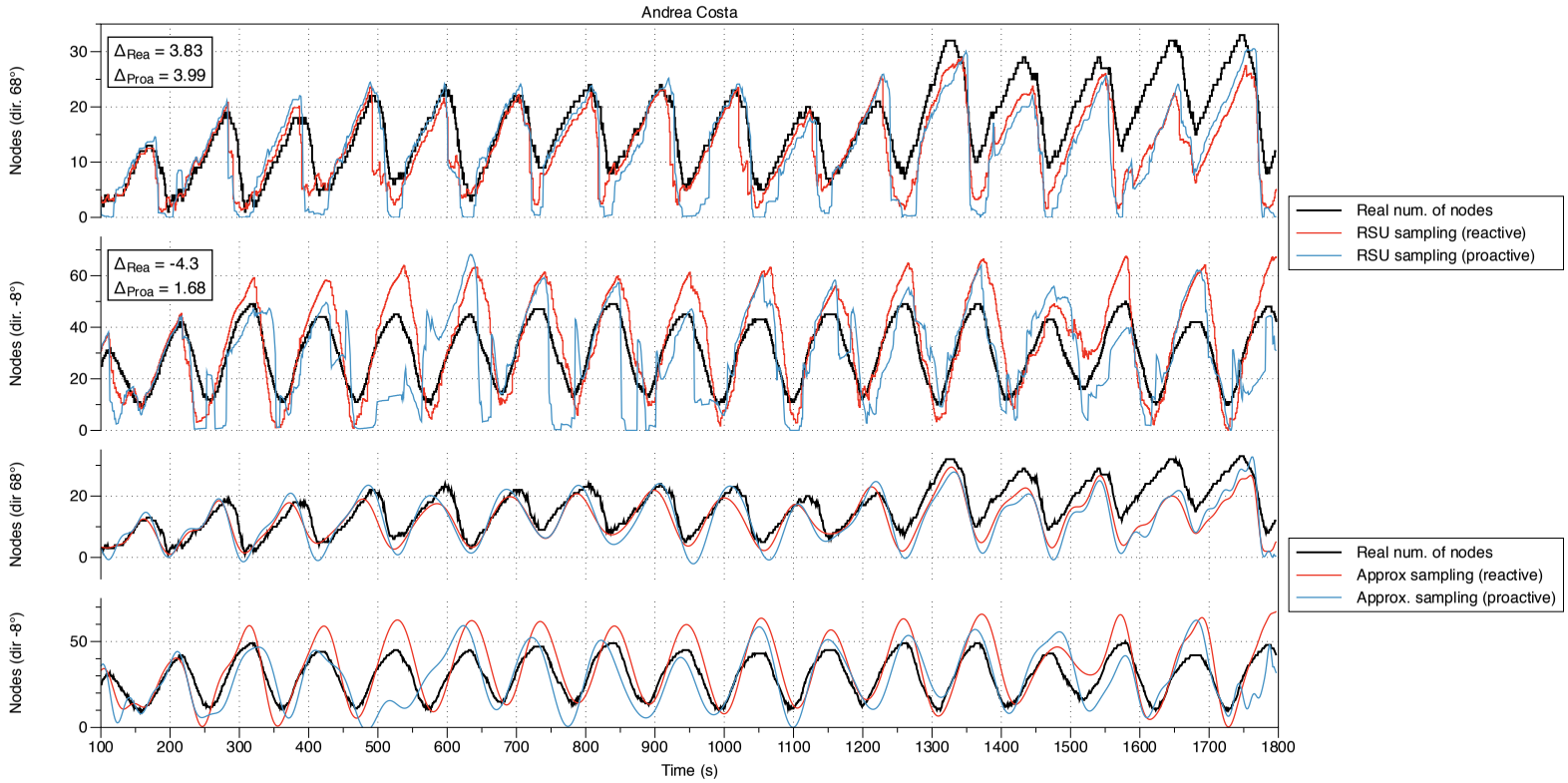

Figura 5.2: Andamento del numero di veicoli rilevati dai protocolli reattivi e proattivi nelle direzioni 68° e -8° dello scenario Andrea Costa. I due grafici inferiori mostrano l'effetto di un'interpolazione dei dati ottenuta con un polinomio di 90simo grado.

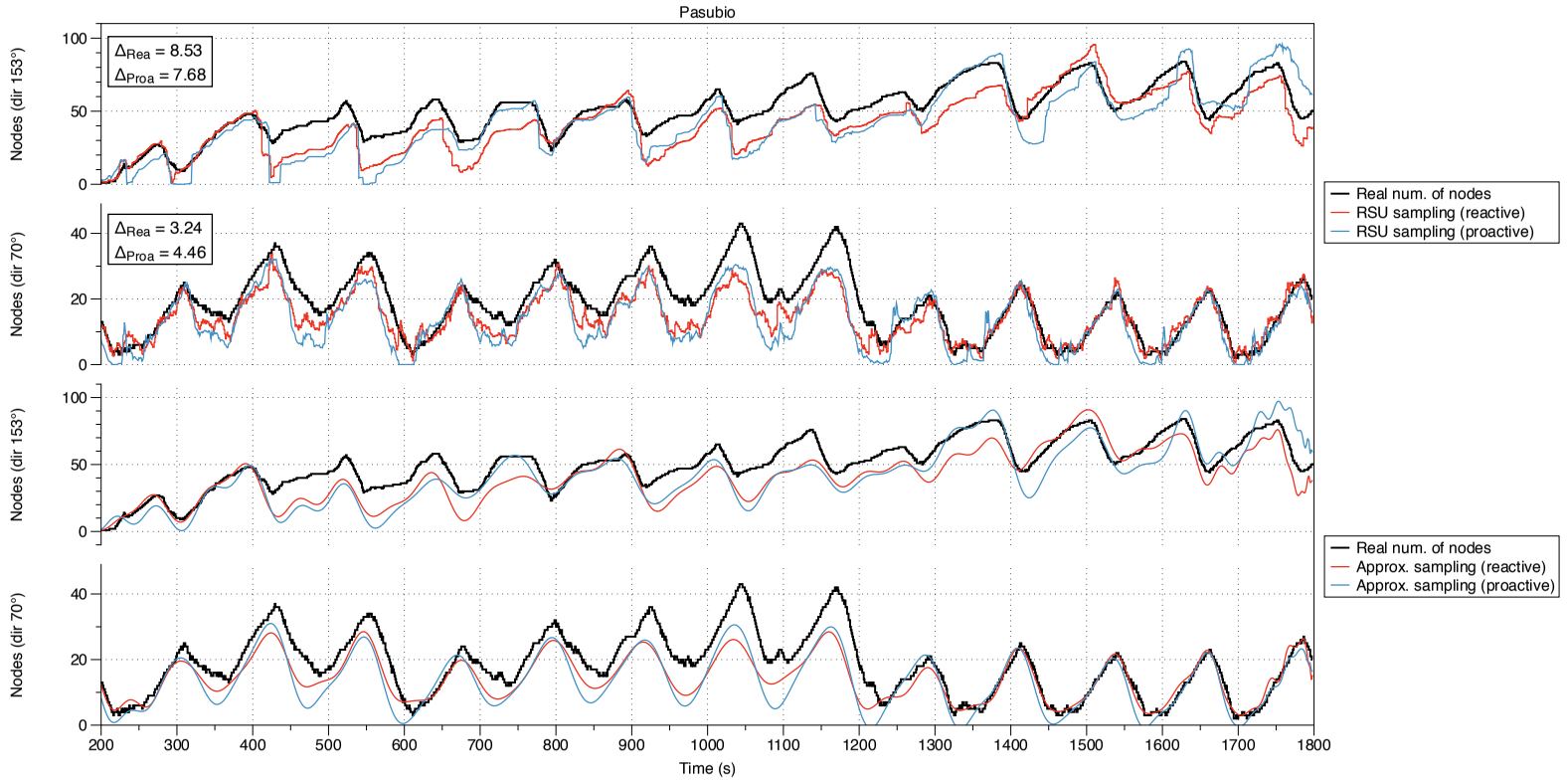

Figura 5.3: Andamento del numero di veicoli rilevati dai protocolli reattivi e proattivi nelle direzioni 153° e 70° dello scenario Pasubio. I due grafici inferiori mostrano l'effetto di un'interpolazione dei dati ottenuta con un polinomio di 90simo grado.

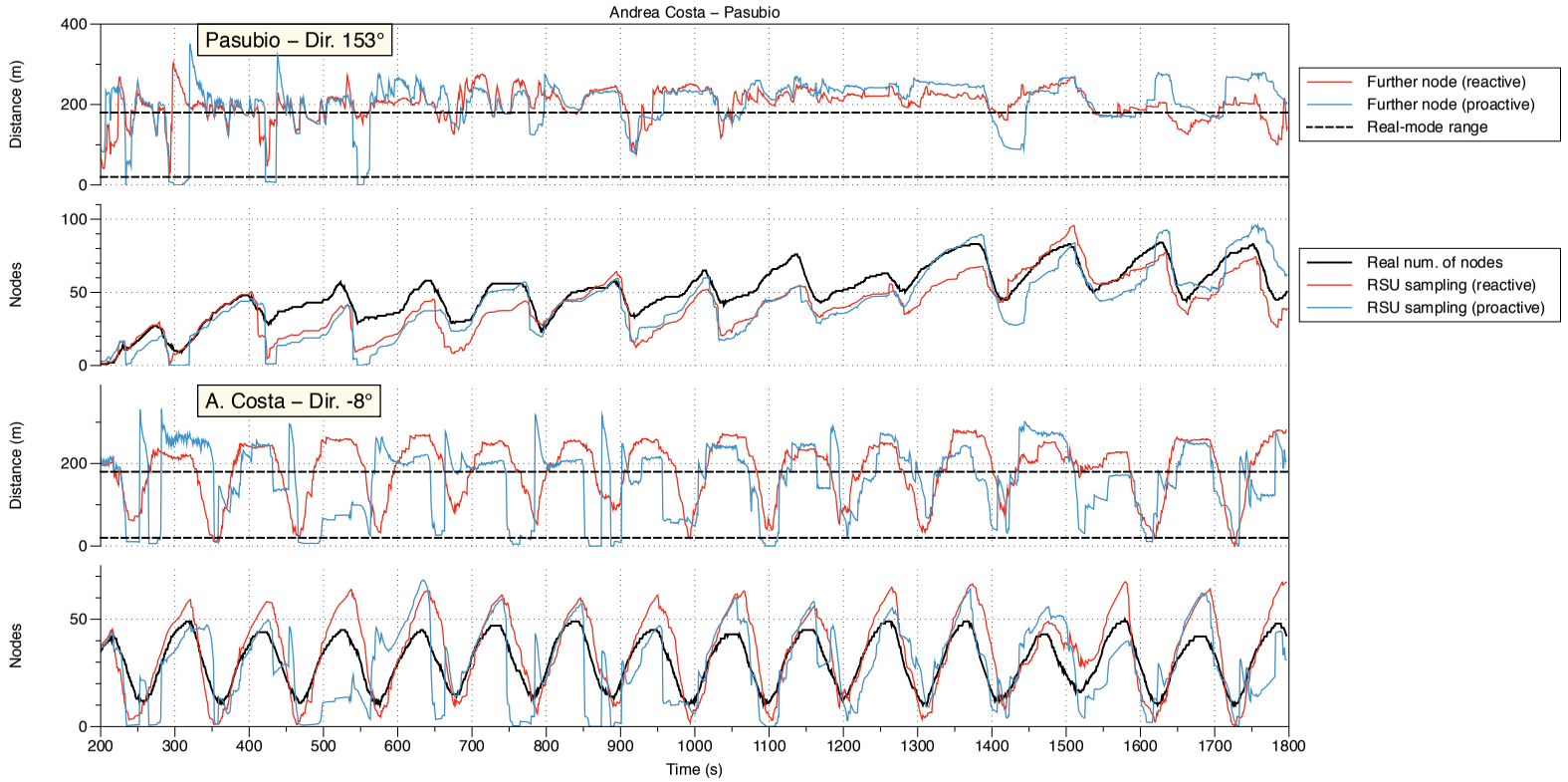

Figura 5.4: Andamento dei veicoli rilevati dai protocolli reattivi e proattivi nelle direzioni 153° dello scenario Pasubio e -8° di Andrea Costa. Sopra a ciascuna delle due rilevazioni viene graficato il valore di distanza relativo al veicolo più distante comunicato alla RSU.

re all'incrocio e in seguito superano l'RSU e abbandonano il tratto stradale monitorato non appena ricevono un segnale verde dal semaforo. Nel complesso i protocolli riescono a seguire piuttosto fedelmente il reale andamento del traffico, pur presentando frequenti sottostime e – meno frequentemente – sovrastime.

Le direzioni che raggiungono valori di densità maggiore negli scenari di Costa e Pasubio sono rispettivamente quelle di -8° e 153°; incidentalmente tali tratti stradali dispongono anche di un maggior numero di corsie per senso di marcia (rispettivamente 3 e 4). Queste direzioni sembrano essere maggiormente afflitte da imprecisioni e disturbi; in realtà i valori di delta indicati sembrano suggerire che nella direzione -8° di A. Costa il sistema riesce nel complesso a riportare una stima migliore rispetto alle rilevazioni ottenute mediante la modalità reale, cosa che però non accade nello scenario Pasubio.

