Alma Mater Studiorum · Università di Bologna

FACOLTÀ DI SCIENZE MATEMATICHE, FISICHE E NATURALI
Corso di Laurea Magistrale in Informatica

PROGETTAZIONE, MODELLAZIONE, ANALISI E SVILUPPO DI PROTOCOLLI MAC IN AMBITO VEICOLARE

Tesi di Laurea in Sistemi e Reti Wireless

Relatore:
Chiar.mo Prof.
Luciano Bononi

Correlatore:
Dott.
Marco Di Felice

Presentata da:
Luca Bedogni

Sessione II
Anno Accademico 2010-2011

A Serena, Stefania, Claudio

I computer sono incredibilmente veloci, accurati e stupidi.
Gli uomini sono incredibilmente lenti, inaccurati e intelligenti.
Insieme sono una potenza che supera l'immaginazione.

A. Einstein

Introduzione

Con questo documento viene presentato il lavoro svolto per la progettazione, la realizzazione e la valutazione di un protocollo per la clusterizzazione di veicoli, volto alla trasmissione di messaggi in modo rapido in ambito veicolare.

Il protocollo è chiamato DBA-MAC, acronimo di *Dynamic Backbone Assisted MAC*, e si basa sulla costruzione di una backbone virtuale tra veicoli. In questo documento vengono analizzate le *VANET*, acronimo di *Vehicular AdHoc Network*, reti wireless AdHoc volte alla comunicazione in ambito veicolare, e ne vengono spiegati i principali problemi e le soluzioni proposte. In particolare, si propone un protocollo basato sulla clusterizzazione dei veicoli in una backbone, e se ne analizzano pregi e difetti, comparandolo anche ad altre soluzioni proposte in letteratura. La comunicazione in ambito veicolari presenta diversi problemi, primo fra tutti il *broadcast storm*, ovvero le molteplici collisioni che si presentano se diversi veicoli tentano di comunicare in broadcast nello stesso istante.

Il protocollo è stato inizialmente proposto in [4], ed è stato da me ulteriormente esteso per permetterne il funzionamento anche in scenari urbani, oltre che migliorarne le performance applicando diverse modifiche alla soluzione originaria.

La ricerca in ambito stradale è attualmente fiorente, sia per quanto riguarda lo studio di soluzioni per la sicurezza stradale che per la navigazione assistita. La costruzione e lo sviluppo di applicazioni che permettano un transito più sicuro per persone e mezzi garantirebbe un abbassamento degli incidenti stradali, anche di quelli mortali.

Il testo è organizzato come segue: dopo una spiegazione generale di cosa sono

le reti veicolari nel capitolo 1, viene presentato il problema della trasmissione broadcast multihop nel capitolo 2, passando poi ad illustrare il protocollo vero e proprio nel capitolo 3, finendo con la presentazione e la discussione dei risultati ottenuti dalle simulazioni nel capitolo 4.

Indice

Elenco delle figure

Elenco delle tabelle

Capitolo 1

VANETs

In questo capitolo vengono presentate le VANET (Vehicular Ad-Hoc Network), unitamente alle loro possibili applicazioni e alle problematiche di ricerca che presentano.

1.1 Cosa sono

Le VANETs, acronimo di *Vehicular Ad-Hoc Networks*, sono reti veicolari ad-hoc, che permettono ai veicoli di comunicare tra di loro. Sono un ambito di ricerca relativamente nuovo, che si spera possa permettere in un prossimo futuro di avere tecnologie di supporto alla guida.

Le VANET, per funzionare, hanno bisogno di diversi veicoli dotati di *OBU* (On Board Unit), ovvero dispositivi wireless montati sui veicoli, ed eventualmente alcune *RSU* (Road Side Unit), ovvero infrastrutture statiche che permettono ad esempio il collegamento a Internet. Gli *RSU* possono essere montati in posti strategici, come ad esempio incroci, parcheggi, stazioni di servizio, gallerie.