Nel caso della direzione -8° si verificano frequenti picchi che eccedono il numero di veicoli rilevati in real-mode. In realtà è dimostrato in Figura 5.4 che tale sovrastima non è dovuta a degli errori del sistema ma è imputabile alle lunghe code che si presentano nel tratto stradale in esame: al crescere del numero di veicoli che si accodano in attesa di accedere all'incrocio, l'RSU riceve frequentemente dei riscontri da gruppi a distanza elevata i cui membri superano molto spesso i 200 metri di distanza, fornendo pertanto dei dati qualitativamente migliori (da cui il valore di delta negativo) rispetto a quelli ideali ma limitati a una visione di 180 metri.

Le imprecisioni riscontrate nel tratto di direzione 153° dello scenario Pasubio e il conseguente valore di delta sono invece sintomo di un certo livello di inaccuratezza raggiunto da entrambi i protocolli in condizioni di densità estremamente elevate, nonostante anche in questo caso la Figura 5.4 dimostri la capacità dei protocolli di lavorare su un'area più ampia rispetto a quella impostata in real-mode. Le imprecisioni non sono tali da inficiare in maniera critica il funzionamento complessivo del sistema ma in alcuni casi, specialmente quando la densità raggiunge valori di circa 90 veicoli, si assiste a delle sottostime superiori alle 20 unità. Il fatto che tale tratto stradale sia oltretutto quello che dispone del più alto numero di corsie (quattro) tra

tutte le simulazioni suggerisce che l'elevato numero di veicoli che si trovano a comunicare in un'area relativamente ristretta comporta necessariamente delle difficoltà per il sistema. Si può comunque ricavare un elemento decisamente positivo se si tiene conto che nelle situazioni di elevata densità il grouping di tipo proattivo presenta una precisione leggermente migliore tra i due protocolli.

Un confronto d'insieme tra le due diverse strategie di group formation evidenzia comunque alcuni dubbi sull'effettiva utilità del protocollo di tipo proattivo. Nelle direzioni a densità più bassa (Costa 68° e Pasubio 70°) il comportamento delle due modalità è pressoché identico e i valori di delta sono approssimativamente uguali; nei tratti più difficoltosi (Costa -8° e Pasubio 153°) i due andamenti si discostano frequentemente e quello della modalità proattiva appare più irregolare e con dei glitch occasionali. Anche se in alcuni casi la precisione in condizioni di elevata densità è leggermente migliore, il confronto tra i due protocolli verte a favore di quello di tipo reattivo a causa della maggiore stabilità presentata e – soprattutto – del costo estremamente inferiore in termini di messaggi inviati e conseguente rumore prodotto.

Queste considerazioni trovano conferma nel test successivamente condotto nello scenario Pasubio per verificare quanto le due diverse strategie costano in termini di messaggi inviati e collisioni provocate. Nella Figura 5.5 è possibile osservare i risultati aggregati di tale studio.

Il grafico inferiore raffigura il numero di veicoli rilevati ogni secondo in real mode e le curve relative al numero di pacchetti inviati dai nodi a seconda del protocollo di grouping utilizzato; quello superiore mostra due diverse nuvole di punti ciascuna rappresentativa del numero di nodi coinvolti ogni secondo in una collisione tra più trame di pacchetti (ricevuti o inviati). Dai due grafici si evince facilmente che il ricorso a protocolli proattivi e – soprattutto – l'utilizzo del modulo `NodeMap` comportano un numero maggiore di packet loss a causa dell'abbondante quantitativo di pacchetti mediamente inviati da ogni nodo.

Anche se la strategia proattiva è stata concepita per poter effettuare un'elezione più veloce e meno rumorosa in condizioni di elevata densità, il costo in termini di messaggi CAM necessari al suo funzionamento supera decisamente

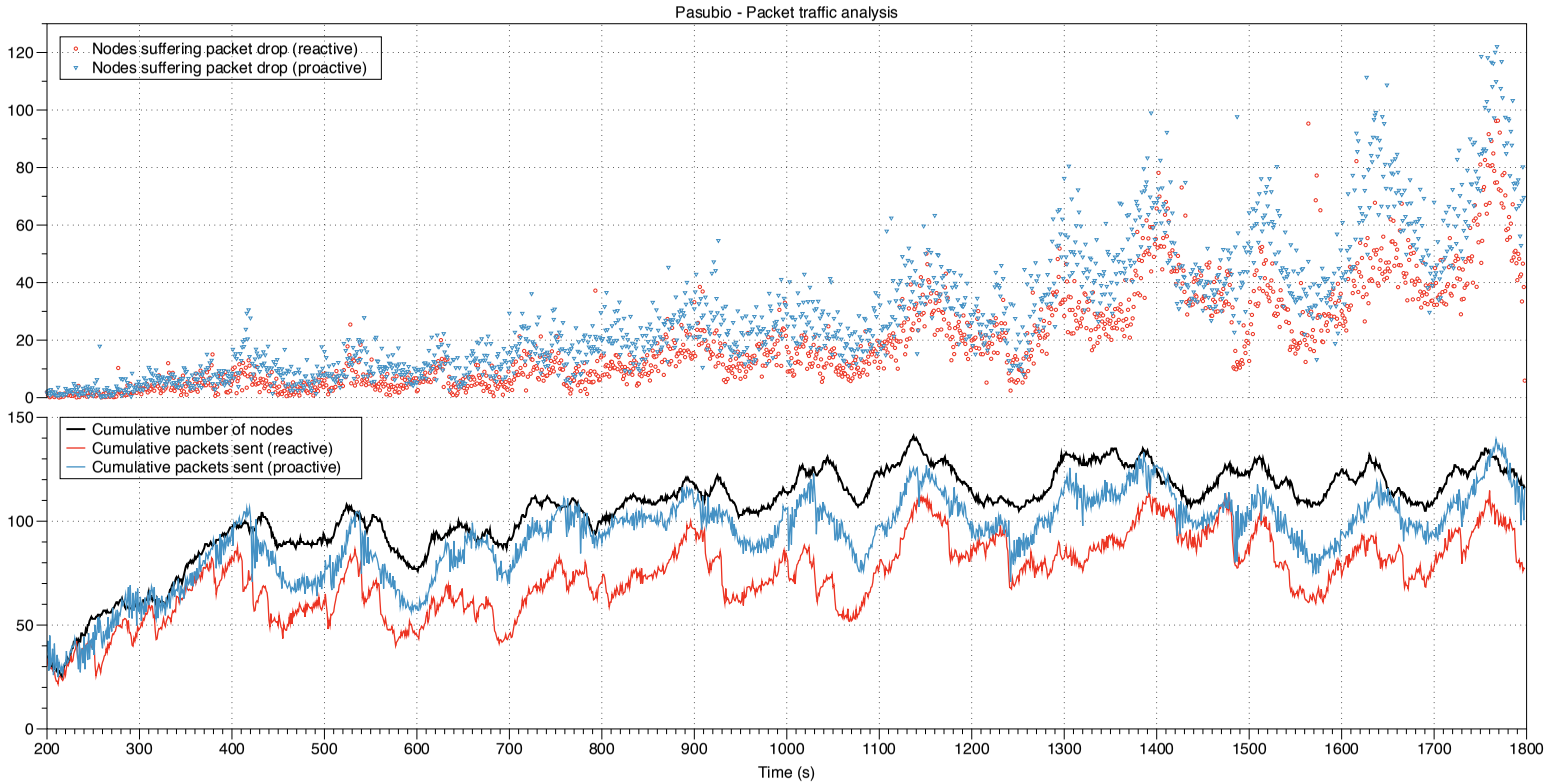

Figura 5.5: Studio sul traffico di rete generato nello scenario Pasubio dai protocolli reattivi e proattivi in prossimità della RSU di riferimento. In basso è indicato il numero di messaggi inviati in ogni secondo dai diversi nodi; in alto è raffigurato il numero di nodi coinvolti ogni secondo da una collisione tra due o più pacchetti.

questi vantaggi. È comunque doveroso ricordare che le potenzialità offerte dal fatto di poter disporre di una mappatura dei nodi circostanti possono spingersi ben oltre al semplice protocollo di group formation e costituiscono il fondamento di molte potenziali funzionalità che possono essere aggiunte all'architettura.

5.3.3 Analisi del traffico

Nel paragrafo precedente è stato affermato che i valori riguardanti il numero di nodi rilevati e la loro velocità media sono i parametri di maggiore interesse per capire l'andamento del flusso di traffico dei veicoli in una direzione. Di seguito verrà condotto un apposito test per dare maggiore supporto a tale asserzione e per capire se esistono altre informazioni utilizzabili dalle singole RSU per migliorare la conoscenza dello stato dell'ambiente.

La Figura 5.6 mostra i risultati di una serie di simulazioni effettuate nello scenario Andrea Costa in direzione 68° utilizzando entrambe le strategie di group formation. Il grafico più in basso mostra il numero dei nodi rilevati dalla RSU e i relativi valori di delta, quelli superiori visualizzano l'andamento dei valori di velocità media, densità media conformante e netta e diametro medio dei gruppi formati.

Numero di nodi rilevati

Il parametro relativo ai nodi rilevati da entrambi i protocolli è stato già commentato abbondantemente nel precedente paragrafo e i risultati evidenziati in questo test confermano quanto già detto. Entrambe le strategie seguono piuttosto bene il valore reale ottenuto mediante god-mode, il che è peraltro attestato da dei valori di delta piuttosto contenuti e pressoché identici.

Ciò che dal punto di vista delle RSU è importante ai fini di una corretta visione delle condizioni del traffico è la possibilità di osservare l'andamento sinusoidale di tale valore per comprendere quando e con quale modalità i veicoli si stanno accodando in attesa di accedere all'incrocio oppure stanno transitando e svuotando la carreggiata.

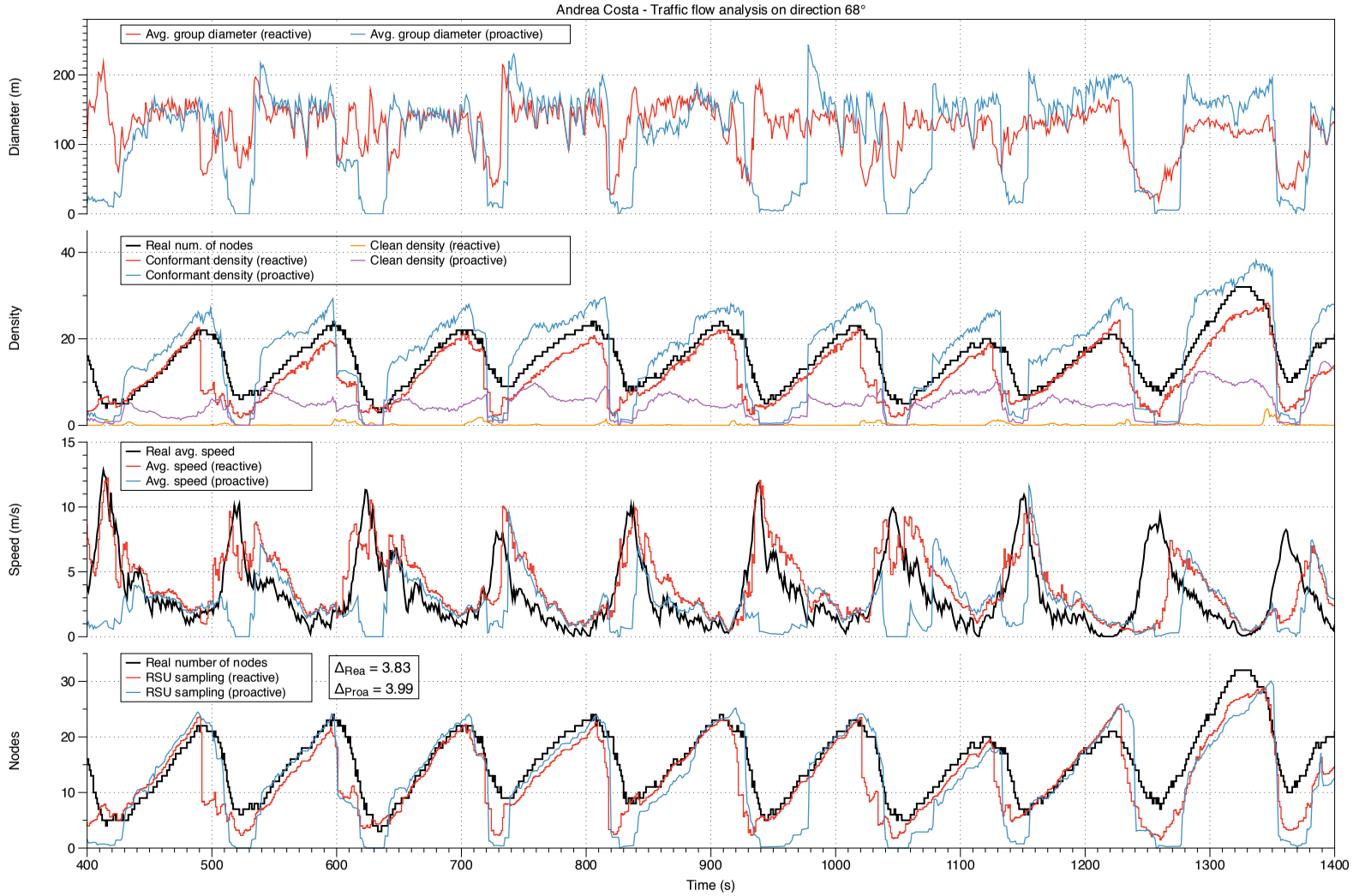

Figura 5.6: Studio sulle informazioni di traffico riportate dal sistema alla RSU di riferimento relativamente alla direzione 68° dello scenario Andrea Costa. Nel grafico vengono riportati l'andamento dei valori di densità media (conformante e netta), velocità media e numero di nodi rilevati per ciascuna delle due strategie di formazione.