Esistono diversi tipi di VANET, che possono essere riassunti come segue:

- **IVC**: acronimo di *Inter-vehicle Communication*. Con questa sigla si indicano tutti i protocolli e le applicazioni che prevedono che i veicoli possano comunicare tra di loro senza una infrastruttura di base.

- **I2V**: acronimo di *Infrastructure to Vehicle*. Con questa sigla si indicano tutti i protocolli e le applicazioni che prevedono che i veicoli comunichino con un'infrastruttura di base, e che mandino e ricevano informazioni da questa entità.

Chiaramente la tipologia **IVC** è potenzialmente più semplice da realizzare dal punto di vista pratico, poichè non occorre piazzare nessuna struttura statica, e tutto è demandato ai singoli veicoli. Questo però rende le scelte progettuali più complesse, poichè si tratta di un'architettura completamente distribuita, con alta mobilità e di conseguenza caratteristiche che possono cambiare molto rapidamente nel tempo.

Al contrario, la tipologia **I2V** richiede il piazzamento di strutture esterne ai veicoli, o fissi in posti strategici, come ad esempio incroci, semafori, oppure anche semi-fisse. Con quest'ultima categorizzazione mi riferisco a dispositivi che possono essere posizionati per segnalare condizioni temporanee, come ad esempio il restringimento di corsie dovuto a lavori in corso, condizioni anormali dell'asfalto, strade chiuse ecc.

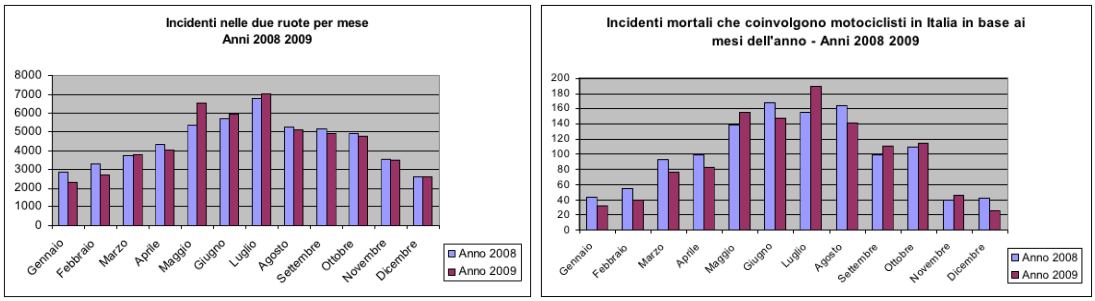
Chiaramente queste due tipologie non sono completamente separate, ma possono essere impiegate entrambe facendo comunicare i veicoli tra di loro e con una infrastruttura di rete. Si può ad esempio pensare a veicoli che raccolgono informazioni sul traffico in tempo reale, e poi le comunicano tramite internet ad un server centrale che si preoccupa di informare gli altri veicoli.

1.2 Perchè sono necessarie le VANET

Solamente in America, nel 2004, le statistiche dell'ufficio dei trasporti statunitensi afferma che ci sono più di 6,4 milioni di autostrade, con più di 243 milioni di veicoli che le percorrono.

La sicurezza sulle strade diventa così di primario interesse, poichè sempre nello stesso anno ci sono stati circa 6,2 milioni di incidenti, con 2,8 milioni di traumi e 42.000 morti. Leggendo questi numeri, si capisce che strumenti di guida

assistita possono essere cruciali nel prevenire e abbassare il numero e l'entità degli incidenti.



(a) (b)

Figura 1.1: Grafico che mostra gli incidenti motociclistici in Italia. Nella figura (a) sono riassunti gli incidenti, nella figura (b) gli incidenti mortali [11].

Le figure 1.1(a) e 1.1(b) mostrano come in Italia, solo per le moto, il numero di incidenti stradali sia molto elevato, con un picco nei mesi estivi.

Anno	Tutti i veicoli		
	Incidenti	Morti	Feriti
2008	218.963	4.725	310.745
2009	215.405	4.237	307.258

Tabella 1.1: Tabella che mostra il numero di incidenti, di feriti e di decessi in Italia nel 2008 e nel 2009 [11].

La tabella 1.1 mostra il numero di incidenti, sia di autoveicoli che di motocicli, negli anni 2008 e 2009. Ci sono stati circa 217.000 incidenti di media nei due anni, in hanno perso la vita circa 4.500 persone all'anno, che equivalgono a un decesso ogni due ore per ogni giorno dell'anno. Sono numeri che spaventano, e che fanno capire come un sistema per prevenire queste fatalità possa contribuire in modo significativo al miglioramento delle condizioni di guida e della sicurezza su strada. C'è comunque da registrare un lieve miglioramento in questi dati, poichè dal 2008 al 2009 cè stata una diminuzione degli incidenti nella misura dell'1.6%, una diminuzione dei decessi del 10.3% e una diminuzione dei feriti dell'1.1%

[11].
La sicurezza stradale chiaramente è un problema sentito anche a livello europeo, che nel libro bianco del 13 settembre 2011 fissava come obiettivo, per i paesi europei, la riduzione della mortalità del 50%. In generali i paesi europei hanno avuto una buona diminuzione del tasso di mortalità, anche se non tutti i paesi sono riusciti a raggiungere l'obiettivo prefissato, pur andandoci vicino in molti casi.

Le VANET in questo senso possono aiutare, fornendo strumenti utili per rendere il tragitto più sicuro, fornendo all'autista informazioni sul percorso e allertandolo su possibili situazioni pericolose. Oltre a far comunicare tra di loro i veicoli, potrebbero anche essere previste delle vere e proprie reti parallele, ad esempio in autostrada, che disseminino i messaggi e informino gli autisti di traffico, code, incidenti. Si pensi ad esempio quando in autostrada si presenta una coda: è abitudine comune segnalare ai veicoli che ci seguono, mediante l'utilizzo delle quattro frecce, che cè un blocco del traffico e di prestare quindi attenzione, frenando. Si pensi ora a un sistema che non ha bisogno dell'intervento dell'uomo, ma che riconosce autonomamente una situazione anomala, come appunto una coda in autostrada, e lo segnala ai veicoli che lo precedono, dando tutto il tempo agli autisti di prendere le norme di sicurezza idonee. Peraltro, in [11] si nota come la distrazione e le manovre errate siano la causa principale di sinistri.

La comunicazione tra veicoli può essere realizzata in modo più semplice dal punto di vista dei dispositivi, poichè è sufficiente dotare ogni veicolo di un dispositivo wireless che permetta la comunicazione, mentre invece creare vere e proprie infrastrutture wireless statiche è sicuramente più dispendioso e complesso. Una cosa non esclude l'altra, ma possono coesistere per formare reti più solide, offrendo magari un numero di servizi limitato nelle aree dove non cè copertura totale e un servizio migliore dove invece è prevista anche una infrastruttura.

1.3 Applicazioni

Le VANET possono avere molteplici applicazioni, per garantire e migliorare la sicurezza stradale o l'efficienza con la quale si percorrono i percorsi.

Come proposto in [31], le applicazioni che si possono pensare per le VANET sono:

- **Informazioni di carattere generale non critico**: Servizi per i quali la perdita di messaggi, o il loro arrivo ritardato, non costituiscono un problema. Esempi di queste applicazioni possono essere:
  - Informazioni sul tempo atmosferico
  - Informazioni sulle traffico della strade
  - Pubblicità
  - Intrattenimento
- **Informazioni di sicurezza**: Servizi per i quali la perdita di messaggi, o il loro arrivo ritardato, può compromettere la sicurezza della guida o può rendere inutile l'applicazione. Esempi di queste applicazioni possono essere:
  - Condizioni dell'asfalto
  - Comportamento anomalo di veicoli
- **Controllo del tragitto individuale**: Servizi che forniscono messaggi all'autista per migliorare la sicurezza della guida. Esempi di queste applicazioni possono essere:
  - Informazioni su veicoli fermi
  - Informazioni su lavori in corso
- **Controllo del tragitto di gruppo**: Servizi che cercano di regolare il flusso generale dei veicoli al fine di migliorare per tutti le condizioni di sicurezza e velocità di guida.