Osservando nel dettaglio le linee graficate in basso è possibile distinguere le due fasi principali che i veicoli attraversano nel corso del loro passaggio: in un primo momento i valori salgono lentamente ad indicare che i veicoli stanno rallentando o sono fermi in attesa del segnale verde emesso dal semaforo; successivamente le curve scendono velocemente indicando che i veicoli hanno ripreso la loro marcia e stanno liberando il tratto stradale osservato.

Durante la prima delle due fasi i protocolli seguono molto fedelmente il reale numero di veicoli che si stanno avvicinando. Nella seconda fase la discesa delle curve è molto più ripida rispetto a quella reale; questo è perfettamente comprensibile se si considera che nel momento in cui un gruppo raggiunge la distanza limite dalla RSU e viene terminato vi è un grande quantitativo di nodi che vengono simultaneamente disattivati. Come discusso nel paragrafo precedente, l'adozione di logiche di contesto intelligenti permetterebbe di migliorare ulteriormente tale flusso informativo (vedi figure 5.2 e 5.3).

Velocità media dei nodi

Una stima della velocità assunta dai nodi in una direzione di interesse è un fattore critico per comprendere appieno le attuali condizioni del traffico. Attraverso i due indicatori di quantità e velocità l'RSU è in grado di stimare rispettivamente la portata e la rapidità di scorrimento del flusso di veicoli in transito.

Nel corso delle due fasi precedentemente introdotte, i valori di velocità tendono inizialmente a scendere lentamente a partire da un picco di massimo locale, indicando in tal modo che i veicoli stanno arrestando la loro corsa e si accodano al semaforo; la curva mantiene valori prossimi allo zero fintanto che ai veicoli viene interdetto l'accesso all'incrocio e risale velocemente quando questi riprendono a scorrere.

Le fedeltà dei risultati ottenuti rispetto alle rilevazioni effettuate in real-mode è generalmente buona ma si assiste molto spesso ad alcuni ritardi e imprecisioni nella segnalazione dei valori. In particolare, i valori riportati dai gruppi formatisi mediante strategia proattiva non riproducono fedelmente il picco di valori in quanto tali plotoni terminano in anticipo il loro ciclo di vita e

vengono disattivati prima di poter riacquistare velocità o addirittura quando i membri sono ancora in attesa di poter riprendere la marcia; quest'ultima eventualità spiega il perché dei frequenti intervalli di 10-20 secondi durante i quali le curve mantengono un valore pari a zero.

Questo comportamento potrebbe essere sintomo di un difetto del protocollo di formazione proattivo a causa del quale i leader del gruppo vengono spesso posizionati troppo vicini alla RSU di riferimento, con conseguente terminazione prematura dell'intero plotone nel momento in cui questi si avvicinano all'incrocio. Il protocollo di group formation reattivo soffre meno frequentemente di questo difetto grazie alla possibilità di effettuare dei salti di attivazione durante la procedura di elezione, come spiegato al paragrafo 4.1.1. Nel paragrafo 4.1.2 questa funzionalità viene invece preclusa al protocollo proattivo in modo da poter beneficiare di una procedura di elezione più veloce e semplice. A seguito di tali risultati si scopre tuttavia che implementare un'analoga strategia anche per tale modalità potrebbe contribuire a migliorarne le prestazioni.

Nonostante le imperfezioni appena discusse, l'RSU rimane comunque in grado di ricevere una stima soddisfacente della velocità media dei veicoli in avvicinamento.

Valori medi di densità

L'insieme delle informazioni che ogni gruppo consegna regolarmente alla propria RSU include due valori medi riguardanti i dati sulla densità conformante e netta percepite complessivamente dai suoi membri. Nel capitolo 4 questi valori sono stati definiti rispettivamente come la quantità di veicoli percepiti da un nodo tali da avere una direzione ad esso conformante e il numero di quanti tra questi risultano inattivi. Nell'architettura è stato incluso anche il concetto di densità assoluta come numero complessivo di veicoli conosciuti da un nodo ma questo parametro non è considerato utile ai fini della misurazione delle condizioni del traffico e non verrà incluso in alcun tipo di esperimento.

Nel grafico di Figura 5.6 si nota una marcata differenza nell'andamento

di tali valori tra il protocollo di tipo reattivo e proattivo. Il primo riporta dei dati pressoché coincidenti con quelli relativi al numero di membri presenti nel gruppo, il secondo presenta dei valori decisamente più elevati che seguono quasi fedelmente il campionamento dei veicoli effettuato in real-mode.

Il motivo di tale diverso comportamento è da ricercarsi nel fatto che nella modalità completamente reattiva il sistema non è in grado di rilevare la presenza di nodi inattivi e questo è testimoniato dalla relativa curva di colore arancio, la quale riporta dei valori di densità netta quasi sempre nulli. Viceversa, se i veicoli mantengono sempre attivo il modulo di `NodeMap` come spiegato nel paragrafo 4.2, essi possono essere percepiti anche quando non fanno parte di un gruppo.

La curva del grafico relativa ai valori di densità netta mostra difatti un andamento piuttosto interessante per la strategia proattiva: nella prima fase di avvicinamento all'RSU questa presenta normalmente un leggero picco per poi assestarsi su valori inferiori mano a mano che i veicoli rallentano e si accodano al semaforo. Tale iniziale innalzamento è dovuto alla rilevazione di nodi inattivi che procedono nella stessa direzione dei membri di gruppi già costituiti e anticipano in questo modo la loro futura annessione ai plotoni più vicini alla RSU. Proseguendo nella loro marcia, buona parte dei nodi inattivi precedentemente osservati verrà infatti inclusa in un plotone e il relativo contributo si trasferirà al valore di densità conformante.

Nel caso in cui il sistema ITS viene impostato per avvalersi di protocolli proattivi, le informazioni appena descritte possono essere sfruttate come rinforzo per affinare le rilevazioni riguardanti i nodi in avvicinamento alla RSU. Una semplicissima logica di correzione potrebbe calcolare la media tra i valori di densità conformante e il numero di nodi presenti in un gruppo per tentare di produrre una stima più accurata della quantità di traffico in ingresso.

La Figura 5.7 mostra l'effetto di tale correzione su due direzioni di esempio prelevate da entrambi gli scenari Costa e Pasubio: la direzione 68° è la stessa esaminata in Figura 5.6, mentre la direzione 153° in Pasubio era stata già commentata relativamente alla Figura 5.3 ed erano state evidenziate delle frequenti sottostime da parte di entrambi i protocolli, specialmente nell'arco

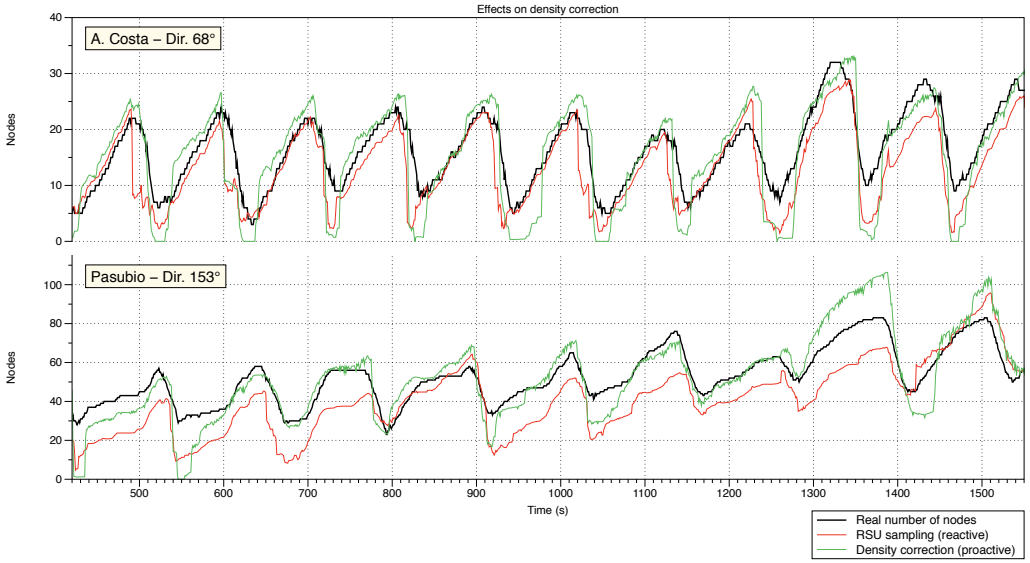

Figura 5.7: Effetti della logica di correzione di densità conformante sulla rilevazione del numero di nodi in avvicinamento alla RSU negli scenari Andrea Costa (direzione 68°) e Pasubio (direzione 153°). Sfruttando tale miglioramento il protocollo proattivo è in grado di ottenere dei risultati sensibilmente più precisi rispetto alla controparte reattiva.

di tempo compreso tra l'istante 450 e 1500.

Nel grafico riportato, la linea rossa riporta intatto il valore prodotto dal protocollo reattivo, mentre quella verde restituisce il risultato della semplice correzione effettuata grazie alle informazioni addizionali disponibili nella modalità proattiva. È possibile osservare con facilità che questa funzionalità è in grado di rendere le informazioni sul numero di nodi in avvicinamento molto più precise, anche nei casi in cui un elevato addensamento di nodi costituirebbe un fattore penalizzante per le prestazioni del sistema.

Diametro medio dei gruppi

Il grafico in Figura 5.6 riporta i valori sulla dimensione media dei gruppi che comunicano con la RSU per entrambe le configurazioni possibili del sistema. L'andamento di entrambe le linee si dimostra estremamente irregolare a riprova del fatto che nel corso del loro ciclo vitale i gruppi vanno incontro a numerose variazioni in termini di dimensione e composizione dei membri.

Escludendo il forte "rumore" nelle curve, l'andamento di tale valore presenta un pattern nel quale i gruppi tendono ad oscillare mediamente attorno ai 150 metri di diametro e a calare brevemente durante la fase di svuotamento del tratto stradale, quando cioè i gruppi vengono terminati o comunque perdono gran parte dei loro membri. In via generale è possibile affermare – pur disponendo di alcune eccezioni – che i gruppi formati dal protocollo proattivo appaiono leggermente più estesi rispetto a quelli costruiti con una strategia reattiva.

La sola informazione sulla dimensione media dei gruppi in comunicazione con l'RSU non è particolarmente utile al fine di comprendere le reali condizioni del traffico; tuttavia essa può dimostrarsi fondamentale come input per logiche più avanzate che sfruttino l'intero insieme di informazioni riguardanti i punti notevoli di ciascun gruppo (vedi Figura 4.7) conosciuto per una particolare direzione di interesse. Si potrebbe quindi ipotizzare come complemento al sistema realizzato un modulo installato nelle diverse RSU capace di costruire una mappatura spaziale dei gruppi e di capire meglio le modalità con cui questi si posizionano e si sovrappongono

5.3.4 Studio del fattore di penetrazione

Nel campo dei sistemi mobili, un indicatore di grande interesse nello studio di nuovi protocolli riguarda la performance del sistema in esame per diversi fattori di penetrazione. In altre parole, è spesso desiderabile che l'analisi di una tecnologia che non è ancora emersa nel mercato consumer includa una verifica di come il prodotto è in grado di comportarsi quando esso non è ancora stato adottato da una grossa quantità di utenti. Nel caso in questione si vuole comprendere quali performance è possibile aspettarsi dall'architettura presentata nei precedenti capitoli quando questa si trova ad operare in un ambiente caratterizzato da percentuali di adozione significativamente inferiori al 100%.