1.3.1 Applicazioni per la sicurezza

Per quanto riguarda la sicurezza, le applicazioni per le VANET devono garantire una tempestiva disseminazione dei messaggi di allerta che provengono da altri veicoli o da infrastrutture.

Possiamo pensare a diversi scenari possibili:

- In una giornata con visibilità scarsa, abbiamo chiaramente difficoltà a vedere i veicoli che ci precedono. Se uno di questi dovesse improvvisamente frenare, potremmo non fare in tempo ad accorgercene e quindi ad agire di conseguenza. In questo caso, se il veicolo che ci precede ci comunicasse che ha dovuto effettuare una brusca frenata, noi che lo seguiamo potremmo esserne notificati e quindi prestare maggiore attenzione al tratto di strada che dobbiamo percorrere.

- Se una strada è in condizioni non ottimali, presentando buche e asfalto irregolare, si potrebbe pensare a un sistema statico (RSU) che comunichi questo a tutti i veicoli che stanno per affrontare quel tratto di strada, di modo che i conducenti siano informati della condizioni anomale del percorso.

1.3.2 Applicazioni per la navigazione assistita

Sono tutte quelle applicazioni che permettono un tragitto più efficiente, veloce e con meno rallentamenti. In questo caso i messaggi non hanno requisiti di consegna così rigidi come nel caso della sicurezza, poichè la perdita di alcune informazioni non compromette la sicurezza e non rende l'applicazione inutilizzabile, ma solo meno efficiente.

Anche per questa tipologia di applicazioni, possiamo immaginarci alcuni scenari:

- Abbiamo impostato sul nostro navigatore la destinazione desiderata. Il navigatore calcola la strada più veloce, e noi cominciamo il tragitto. A un certo punto, avviene un incidente proprio in un tratto che dobbiamo percorrere, e la strada si blocca provocando code. Il nostro navigatore ricalcola una nuova strada che escluda quel percorso, e aggiorna quindi il tragitto da percorrere, riducendo il tempo di percorrenza. Allo stesso modo, tratti esclusi per lo stesso motivo possono essere reinseriti nel tragitto se al navigatore arrivasse l'informazione che una coda è terminata e quindi il tratto interessato è nuovamente libero.

- Possiamo supporre che ogni strada, a seconda della sua larghezza, velocità e lunghezza, abbia una certa capacità di veicoli. 100 veicoli in 1km di autostrada sono pochi, ma lo stesso numero in una stradina di campagna possono farla congestionare rapidamente. Anche in questo caso, i navigatori dei vari veicoli potrebbero accordarsi, cercando di bilanciare il traffico secondo la capienza di ogni strada, per sfruttarle al meglio e provocare meno code.

Queste ultime applicazioni affrontano anche un altro problema, che magari può sembrare scontato o passare in secondo piano, ovvero l'inquinamento atmosferico. Avere dei sistemi che rendono il traffico più scorrevole, con meno veicoli fermi e quindi tragitti in media più brevi, porta anche a un consumo di benzina inferiori e quindi a rilasci di CO_2 minori.

1.4 Progetti

Esistono molti progetti in giro per il mondo che cercano di studiare le VANET e le loro applicazioni. Alcuni si focalizzano sulla standardizzazione delle comunicazioni, altri cercano di mettere a disposizione dei ricercatori strumenti sempre migliori e il più realistici possibile.

802.11p

L'802.11p[1] è un'aggiunta al protocollo 802.11 comunemente conosciuto, denominata *WAVE (Wireless Access in Vehicular Environment)*, che cerca di standardizzare le comunicazioni tra veicoli del protocollo 802.11. Comprende alcune modifiche e aggiunte al normale protocollo 802.11, per supportare lo sviluppo di applicazioni *ITS* e la comunicazione tra veicoli, oltre alla possibilità di comunicare anche con infrastrutture poste ai lati della strada, nella banda tra 5.85 GHz e 5.925 GHz.

Il Task group che doveva completare questo standard ha concluso il lavoro, e che è stato ufficialmente riconosciuto e pubblicato il 15 Luglio 2010[1].

DSRC

Il *Dedicated Short Range Communications (DSRC)* [2] fa comunicare i veicoli con particolari infrastrutture piazzate in punti strategici, come ad esempio piazze, stazioni di servizio ecc. Sono comunicazioni di soli dati, operanti nello spettro tra 5,725 MHz e 5,875 MHz, e consistono di *Road side units (RSU)* e *On Board Units (OBU)*. Lo standard specifica le frequenze su cui operare, ma lascia comunque aperta la possibilità di utilizzare altre frequenze, se le leggi locali lo permettono.

La scelta dello spettro di frequenze di 5 GHz è stata fatta per le caratteristiche di propagazione eccellenti per l'ambito veicolari, fino a 1000 m in condizioni ideali.

Purtroppo al momento il più grosso freno a questa tecnologia è la mancanza di standardizzazione, anche all'interno dello stesso protocollo. Difatti, i sistemi DSRC presenti in Europa, Stati Uniti d'America e Giappone non sono al momento compatibili tra di loro.

Car-to-Car Communication Consortium

L'obiettivo del *Car-to-Car communication Consortium* [3] è lo sviluppo e il rilascio di uno standard europeo per la comunicazione nell'ambito dei sistemi di trasporto intelligenti, focalizzato sui sistemi di comunicazione tra veicoli.

Il progetto è già attivo da diversi anni, e continua la ricerca e il lavoro al fine di:

- Sviluppare e rilasciare uno standard europeo per i sistemi *ITS*, in particolare applicazioni *IVC*

[1] IEEE 802.11 Timelines - http://grouper.ieee.org/groups/802/11/Reports/802.11_Timelines.htm
[2] DSRC Home, http://www.etsi.org/WebSite/technologies/DSRC.aspx
[3] Car-to-Car communication consortium http://www.car-2-car.org/

- Essere un attore principale nello sviluppo di uno standard europeo per applicazioni I2V interoperabile con applicazioni IVC
- Allocare una banda di frequenza libera per applicazioni veicolari

Intelligent Transportation System

L'*Intelligent Transportation System*[4], risposta giapponese per lo sviluppo di applicazioni veicolari wireless intelligenti, è stato fondato nel 1994 con il nome di *VERTIS*.

Non è da vedere come un progetto asiatico a se, quanto a un ulteriore centro di ricerca e di conoscenza, che collabora e va avanti assieme ai corrispettivi europei e americani.

Tra le altre cose, organizza anche l'*ITS World Congress*, un forum internazionale per promuovere la cooperazione e lo stato dell'arte nel campo dell'ITS. È annuale, e viene tenuto ogni anno in un posto diverso, in Europa, Asia o Nord America.

CVIS

Il progetto CVIS[5] cerca di capire come connettere i veicoli tra di loro e con le attuali infrastrutture, mediante quindi l'utilizzo di reti wireless eterogenee.

Uno degli obiettivi è quello di produrre un terminale multi canale che permetta di mantenere una connessione Internet stabile, passando su molteplici sistemi di comunicazione come le reti cellulari, Wi-Fi, infrarossi e altre, garantendo la massima interoperabilità tra di esse in modo trasparente.

Si ritiene che questo sia un progetto importante, poichè non impone nessuna regola, ma cerca di ottenere il meglio da ciò che esiste già. L'interoperabilità tra le tecnologie esistenti è fondamentale, poichè il rischio che si sviluppino diversi standard è alto, e quindi avere un modo per farli parlare tra di loro è cruciale.