Per presentare questo tipo di test in maniera concisa e facilmente leggibile si è scelto di concentrare l'attenzione sullo scenario Andrea Costa e ci si è limitati a graficare i soli risultati derivati dall'utilizzo dei protocolli di tipo

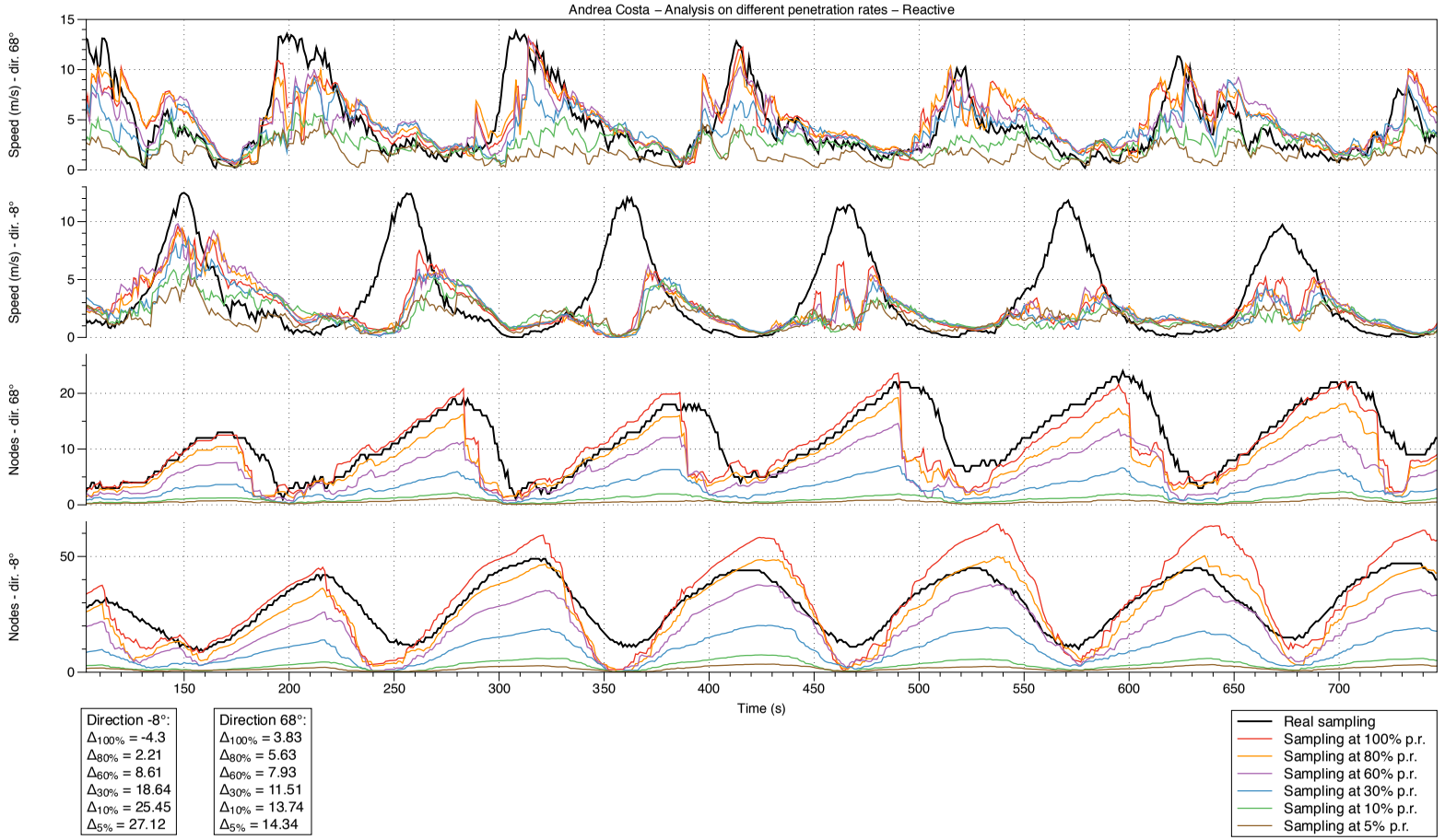

Figura 5.8: Comportamento del sistema nello scenario Andrea Costa a fronte di diversi fattori di penetrazione della tecnologia in esame. I risultati si riferiscono alle direzioni di 68° e -8° e all'utilizzo esclusivo di protocolli di tipo reattivo.

reattivo. Questa scelta è giustificata dal fatto che analoghi studi condotti sullo scenario Pasubio servendosi di entrambe le strategie di grouping hanno mostrato risultati del tutto equivalenti a quelli di seguito esposti.

Nella Figura 5.8 viene riportato il comportamento del sistema relativamente alle informazioni di quantità di nodi rilevati e velocità media per le direzioni di valore 68° e -8°. Il tipo di scenario e le impostazioni utilizzate sono le stesse dei test condotti al paragrafo 5.3.2 ma si è qui proceduto a svolgere sei classi di simulazioni ciascuna con un numero di veicoli shadow (cioè sprovvisti del sistema ITS) via via crescente; ogni classe di simulazione prevede comunque l'esecuzione di 30 iterazioni di test con lo stesso input. I fattori di penetrazione considerati sono pari al 100%, 80%, 60%, 30%, 10% e 5%. I veicoli di tipo shadow vengono scelti in maniera uniforme all'interno dello stesso insieme di veicoli registrati nella relativa traccia di mobilità.

Relativamente al numero dei nodi rilevati, le curve mostrate in figura presentano un declino in proporzione diretta e perfettamente lineare al decrescere della percentuale di veicoli dotati del sistema ITS. Questo risultato è del tutto normale nel momento in cui la scelta della composizione dei veicoli avviene in maniera uniforme. Si noti in ogni caso come le curve rappresentative delle percentuali più basse presentino una forma decisamente più morbida e regolare: questa sorta di perdita di definizione è indubbiamente imputabile al progressivo decrescere del numero di nodi coinvolti nel sistema, il che comporta inevitabilmente una sensibile diminuzione dei dettagli relativi allo stato reale del traffico.

Una variazione lineare del tutto analoga si ottiene anche considerando uno stesso scenario sotto diversi livelli di densità. Sono stati compiuti alcuni test per studiare la risposta del sistema a delle istanze in cui veniva progressivamente ridotto il numero complessivo di veicoli simulati. In questi casi le diverse curve di valori hanno presentato un andamento simile a quello mostrato in Figura 5.8 e si è pertanto scelto di non presentarle in questo testo.

L'effetto che le variazioni del fattore di penetrazione comportano sul rilevamento della velocità media dei nodi mostra un risultato prevedibile ma comunque molto interessante: specialmente nel caso della direzione -8° le

diverse curve non presentano una correlazione molto stretta e lineare con la percentuale di veicoli shadow presenti nell'ambiente. Questo fenomeno è perfettamente giustificabile se si considera che una diminuzione del numero di veicoli in grado di partecipare alle comunicazioni non influisce in alcun modo sul comportamento degli stessi in termini di posizione e velocità assunte nel tempo.

Anche quando il numero di nodi in grado di scambiare dati con le RSU è piuttosto basso, le informazioni relative alla loro velocità riescono comunque a riflettere adeguatamente le condizioni complessive dell'insieme di veicoli nel loro stesso senso di marcia. Questa affermazione sembra avere maggiore validità se applicata nei tratti stradali maggiormente afflitti da fenomeni di congestione. I veicoli che transitano in direzione 68° mantengono in media velocità più elevate rispetto a quelli che viaggiano lungo il percorso in -8° e difatti le diverse linee relative al primo caso mostrano una maggiore variabilità al passaggio da una percentuale di adozione all'altra. Con ogni probabilità questo effetto è dovuto a due diverse concause: da una parte un maggiore numero di veicoli in transito lungo un percorso comporta sicuramente una maggiore possibilità che più nodi in grado di comunicare vengano a raggrupparsi e migliorino pertanto le informazioni consegnate alla relativa RSU; è inoltre possibile che i casi nei quali un insieme di veicoli si sposta in maniera lenta e compatta verso il centro dell'incrocio a causa di una congestione comportino una maggiore capacità dei singoli nodi di rappresentare lo stato complessivo degli altri loro pari, anche se questi ultimi non dispongono di un sistema di comunicazione wireless.

Come è già stato detto precedentemente, la maggior parte dei risultati riportati nei grafici di questo capitolo è ulteriormente migliorabile per mezzo di una logica addizionale installata sulle singole RSU. Anche la presenza di numerosi veicoli shadow può essere compensata a priori a patto di potere assumere come valide alcune importanti ipotesi di partenza. La prima riguarda il fatto di conoscere con buona precisione la percentuale di veicoli dotati di sistema ITS; la seconda condizione – non sempre molto realistica – richiede che la distribuzione delle due classi di nodi sia uniforme, così come è stata assunta per il test sopra presentato.

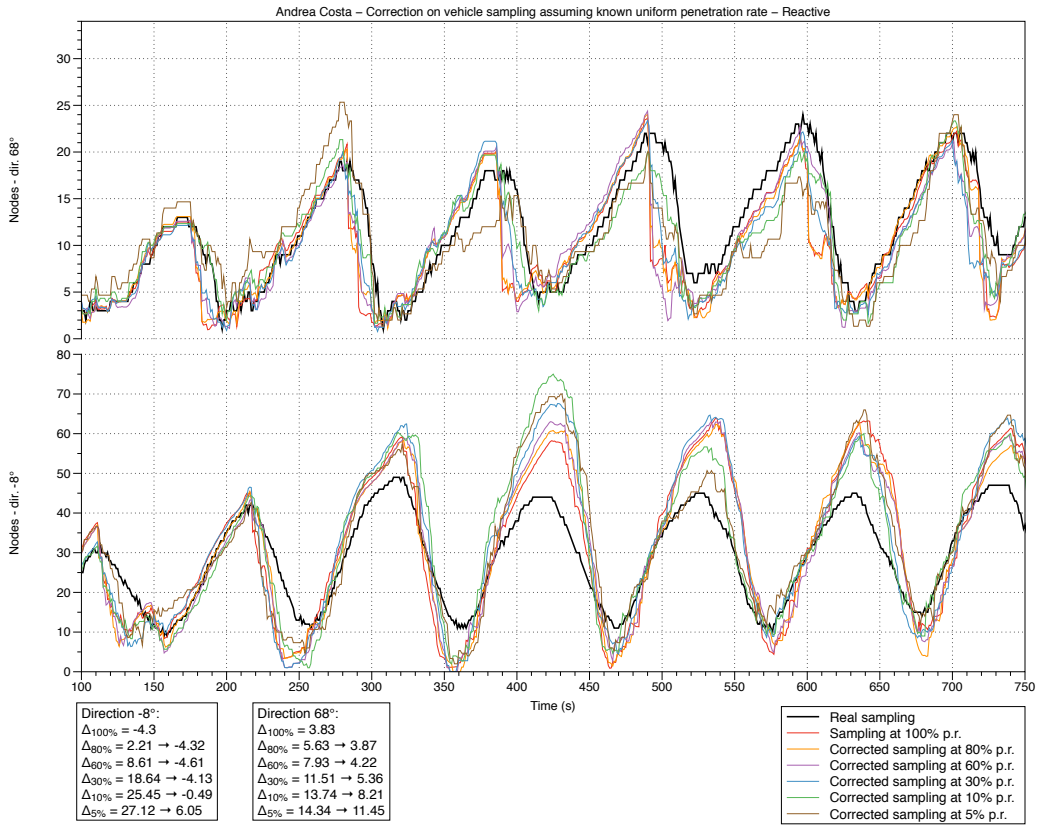

Figura 5.9: Correzione dei dati parziali ottenuti in Figura 5.8 sotto ipotesi di percentuale di penetrazione nota e distribuzione uniforme delle diverse classi di veicoli.

Se entrambe le assunzioni sono ammissibili, è possibile introdurre una semplicissima logica a livello di RSU per correggere le rilevazioni ottenute negli scenari dove la percentuale di penetrazione è conosciuta. Se per esempio è noto a priori che la popolazione dei veicoli è composta al 40% di veicoli shadow (curve a 60% del grafico precedente) si potrà applicare un fattore correttivo pari a 1.667 per tentare di rettificare la stima in proprio possesso. Ovviamente questa tecnica va applicata soltanto a quelle tipologie di dati che risentono in proporzione diretta dell'abbassamento della percentuale di adozione: i dati relativi alla velocità media ad esempio non devono essere corretti utilizzando questo principio.

La Figura 5.9 mostra gli effetti di tale operazione applicandola ai dati relativi al numero di veicoli rilevati in Figura 5.8. Gli effetti della correzione

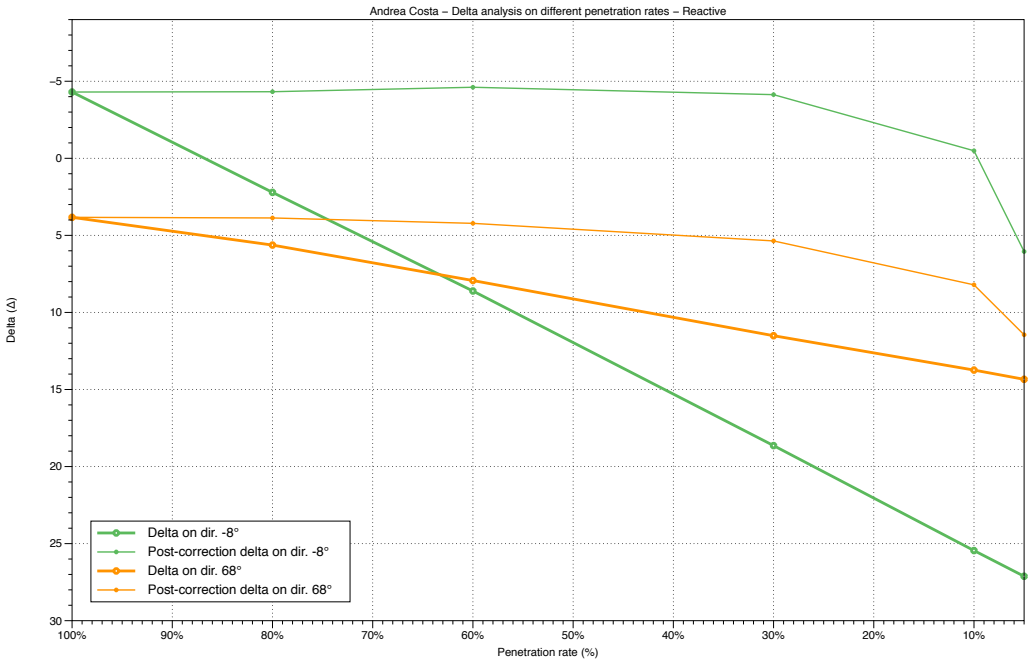

Figura 5.10: Andamento dei valori delta in presenza o assenza della logica correttiva presentata in Figura 5.9.

sono piuttosto buoni e le varie curve di percentuale tendono ad allinearsi come previsto, in particolare per percentuali superiori al 25%. Anche i valori di delta riportati in basso confermano che il sistema è in grado di mantenere prestazioni decisamente buone anche con una percentuale di penetrazione vicina al 10%. Nei casi caratterizzati da una forte congestione (particolarmente degno di nota è il picco in direzione -8° al secondo 420) si assiste tuttavia ad una marcata sovrastima del reale numero di veicoli da parte delle curve a percentuale più bassa.

La Figura 5.10 permette di comprendere in maniera diretta sia l'andamento delle prestazioni del sistema al variare della percentuale di nodi in grado di comunicare, sia l'effetto della correzione sopra indicata sulla qualità dell'euristica. In assenza di interventi sui dati le curve relative ai valori di delta mostrano un andamento estremamente lineare, in linea con le considerazioni precedenti. La pendenza delle rette varia sensibilmente tre le due diverse direzioni il che suggerisce che l'entità del calo di prestazioni dipende in larga misura anche dalla particolare conformazione del tratto stradale o

Tabella 5.2: Differenza tra i parametri che differenziano le due classi di veicoli ITS.

Parametro	Modulo	Full-class vehicle	Mid-class vehicle
`TxGain`	`ns3::YansWifiPhy`	1 dB	-5.1 dB
`RxGain`	`ns3::YansWifiPhy`	1 dB	-4.9 dB
`Resolution`	`NodeSampler`	1000 ms	2000 ms
`PositionRadius`	`NodeSampler`	20 m	30 m
`PositionVariance`	`NodeSampler`	1.8	2.0
`DirectionVariance`	`NodeSampler`	0.002	0.005

dalle caratteristiche complessive del traffico interessato per quel percorso.

A seguito delle correzioni, le curve mostrano un comportamento molto simile, pur disponendo ovviamente di quote differenti. Si noti come la qualità dei dati manipolati rimanga sostanzialmente immutata in caso di penetrazioni superiori al 30% per poi scendere in maniera sempre più marcata mano a mano che si raggiungono percentuali inferiori al 10%.

5.3.5 Studio sulle diverse classi di veicoli

I risultati esposti fino a questo punto prevedevano una popolazione di nodi composta esclusivamente da veicoli shadow oppure appartenenti alla classe full. Nell'analisi seguente si vuole verificare come varia il comportamento del sistema a seguito dell'introduzione di una certa percentuale di veicoli di classe media.

La Tabella 5.2 mostra le differenze di valore per i sei attributi che differenziano le due classi di veicoli: per effetto dei primi due parametri il range in ricezione e trasmissione delle interfacce di comunicazione viene abbassato a circa 100 metri, gli altri attributi intervengono sul modulo di campionamento introdotto nel paragrafo 3.3 e comportano una diminuzione della precisione in termini di posizione e direzione rilevate dal nodo, oltre ad innalzare a due secondi l'intervallo di campionamento. Come spiegato nel paragrafo 2.3, questi accorgimenti vengono presi per simulare dei veicoli le cui funzionalità di comunicazione e posizionamento vengono realizzate da dispositivi *handheld* con prestazioni inferiori rispetto a dei sistemi V2X dedicati.

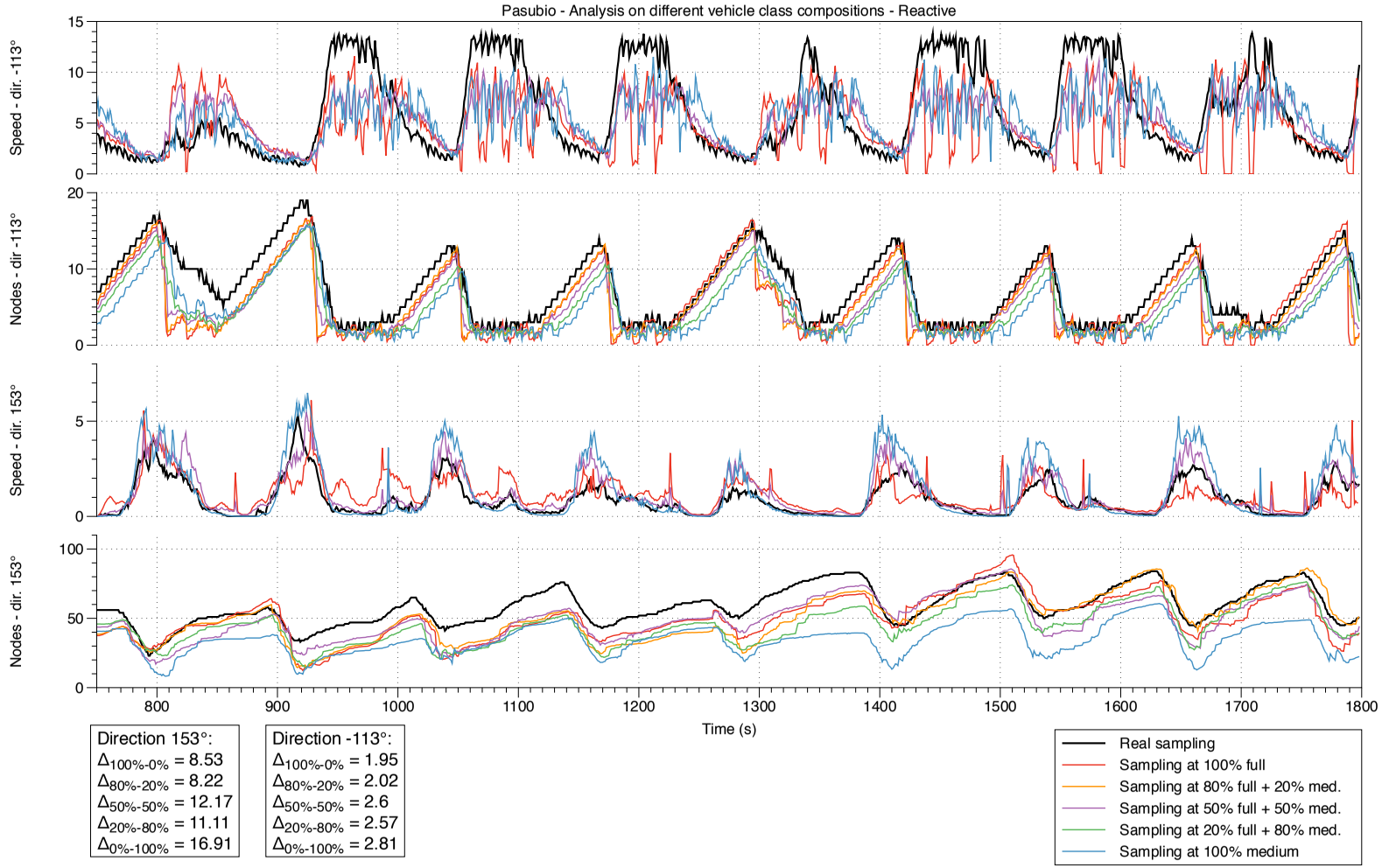

Figura 5.11: Comportamento del sistema nello scenario Pasubio a fronte di diverse composizioni di veicoli di classe media e full. I risultati si riferiscono alle direzioni di 153° e -113° e all'utilizzo esclusivo di protocolli di tipo reattivo.

La Figura 5.11 mostra i risultati di cinque diverse simulazioni eseguite nello scenario Pasubio variando di volta in volta la composizione della popolazione di veicoli utilizzata. I due estremi sono rappresentati dalle linee rosse e blu che rappresentano rispettivamente popolazioni composte da soli veicoli full e medi. La figura riporta i valori relativi al numero di nodi rilevati e alla loro velocità media nelle due direzioni di 153° e -113°. Nei grafici di velocità le curve appartenenti ai mix di popolazione intermedi (20%+80% e 80%+20%) sono state omesse per esigenze di maggiore leggibilità, il loro comportamento risulterebbe comunque analogo alle considerazioni esposte di seguito.

Per rendere il grafico più leggibile si è scelto di mostrare esclusivamente il comportamento dei protocolli reattivi. Il ricorso a strategie proattive ha evidenziato una serie di dati del tutto simile a quella di seguito discussa, pur con alcune occasionali irregolarità dovute alle già citate imperfezioni di tale metodo di raggruppamento.

A una veloce analisi è possibile notare come i valori che vengono maggiormente influenzati dal passaggio da una classe all'altra risultano essere quelli relativi al numero di veicoli registrati dai gruppi: le popolazioni con prevalenza di veicoli full registrano quasi ovunque dei valori più elevati, mentre l'uso di nodi di classe inferiore porta a sottostimare sensibilmente il reale numero di veicoli nell'area interessata.

Questo risultato è del tutto prevedibile e risulta facilmente giustificabile considerando la minore portata dei veicoli di fascia media che preclude loro la possibilità di formare gruppi molto vasti e soprattutto di comunicare con l'RSU a distanze superiori a circa 100 metri. Occorre inoltre ricordare che in tutti i test effettuati fino a questo momento il rilevamento dei veicoli in real-mode è impostato per tracciare nodi a una distanza compresa tra 20 e 170 metri, il che comporta un notevole svantaggio per i membri della classe meno performante.

Lo studio delle diverse curve relative ai valori di velocità media non evidenzia grosse differenze tra le diverse classi. La leggera tendenza dei dispositivi meno performanti a sovrastimare il reale valore di velocità è in realtà dovuto ancora una volta a un disallineamento dei parametri in real-mode: il

sistema composto da nodi di fascia inferiore è in grado di operare su una distanza massima di circa 100 metri mentre la procedura di log è programmata per considerare veicoli distanti fino a 170 metri. Durante la fase di svuotamento del tratto stradale, i valori di velocità consegnati alla RSU coprono quindi soltanto quei nodi più vicini ad essa i quali riescono a riacquistare velocità con maggiore anticipo rispetto a quelli più distanti.

La Figura 5.12 ripropone gli stessi risultati appena discussi ma con una procedura di output impostata per registrare i veicoli in un tratto che spazia da 20 a 100 metri di distanza dalla RSU; vengono inoltre graficate le linee relative all'utilizzo dei protocolli proattivi. Le rilevazioni sul numero di veicoli e i valori di delta tornano ad essere in linea con quelle reali, confermando quanto detto in precedenza.

Si noti che le curve (in particolar modo quelle relative ai protocolli proattivi) tendono a sovrastimare abbondantemente il numero di veicoli presenti nel tratto stradale a partire dall'istante 1300: questo fenomeno è dovuto al fatto che si vengono in tal caso a formare gruppi di veicoli che riescono complessivamente a coprire una distanza superiore ai 100 metri e i valori consegnati all'RSU riflettono le reali condizioni di traffico in quanto comprendono il contributo dei nodi più distanti che vengono ignorati dalla procedura in real-mode.

Il grafico più in alto in figura permette di osservare un ulteriore risultato degno di nota. Se si confronta la nuvola dei punti relativi alle perdite di pacchetti con quella di Figura 5.5 è possibile notare come il sistema presenti un numero di collisioni sensibilmente inferiore e stabile rispetto alle simulazioni effettuate con nodi dotati di una maggiore portata. Questo esito costituisce a tutti gli effetti un vantaggio per il corretto funzionamento dei protocolli ed è interamente dovuto al fatto che i messaggi trasmessi da un nodo hanno una probabilità tanto maggiore di disturbare o essere disturbati da altri dispositivi quanto maggiormente estesa è la portata del segnale. Tale caratteristica si percuote positivamente anche sui valori di delta, i quali presentano un valore inferiore a quelli ottenuti con veicoli di classe superiore.

La possibilità di monitorare veicoli in un'area più estesa sembra pertanto essere l'unico reale svantaggio dovuto all'utilizzo di dispositivi di fascia media.