Infatti se un'azienda automobilistica A abbraccia la tecnologia X, e un'azienda automobilistica B abbracciasse invece la Y, avremmo bisogno di dispositivi

[4]ITS Consortium http://www.its-jp.org/english

[5]CVIS Home Page - http://www.cvisproject.org/

che riescano a comunicare sia con i veicoli di A che con i veicoli di B. Trovare metodologie per la comunicazione tra protocolli diversi garantisce quindi una maggiore libertà ai costruttori, che possono scegliere liberamente cosa installare sulle proprie autovetture senza aver paura di essere tagliati fuori dalla comunicazione.

Il progetto CVIS ha 3 sotto-progetti dedicati alla realizzazione di applicazioni in ambito veicolare, chiamati *Cooperative Inter-Urban Applications* (CINT), *Cooperative Urban Applications* (CURB) e *Cooperative Fleet and Freight Applications* (CF&F).

Il progetto **CURB** sviluppa servizi cooperativi per migliorare l'utilizzo delle strade urbane. Lo scambio di messaggi tramite la comunicazione intraveicolare permetterà un utilizzo più efficiente della rete stradale, con effetti positivi sia per i guidatori sia per l'ambiente. Le 4 principali applicazioni sviluppate sono:

- Supporto al tragitto, dando informazioni al guidatore sul traffico presente nell'area e consigliandolo sulle possibili alternative.
- Creazione di onde verdi consigliando i veicoli sulla velocità da tenere, migliorando e velocizzando l'andamento del traffico.
- L'utilizzo di corsie di autobus per i veicoli equipaggiati con dispositivi CVIS, qualora non siano presenti autobus e questo non comprometta le funzionalità del trasporto pubblico.

Il progetto **CINT** sviluppa servizi per migliorare l'efficienza, la sicurezza e il rispetto dell'ambiente nel traffico urbano. Alcune delle tecnologie utilizzate vanno dal posizionamento avanzato dei veicoli tramite la geo-localizzazione, comunicazioni intraveicolari e ad infrastrutture. Le applicazioni che sviluppa sono principalmente due, ovvero *EDA* (Enhanced Driver Awarness) e *CTA* (Cooperative Travelers Assistance).

EDA, illustrato nella figura 1.2, è maggiormente incentrata sulla sicurezza, e informa i conducenti entro 5 secondi dalla comunicazione di potenziali pericoli rilevati sul tratto stradale, come ad esempio velocità elevate di altri veicoli, condizioni della strada e del tempo.

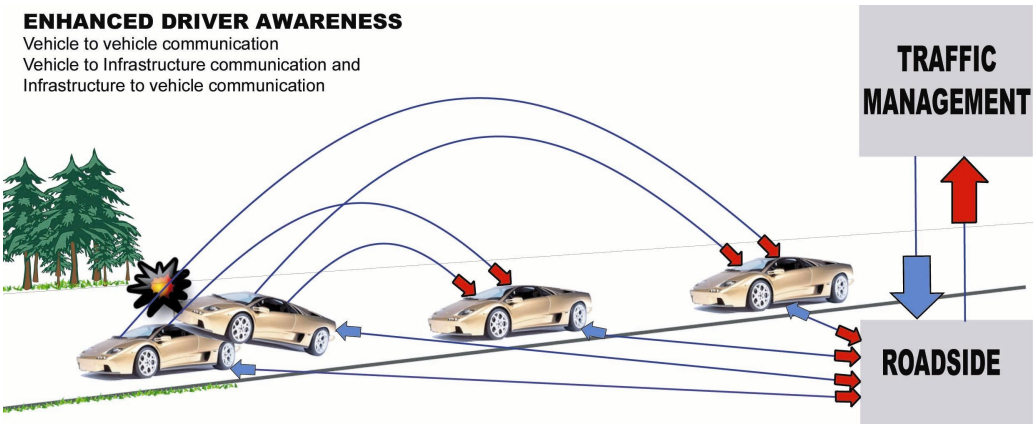


Figura 1.2: Il progetto EDA

CTA, che si può vedere schematizzato nella figura 1.3, si concentra invece sull'assistenza per i conducenti, informandoli sulle condizioni del traffico lungo la strada personalizzando le informazioni a seconda del guidatore. Questo sistema informa i conducenti entro 15 secondi su una congestione dovuta ad esempio a un incidente, ed entro altri 15 secondi propone una strada alternativa che eviti l'ingorgo.