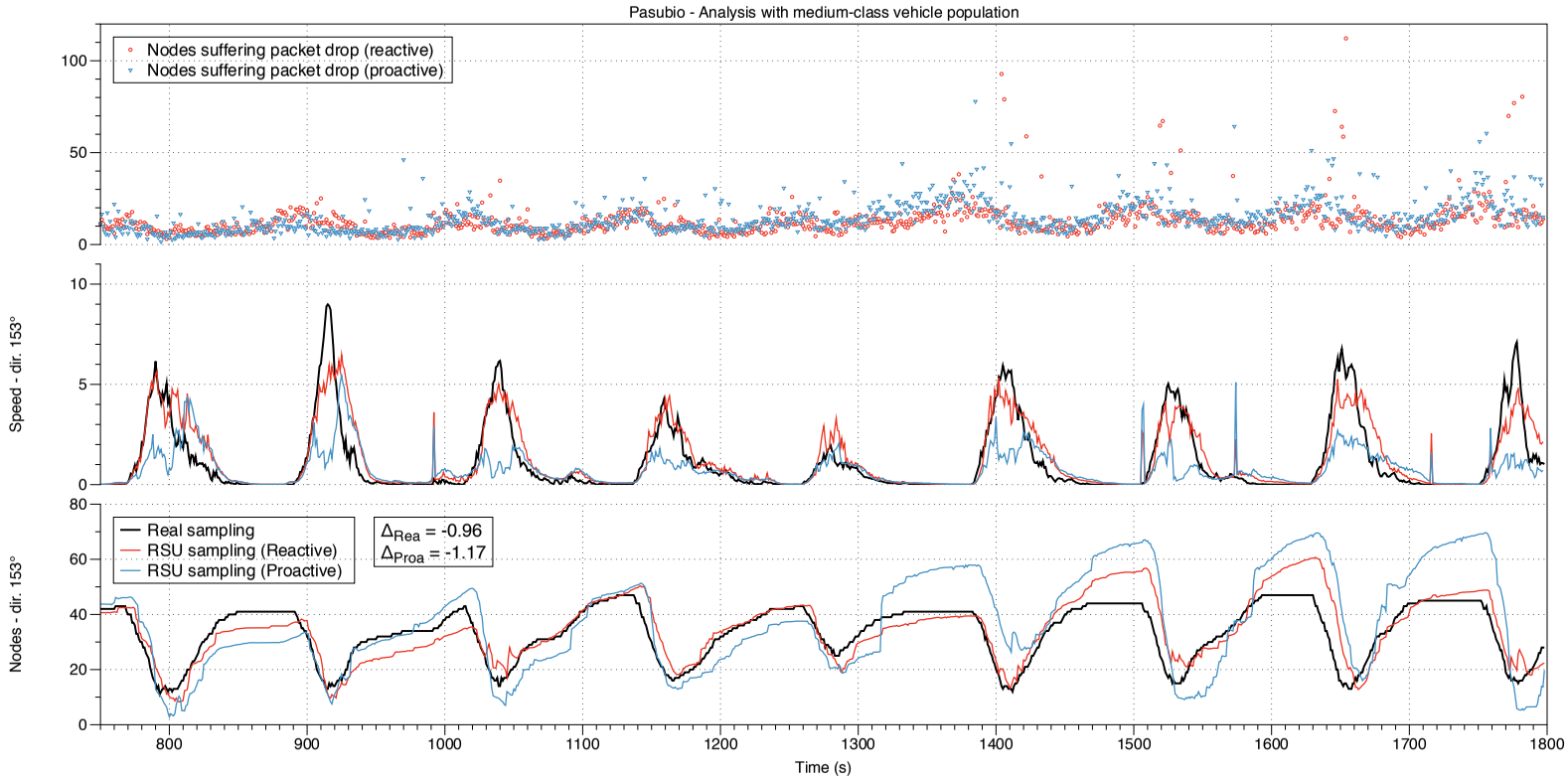

Figura 5.12: Simulazione nello scenario Pasubio con popolazione interamente composta da veicoli di classe media. Il sistema si mantiene capace di produrre stime di traffico di buona qualità e mostra inoltre una minore sensibilità ai problemi di collisione tra pacchetti.

Nel complesso, l'insieme dei protocolli sviluppati riesce comunque a portare a compimento le funzionalità critiche per il sistema e a fornire una buona stima dello stato del traffico, pur essendo quest'ultima inevitabilmente ristretta a un'area ridotta.

Per contrastare le limitazioni dovute a interfacce di comunicazione dotate di uno scarso raggio operativo è possibile in ogni caso estendere l'architettura del sistema per fare in modo che sia possibile consegnare all'RSU i dati prodotti da eventuali gruppi fuori dalla sua portata, in modo che il loro contributo possa comunque essere considerato. Per realizzare tale funzionalità è necessario predisporre una forma di comunicazione e collaborazione tra gruppi conformanti vicini al fine di trasportare le informazioni relative a un'area di interesse verso dei nodi situati a distanze altrimenti irraggiungibili, come verrà suggerito nel paragrafo 6.3.

Nel prossimo capitolo ci si preoccuperà di riassumere i principali difetti del sistema riscontrati durante l'ampio lavoro di test al quale esso è stato sottoposto. In relazione alle problematiche riscontrate e a seguito dell'esperienza maturata nel corso della progettazione e implementazione dell'architettura verranno proposte alcune modifiche e funzionalità aggiuntive per il progetto in grado di incrementarne le prestazioni e la robustezza.

Capitolo 6

Conclusioni e sviluppi futuri

La motivazione principale che ha spinto alla realizzazione dello studio del sistema presentato nel corso dei capitoli precedenti è stata l'esigenza di sperimentare e analizzare delle metodologie e dei protocolli semplici ma comunque efficaci per il raggruppamento dei veicoli in transito verso dei punti di interesse e per la raccolta di dati da questi ultimi finalizzata a trasferire verso dei dispositivi infrastrutturali fissi un flusso di informazioni integrato e compatto in grado di fornire delle euristiche sulla composizione e sull'entità del traffico suddivise in base alla particolare direzione percorsa dalle vetture in arrivo. La base di conoscenza prodotta dal sistema deve essere tale da potere essere successivamente impiegata come input per l'attuazione di logiche di gestione del traffico dinamiche e intelligenti.

I numerosi test condotti sull'architettura nel corso del suo sviluppo e in particolare quelli esposti nel capitolo 5 hanno dimostrato come il sistema sia in grado di rispondere a tali requisiti: i protocolli di group formation permettono di raggruppare correttamente i veicoli in avvicinamento alle unità infrastrutturali in base alla loro direzione e di eleggere correttamente un gestore per il plotone. Le ulteriori logiche prodotte consentono poi di gestire adeguatamente il ciclo di vita dei gruppi formati e promuovono la collaborazione reciproca dei relativi membri, anche a fronte di densità relativamente elevate, hardware eterogeneo e pattern di mobilità differenti e spesso imprevedibili.

Il flusso di dati che i diversi gruppi riescono a consegnare alle unità infra-

strutturali a cui fanno riferimento riguardano informazioni sulla composizione e la dimensione del gruppo stesso, sulla velocità dei suoi componenti e su diverse tipologie di densità percepite da questi ultimi. Confrontando i dati ricavati in real-mode con i risultati forniti dai protocolli alle varie RSU si è visto come questi ultimi siano in grado di fornire una stima sufficientemente accurata delle attuali condizioni del traffico circostante. I numerosi parametri che regolano le funzionalità dei diversi componenti dell'architettura discussa permettono in molti casi di controllare il comportamento delle varie logiche di protocollo, personalizzandole in questo modo alle proprie esigenze; il sistema si presta inoltre a ulteriori possibilità di estensione e revisione, alcune delle quali verranno discusse al termine del presente capitolo.

6.1 Risultati ottenuti

Oltre ad averne verificato il corretto funzionamento in base ai requisiti richiesti, i diversi test condotti nel precedente capitolo hanno consentito di comprendere in quali casi d'uso è possibile ottenere prestazioni migliori dal sistema e che tipologie di vantaggi è lecito attendersi da ciascuna delle due modalità di funzionamento implementate.

Nel paragrafo 5.3.2 si è immediatamente osservato che i protocolli reattivi sono generalmente in grado di ottenere prestazioni simili o leggermente superiori rispetto a quelli proattivi (Figure 5.2 e 5.3), i quali comportano molto spesso un incremento del numero di messaggi inviati superiore al 20% e un conseguente aumento di pacchetti persi a causa di collisioni (Figura 5.5).

Il protocollo di formazione proattiva presenta un ulteriore svantaggio dovuto alla sua tendenza ad eleggere leader in posizione troppo frontale, il che comporta un ciclo di vita di minore durata per i gruppi e la conseguente impossibilità di monitorarne lo stato negli ultimi metri che separano i nodi dal centro dell'incrocio (Figura 5.6); in particolare, questo comportamento incide significativamente sulla capacità di tracciare la velocità media dei veicoli durante le ultime fasi di avvicinamento all'RSU.

I protocolli proattivi riescono in ogni caso a restituire una stima più accurata delle informazioni sulla densità circostante per via della loro capacità di

percepire nodi non ancora raggruppati; questo vantaggio si traduce eventualmente in una precisione nel conteggio del numero di nodi spesso superiore a quella offerta dalla controparte reattiva quando i dati sulla densità vengono usati come rinforzo (Figura 5.7).

La distanza dall'incrocio a partire dalla quale il sistema riesce ad operare dipende principalmente dalla particolare conformazione del tratto stradale monitorato e dal raggio di portata dei veicoli (Figura 5.4). I leader di gruppo non possono ovviamente comunicare con le RSU oltre tale soglia, ma riescono comunque a informare l'infrastruttura della presenza e dello stato di nodi posizionati leggermente oltre ad essa.

Nel paragrafo 5.3.4 viene condotta una serie di test per verificare il comportamento dell'architettura in scenari con percentuali di penetrazione più bassa e determinare con quali modalità le qualità delle rilevazioni diminuiscono in relazione a tale parametro. La Figura 5.10 riassume in estrema sintesi questi risultati: in assenza di interventi, il sistema vede un deterioramento lineare della qualità dei dati raccolti e un crollo di funzionalità attorno a una percentuale poco inferiore al 10%. Le semplici correzioni proposte nel corrispondente paragrafo permettono di mantenere relativamente alta l'affidabilità delle informazioni ricevute fino a percentuali superiori al 5% (Figura 5.9).

Gli esperimenti condotti nel paragrafo 5.3.5 dimostrano infine che il sistema è in grado di funzionare degnamente anche ricorrendo a sensori e interfacce appartenenti a una classe meno performante (Figura 5.11). L'elemento che più di tutti compromette l'output dei protocolli risulta essere la portata dei singoli nodi, la quale incide ovviamente sulla capacità del sistema di formare gruppi di dimensione maggiore e di ricevere informazioni relative a nodi maggiormente distanti. Nonostante questi vantaggi, la riduzione del range operativo dei nodi comporta un minore rumore complessivo in quanto i pacchetti in uscita hanno una minore probabilità di interferire con quelli trasmessi da altri veicoli.

6.2 Limitazioni del sistema

Nonostante la buona resa del sistema realizzato, esso non ha di certo la pretesa di risultare esente da difetti o punti di debolezza. Scopo del presente paragrafo è quello di discutere le limitazioni attuali dell'architettura proposta in questo testo, individuando particolari scenari e situazioni limite che possono precluderne l'utilizzo o limitarne drasticamente le performance. Questa analisi verrà infine seguita da un insieme di proposte e suggerimenti su come migliorare le funzionalità del sistema attraverso ulteriori modifiche e sviluppi. Alcuni di questi *enhancement* sono stati effettivamente considerati durante la fase di progettazione ma successivamente scartati per dare maggiore spazio a elementi più prioritari o per limitazioni di tempo.

Le maggiori difficoltà che possono presentarsi nel momento in cui si tenta di applicare l'uso del sistema a degli scenari d'uso reali sono provocate dal concetto di direzione formalizzato nel paragrafo 2.2.2: tale principio è infatti alla base del processo di raggruppamento dei veicoli e permette oltretutto di differenziare i flussi di dati verso le RSU, permettendo di separarli in base alla specifica direzione di provenienza. Questa strategia funziona molto bene negli incroci stradali di conformazione più "classica", ovvero caratterizzati da due o al massimo tre strade approssimativamente perpendicolari che si incontrano nel punto in cui viene posizionata l'unità infrastrutturale.

Possono esistere tuttavia numerosi casi in cui una conformazione particolarmente atipica della rete stradale comporta una serie difficoltà nel momento in cui devono essere individuati gli esatti valori di direzione che una RSU vuole monitorare. Nel seguito si farà riferimento alla Figura 6.1 per meglio comprendere i casi critici enunciati.

Interferenza tra direzioni parallele

In entrambi gli scenari presi come esempio vengono evidenziate nelle ellissi tratteggiate in rosso alcune coppie di strade appartenenti a percorsi nettamente differenti e i cui relativi flussi di veicoli andrebbero mantenuti separati. Tali carreggiate presentano tuttavia valori di direzione quasi identici tali per

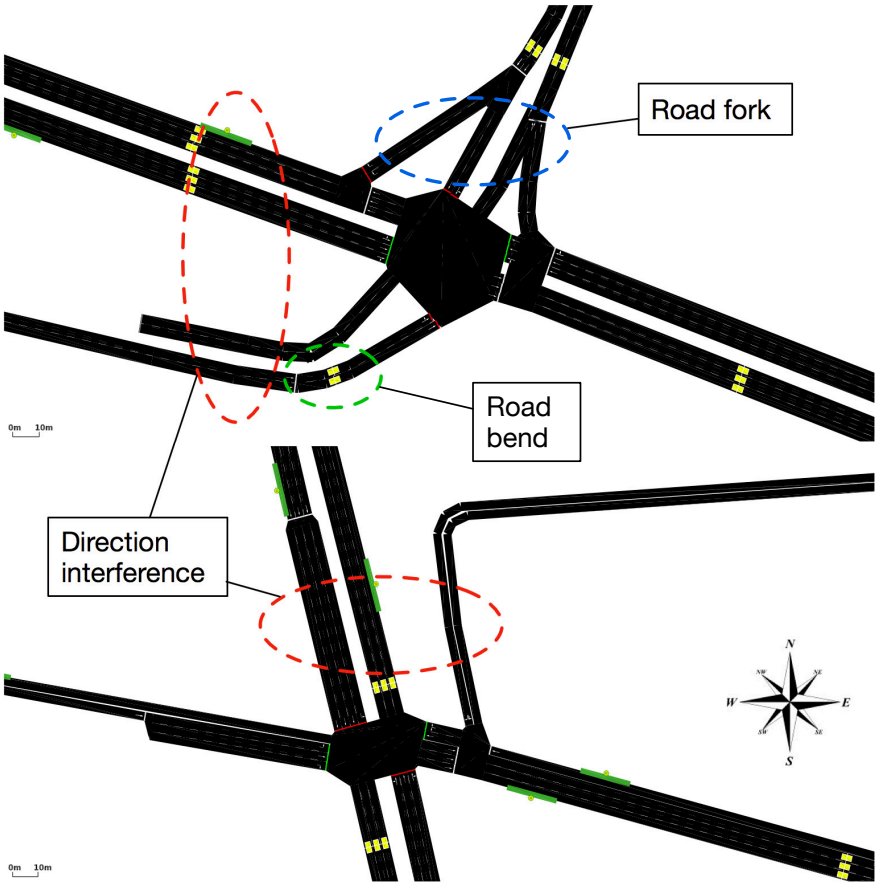

Figura 6.1: Esempi di scenari che possono presentare difficoltà di impiego per il sistema realizzato: i cerchi rossi indicano interferenze di direzioni tra strade contigue approssimativamente parallele, in quello blu vediamo la presenza di svincoli o deviazioni di forma atipica, il punto in verde indica una strada che effettua un brusco cambiamento di direzione poco prima dell'incrocio.

cui il sistema non riuscirebbe a distinguere quali veicoli stanno effettivamente percorrendo una o l'altra strada.

Più in generale, nel momento in cui viene individuata una qualsiasi direzione di interesse per una RSU, qualsiasi percorso stradale – anche secondario oppure del tutto irrilevante con il controllo del traffico per quell'incrocio – caratterizzato da un valore di direzione simile o identico ad esso comporta un problema di interferenza tra due differenti strade. A seguito di tale disturbo, il sistema perde la capacità di distinguere i veicoli che transitano lungo uno qualsiasi dei due percorsi, provocando così una "fusione" tra due diverse

strade e alterando considerevolmente le informazioni di traffico consegnate all'infrastruttura.

Nel paragrafo 6.3 verranno presentate alcune modifiche che è possibile applicare all'architettura per risolvere o comunque alleviare le problematiche appena individuate.

Diramazioni in prossimità di incroci

Nel primo dei due esempi viene indicata in blu una particolare conformazione stradale in grado di comportare alcune difficoltà di utilizzo per il sistema realizzato: i veicoli che raggiungono l'incrocio seguendo una direzione sud-ovest si trovano a poter scegliere tra due diversi percorsi a una distanza di poche decine di metri dal punto in cui verrebbe idealmente posizionata una RSU. Le vetture che scelgono di immettersi direttamente nell'arteria stradale principale orientata verso nord-ovest seguono per alcuni metri una direzione differente rispetto agli altri veicoli che procedono verso sud-ovest ed entrano effettivamente nell'incrocio.

A fronte di questa situazione possono presentarsi dei risultati differenti. Se l'RSU non è interessata a tenere traccia dei veicoli che non entrano nell'incrocio si troverebbe comunque in ingresso dei dati che includono il contributo dei veicoli indesiderati, almeno fino a quando questi non effettuano la svolta al punto indicato e percorrono alcuni metri. Se invece si è intenzionati a monitorare anche il tratto stradale successivo alla diramazione, i gruppi che andranno a formarsi avrebbero un ciclo di vita estremamente breve considerando che il segmento che congiunge le due strade principali ha una lunghezza di poche decine di metri; di conseguenza il flusso di informazioni che essi consegnano alla RSU di riferimento potrebbe essere incompleto o comunque estremamente frammentato a causa del breve periodo di attività concesso ai plotoni. Infine, se i veicoli che percorrono la diramazione non raggiungono la distanza minima prefissata dalla RSU[1], i gruppi non concluderebbero correttamente il loro ciclo vitale e questo potrebbe portare a un'ulteriore diminuzione della qualità dell'euristica.

[1]Si consulti il paragrafo 4.3.2.

Direzioni irregolari

Nel cerchio verde in Figura 6.1 viene indicato un punto in cui una strada di interesse presenta una forte irregolarità alcuni metri prima di confluire nell'incrocio. Tralasciando i problemi derivanti dalle interferenze tra strade parallele, i veicoli che percorrono la carreggiata subiscono un brusco cambio di direzione quando si trovano ad essere piuttosto vicini alla RSU.

Il sistema funziona generalmente bene in presenza di rettilinei sufficientemente lunghi posti immediatamente prima all'incrocio che si intende gestire; a seguito dell'irregolarità di alcune strade i veicoli potrebbero assumere direzioni non previste dalla RSU e inattivarsi per diversi secondi oppure i gruppi potrebbero arrivare a formarsi molto in ritardo, riducendo nettamente il loro ciclo di vita e deteriorando di conseguenza le performance del sistema.

Anche se è possibile ricorrere ad alcuni *workaround* per mitigare i cali di prestazioni imputabili a percorsi stradali irregolari, il problema resta comunque di difficile soluzione e richiede certamente uno studio più approfondito. Nel paragrafo successivo viene proposto un sistema basato sull'uso di *waypoint* per poter risolvere questo inconveniente.

6.3 Sviluppi possibili

Vengono di seguito suggerite alcune possibili estensioni al sistema che se implementate potrebbero contribuire in maniera significativa ad aumentarne le performance, a migliorare la qualità dei dati ottenuti e sopratutto a rendere l'architettura più flessibile e robusta. Tutte le strategie seguenti sono state ideate nel corso delle ultime fasi di implementazione del progetto e le limitazioni di tempo e risorse non ne hanno purtroppo reso possibile l'inclusione nella soluzione fin qui presentata.

Ottimizzazione dei parametri di sistema

Nel corso dei capitoli 3 e 4 sono stati presentati numerosi attributi – in particolare quelli relativi alle diverse logiche di protocollo – mediante i quali è

possibile controllare in maniera molto flessibile i parametri di funzionamento del sistema e il comportamento dei componeti sviluppati.

Nell'ottenere i risultati esposti nel capitolo 5 sono stati utilizzati per i vari attributi i valori di default indicati nelle rispettive tabelle, salvo ove diversamente specificato. Tali impostazioni predefinite non sono tuttavia frutto di un'estesa ricerca finalizzata all'ottenimento della migliore configurazione possibile per il sistema e sono pertanto da intendersi come "sperimentali". In molti casi, un attento studio sulla messa a punto dei valori di alcuni attributi chiave potrebbe rivelarsi molto proficua per ottenere prestazioni complessive più soddisfacenti.

È oltretutto estremamente importante comprendere che in nessun caso è possibile raggiungere una configurazione ottimale valida per qualsiasi tipologia di scenario e allo stesso tempo indipendente dalle contingenti caratteristiche di traffico, mobilità e composizione della popolazione dei nodi. Si potrebbe quindi ricercare un insieme di diversi *profili* di configurazione validi ciascuno per uno specifico caso d'uso. Più in generale, si potrebbe prevedere la realizzazione di un componente che offra funzionalità di *dynamic tuning* per ogni nodo del sistema. Tale modulo dovrebbe cioè provvedere a una costante messa a punto automatica di alcuni attributi sfruttando alcune informazioni di contesto sul sistema circostante o sullo stato attuale del nodo stesso.

Strategie proattive per la prevenzione di collisioni tra pacchetti

Nel corso del capitolo 5, nel commentare i vari risultati prodotti dal sistema, si è visto come il maggiore rischio per la comunicazione dei vari nodi presenti nell'ambiente sia dato in buona parte dall'elevata possibilità che le trame dei messaggi inviati hanno di collidere tra loro in mancanza di un adeguato controllo degli accessi al canale fisico. Nei protocolli realizzati si è sempre tenuto conto della probabilità relativamente alta che ha un messaggio di non venire recapitato, di conseguenza le logiche cercano di tollerare per quanto possibile la mancata ricezione di alcuni pacchetti.

Una funzionalità che contribuirebbe ad incrementare ulteriormente le

performance e l'affidabilità dell'architettura potrebbe essere una maggiore *context-awareness* da parte dei singoli nodi in quei momenti dove è richiesto un frequente invio di messaggi da parte di più nodi in un breve lasso di tempo, eventualità che si presenta ad esempio durante l'avvicinamento a un incrocio controllato da una RSU.

Come semplice dimostrazione di tale concetto si consideri la procedura di ispezione dei nodi descritta al paragrafo 4.3.1. Tale protocollo è estremamente importante per gestire correttamente il ciclo di vita di un plotone e permette allo stesso tempo di recuperare i dati di interesse dai vari nodi membri. È già stato sottolineato come tale procedura comporti l'invio di un pacchetto informativo da parte di ogni affiliato a un gruppo entro un intervallo di tempo stabilito in maniera dinamica. La semplice modalità casuale con la quale i singoli nodi coinvolti nella procedura scelgono il momento per l'invio del proprio messaggio non è sempre sufficiente a garantire che non si verifichino numerose cancellazioni di pacchetti dovute alle possibili collisioni con altre trasmissioni ad opera degli stessi membri o di altri nodi nelle vicinanze. Riuscire ad organizzare in maniera più efficiente la sequenza di istanti in cui i diversi partecipanti di un gruppo devono rispondere al beacon del loro leader porterebbe sicuramente ad una minore probabilità di collisione tra i pacchetti, ridurrebbe il tempo necessario per ogni probe e nel complesso renderebbe l'intero protocollo più stabile e performante.

Più in generale, per incrementare ulteriormente la qualità complessiva del sistema, andrebbe promosso ogni genere di sforzo da parte di un qualsiasi nodo per limitare o regolamentare l'invio dei propri messaggi, sfruttando a tale scopo informazioni riguardanti la propria posizione, la presenza di altri veicoli nelle vicinanze o comunque qualsiasi tipo di conoscenza del contesto nel quale esso si trova ad operare.

Posizionamento dei nodi mediante *waypoint*

Durante le fasi finali di implementazione del sistema è stata considerata una tecnica alternativa e più avanzata per poter superare il concetto di direzione e risolvere i punti deboli ad esso legati ed esposti nel corso del pa-

ragrafo precedente. Tale strategia ricorre all'utilizzo di una lista di *waypoint* per ciascuna delle direzioni di provenienza dei veicoli e consente di individuare in maniera estremamente più precisa il percorso seguito dai veicoli in avvicinamento all'RSU, superando allo stesso tempo le limitazioni dovute a percorsi stradali irregolari e direzioni parallele sovrapposte.

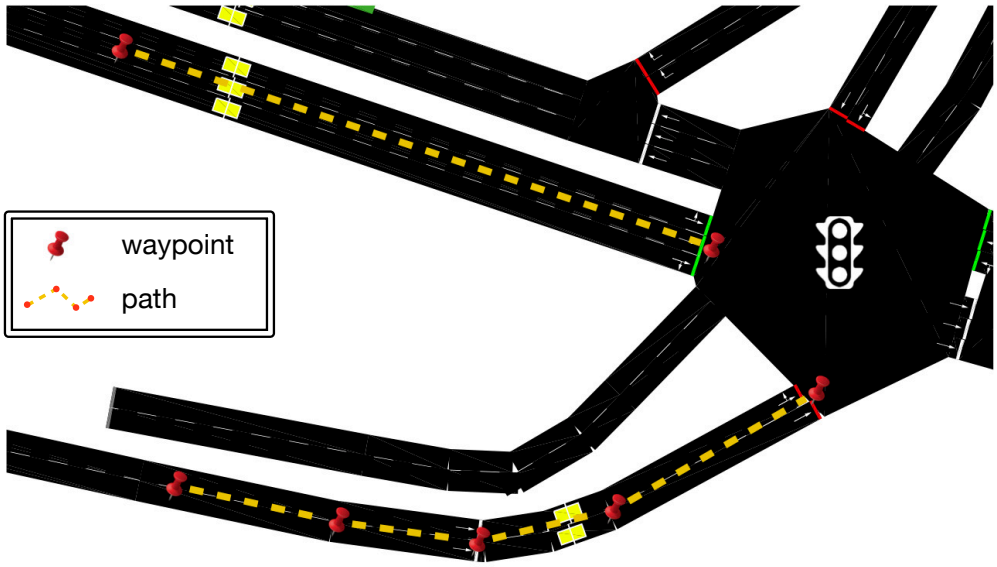

Figura 6.2: Esempio di un sistema di posizionamento dei veicoli basato su waypoint. Considerando un segmento tra ogni coppia di waypoint successivi e calcolando la distanza di un nodo da quest'ultimo è possibile determinare con una maggiore precisione l'esatto tratto stradale percorso dai vari veicoli e risolvere quelle limitazioni del sistema dovute all'uso della sola informazione di direzione.

Ogni nodo in avvicinamento all'incrocio potrebbe ricevere dalla RSU ivi posizionata una collezione di posizioni relative ai waypoint previsti per quell'area di interesse, eventualmente suddivisa in più liste in modo da poter distinguere i diversi *percorsi* che l'RSU intende monitorare, in maniera del tutto analoga a quanto viene attualmente previsto dal sistema[2] nel momento in cui le unità infrastrutturali comunicano le proprie direzioni di riferimento all'interno dei propri segnali beacon.

[2]Si consulti il paragrafo 4.1.

Una volta ricevuto l'insieme dei waypoint, è possibile individuare con facilità quello più vicino alla propria posizione attuale e scoprire in questo modo l'esatto percorso che si sta seguendo e utilizzare quindi un suo indicatore univoco per organizzare la formazione di un gruppo assieme ad altri eventuali nodi nelle vicinanze. Per verificare con maggiore precisione se un veicolo sta effettivamente seguendo un particolare tragitto è possibile introdurre il concetto di *distanza* dal percorso, intesa come spazio in linea d'aria che separa il nodo dal segmento che unisce i due waypoint contigui più vicini al nodo.

Pur comportando uno sforzo implementativo superiore, questa strategia permetterebbe di capire con estrema precisione quale tragitto ogni veicolo sta seguendo durante l'avvicinamento a un punto di interesse, anche in presenza di tratti stradali complessi e curvilinei nei quali il concetto di direzione mostra inevitabilmente i propri limiti.

Comunicazioni tra gruppi adiacenti

Per come è stato definito il concetto di gruppo nell'architettura presentata, esso risulta un'entità autonoma e isolata da altri eventuali plotoni circostanti, inclusi quelli appartenenti alla stessa direzione di riferimento. Potrebbero in realtà verificarsi numerose situazioni nelle quali una forma di comunicazione tra gruppi caratterizzati da una direzione (o percorso) conformante può contribuire a migliorare considerevolmente la resa complessiva dei protocolli sviluppati.

Un'informazione importante per la stima del flusso del traffico in situazioni di congestione consiste nella lunghezza approssimata delle code (*tailback*) che i veicoli vanno formare nel momento in cui si fermano in un incrocio in attesa di poter procedere [Ble+12]. Tale parametro non è implementato nella soluzione di progetto in quanto non è possibile ottenerne una stima sufficientemente accurata servendosi delle informazioni messe a disposizione da un solo plotone.

Una prima motivazione per l'adozione di tecniche che realizzino la collaborazione tra gruppi in grado di comunicare direttamente o indirettamente

è quindi la possibilità di migliorare notevolmente la qualità dei dati già prodotti dal sistema e di sintetizzarne di nuovi. Si otterrebbe infatti la capacità di poter sfruttare un secondo livello di *data fusion* integrando le informazioni non soltanto tra i veicoli raggruppati ma anche tra gruppi di vetture presenti in un'area anche relativamente vasta.

Una possibile strategia per implementare in maniera particolarmente semplice tali funzionalità senza necessariamente ricorrere all'uso di sofisticati protocolli di routing è quella di sfruttare i nodi marker messi a disposizione dal sistema per realizzare una sorta di *backbone* per il trasporto di dati lungo un percorso stradale predefinito. Tale canale può essere sfruttato sia dalle singole RSU per la distribuzione di segnali verso veicoli e gruppi in lontananza, sia dai plotoni stessi per comunicare con altri gruppi all'interno dello stesso tratto stradale.

Conclusioni

Il lavoro di tesi presentato in questo testo ha approfondito ed esteso le conoscenze acquisite nel corso di Sistemi Mobili M e si è proposto di metterle in pratica mediante l'analisi del problema reale discusso nel capitolo 1 e la successiva sperimentazione di un insieme di tecniche e protocolli in grado di rispondere adeguatamente alle esigenze sollevate.

Il progetto intrapreso ha quindi permesso di acquisire una preziosa esperienza con alcuni importanti strumenti di simulazione open-source estremamente diffusi in ambito accademico. L'intera architettura software proposta e presentata nel capitolo 2 è stata interamente implementata e collaudata all'interno dell'ambiente di simulazione di rete offerto dal pacchetto software ns-3 con le modalità esposte nel corso dei capitoli 3 e 4. La successiva fase di collaudo ha inoltre richiesto l'assimilazione di alcune nozioni di base sull'utilizzo di SUMO come simulatore di mobilità per gli scenari urbani utilizzati.

L'insieme dei protocolli e dei componenti ottenuti è stato progettato interamente da zero, senza procedere ovvero a una semplice applicazione o estensione di soluzioni già note in letteratura. Questo da una parte ha concesso una grandissima libertà nella fase di design e sviluppo, permettendo sia di concentrarsi su gli aspetti di maggiore interesse per lo studio, sia di compiere decisioni critiche con la massima autonomia per poter giungere alla realizzazione di un sistema originale e di un punto di vista alternativo all'interno del vastissimo panorama disponibile nel campo della ricerca sulle soluzioni di *Intelligent Transportation*.

Per contro, la scelta di non appoggiarsi a metodologie già documentate e consolidate ha comportato indubbiamente un lavoro maggiore e spesso dif-

ficoltoso, considerando che si sono dovuti progettare e costruire sia i livelli di base dell'architettura quanto le logiche superiori e si è più volte dovuto misurarsi con le numerose difficoltà insite negli ambienti VANET tra cui l'elevata e spesso imprevedibile mobilità dei nodi, la comunicazione inaffidabile, le frequenti contese al canale fisico e le conseguenti collisioni tra i pacchetti trasmessi.

Una considerevole parte del lavoro svolto ha riguardato la definizione degli algoritmi necessari per il raggruppamento dei diversi veicoli. Le due diverse strategie realizzate e presentate in questo testo – una reattiva e l'altra proattiva – sono state sviluppate e studiate parallelamente in modo da poter comprendere in quali casi d'uso e sotto quali condizioni ciascuna si dimostra più idonea rispetto all'altra.

I numerosi test condotti e in particolare quelli esposti nel capitolo 5, dimostrano come, nonostante un raggruppamento esclusivamente reattivo sia in grado di portare alla formazione di plotoni di veicoli di ottima qualità e con il minimo sforzo, l'adozione di meccaniche proattive risulta essere una scelta necessaria per poter trarre beneficio di quelle funzionalità avanzate che richiedono per ogni nodo una conoscenza pregressa dei veicoli circostanti in modo da garantire al sistema un'appropriata *context-awareness* e una maggiore dinamicità.

A causa della natura prettamente sperimentale del progetto, l'architettura appena esposta presenta alcuni punti di forza ma concede abbondante spazio a ulteriori sviluppi, correzioni e approfondimenti possibili. Nel corso del capitolo 5 si sono evidenziate alcune imperfezioni insite nel protocollo di grouping proattivo e sono state fornite indicazioni su come sia possibile migliorarlo per poterlo rendere maggiormente competitivo in termini di numero di messaggi scambiati e conformazione dei gruppi in output.

Nel capitolo 6 sono state proposte numerose possibilità di espansione e ottimizzazione per il sistema complessivo, molte delle quali sono frutto dell'esperienza maturata nel corso della sua produzione e messa a punto. In particolare viene suggerita e commentata una soluzione basata su un insieme di *waypoint* per migliorare la nozione di posizione di un veicolo e superare i limiti manifestati dal concetto di direzione.

Bibliografia

[11] *NS-2 homepage*. 2011. URL: http://nsnam.isi.edu/nsnam.

[13a] *GNS3 homepage*. 2013. URL: http://www.gns3.net.

[13b] *NetSim homepage*. 2013. URL: http://www.tetcos.com/.

[13c] *NS-3 homepage*. 2013. URL: http://www.nsnam.org/.

[13d] *ns-3 Model Library*. Set. 2013. URL: http://www.nsnam.org/docs/release/3.18/models/ns-3-model-library.pdf.

[13e] *ns-3.18 Documentation*. Nov. 2013. URL: http://www.nsnam.org/docs/release/3.18/manual/ns-3-manual.pdf.

[13f] *OMNet++ homepage*. 2013. URL: http://www.omnetpp.org.

[13g] *OPNET homepage*. 2013. URL: http://www.opnet.com/.

[14a] *COLOMBO Project homepage*. 2014. URL: http://www.colombo-fp7.eu/.

[14b] *iTetris homepage*. 2014. URL: http://www.ict-itetris.eu/.

[ASP08] M. Azarmi, M. Sabaei e H. Pedram. «Adaptive routing protocols for vehicular ad hoc networks». In: *Telecommunications, 2008. IST 2008. International Symposium on*. 2008, pp. 825–830. DOI: 10.1109/ISTEL.2008.4651414.

[Ble+12] O. Bley et al. «Improvement in traffic state estimation at signal controlled intersections by merging induction loop data with V2X data». In: *Archives of Transport System Telematics* Vol. 5, iss. 3 (2012), pp. 3–7.

[CSZ11] Ho Ting Cheng, Hangguan Shan e Weihua Zhuang. «Infotainment and road safety service support in vehicular networking: From a communication perspective». In: *Mechanical Systems and Signal Processing* 25.6 (2011), pp. 2020–2038. ISSN: 0888-3270. DOI: http://dx.doi.org/10.1016/j.ymssp.2010.11.009.

[Dan03] Peter H. Dana. *Geodetic Datum Overview*. Feb. 2003. URL: http://www.colorado.edu/geography/gcraft/notes/datum/datum_f.html.

[ETS13] ETSI EN 302 637-2. *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications*. A cura di Martin Arndt. Version 1.3.0. Sophia Antipolis, France: ETSI ITS WG2, 2013.

[FLK11] Nour-Eddin El Faouzi, Henry Leung e Ajeesh Kurian. «Data fusion in intelligent transportation systems: Progress and challenges – A survey». In: *Information Fusion* 12.1 (2011). Special Issue on Intelligent Transportation Systems, pp. 4–10. ISSN: 1566-2535. DOI: http://dx.doi.org/10.1016/j.inffus.2010.06.001.

[GSL07] S. Guberinic, G. Senborn e B. Lazic. *Optimal Traffic Control: Urban Intersections*. Taylor & Francis, 2007. ISBN: 9781420062700.

[HFB09] J. Harri, F. Filali e C. Bonnet. «Mobility models for vehicular ad hoc networks: a survey and taxonomy». In: *Communications Surveys Tutorials, IEEE* 11.4 (2009), pp. 19–41. ISSN: 1553-877X. DOI: 10.1109/SURV.2009.090403.

[HL01] David L. Hall e James Llinas. *Handbook of Multisensor Data Fusion*. CRC Press, 202001. ISBN: 0849323797.

[HLE10] Hannes Hartenstein, K. Laberteaux e Inc Ebrary. *VANET: vehicular applications and inter-networking technologies*. Wiley Online Library, 2010.

[HXG02] Xiaoyan Hong, Kaixin Xu e M. Gerla. «Scalable routing protocols for mobile ad hoc networks». In: *Network, IEEE* 16.4 (2002), pp. 11–21. ISSN: 0890-8044. DOI: 10.1109/MNET.2002.1020231.

[Jur12] Ronald K. Jurgen. *V2V/V2I Communications for Improved Road Safety and Efficiency*. Warrendale, PA: SAE International, ago. 2012. ISBN: 978-0-7680-7725-4. DOI: 10.4271/pt-154.

[Kra+12] Daniel Krajzewicz et al. «Recent Development and Applications of SUMO - Simulation of Urban MObility». In: *International Journal On Advances in Systems and Measurements*. Internatio-

nal Journal On Advances in Systems and Measurements 5.3&4 (dic. 2012), pp. 128–138. URL: http://elib.dlr.de/80483/.

[Kra+13] Daniel Krajzewicz et al. «COLOMBO: Investigating the Potential of V2X for traffic management purposes assuming low penetration rates». In: *9th ITS European Congress*. Giu. 2013.

[LH06] Mathieu Lacage e Thomas R. Henderson. «Yet Another Network Simulator». In: *Proceeding from the 2006 Workshop on Ns-2: The IP Network Simulator*. WNS2 '06. Pisa, Italy: ACM, 2006. ISBN: 1-59593-508-8. DOI: 10.1145/1190455.1190467.

[PG11] Eldad Perahia e Michelle X. Gong. «Gigabit Wireless LANs: An Overview of IEEE 802.11Ac and 802.11Ad». In: *SIGMOBILE Mob. Comput. Commun. Rev.* 15.3 (nov. 2011), pp. 23–33. ISSN: 1559-1662. DOI: 10.1145/2073290.2073294. URL: http://doi.acm.org/10.1145/2073290.2073294.

[Ran94] J. Rankin. «An error model for sensor simulation GPS and differential GPS». In: *Position Location and Navigation Symposium, 1994., IEEE*. 1994, pp. 260–266. DOI: 10.1109/PLANS.1994.303322.

[Sad11] Sadrozinski. *Applications of field-programmable gate arrays in scientific research*. Boca Raton, FL: CRC Press, 2011.

[SK10] J. Sarik e I. Kymissis. «Lab kits using the Arduino prototyping platform». In: *Frontiers in Education Conference (FIE), 2010 IEEE*. 2010. DOI: 10.1109/FIE.2010.5673417.

[Tan02] Andrew Tanenbaum. *Computer Networks*. 4th. Prentice Hall, 2002. ISBN: 0130661023.

[Tor01] W. Torge. *Geodesy*. De Gruyter, 2001. ISBN: 9783110879957. URL: http://books.google.it/books?id=IFjT5PuX3tYC.

[VBK12] S. Vodopivec, J. Bester e A. Kos. «A survey on clustering algorithms for vehicular ad-hoc networks». In: *Telecommunications and Signal Processing (TSP), 2012 35th International Conference on*. 2012, pp. 52–56. DOI: 10.1109/TSP.2012.6256251.