Il progetto **CF&F** si preoccupa di fornire strumenti utili per la navigazione commerciale dei mezzi, privilegiando quindi trasporti su strada, e fornendo quindi informazioni per migliorare l'efficienza dei trasporti e riducendo l'impatto ambientale. Si monitorano anche il trasporto e la consegna di beni ritenuti pericolosi, quali infiammabili, tossici ed esplosivi. L'autista ha informazioni sulle strade migliori per trasportare le tipologie di prodotti che ha nel camion, tenendo presente della natura dei beni e le condizioni delle strade.

iTetris

Come e in quanto si può stimare la spesa sostenuta per il traffico? La commissione europea ha stimato che gli ingorghi e il traffico stradale costano circa 50 miliardi di euro all'anno, e si prevede che cresca molto rapidamente. Le tecnologie wireless sono quindi un ottimo modo per cercare di migliorare la qualità dei tragitti stradali senza dover rivoluzionare le infrastrutture, costruendo o demolen-

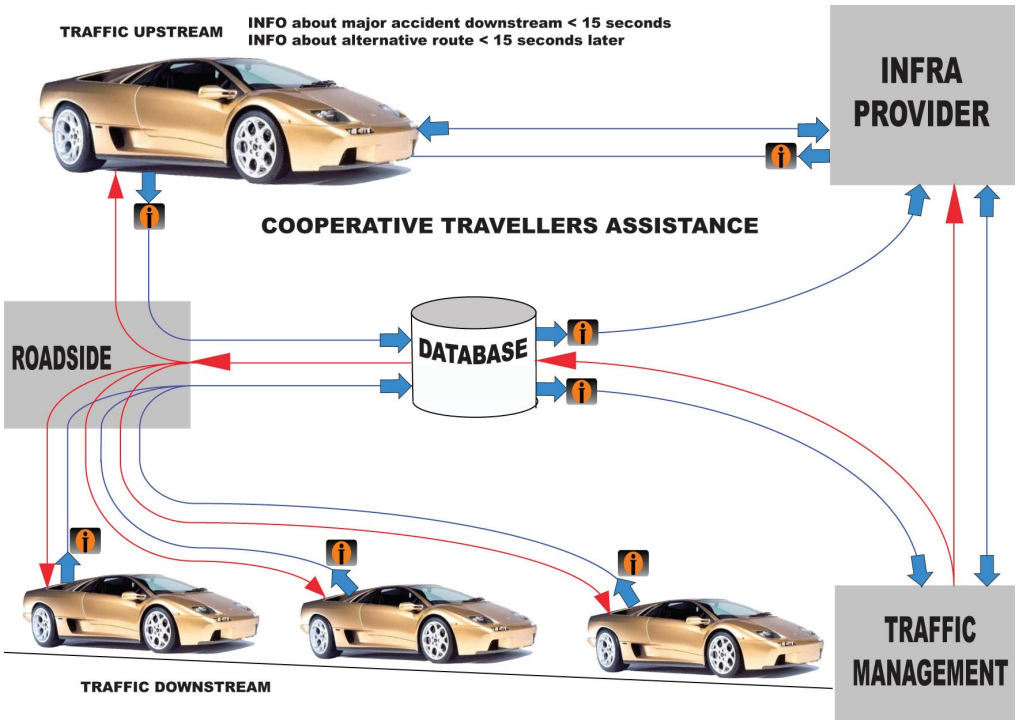


Figura 1.3: Il progetto CTA

do tratti di strade. Chiaramente, prima che le istituzioni decidano di adottare una o l'altra tecnologia, devono avere un prova tangibile che l'adottare dei nuovi sistemi porti a dei concreti vantaggi per la circolazione di persone e mezzi.

È in questa problematica che si inserisce il progetto iTETRIS[6], che cerca di sviluppare una piattaforma di simulazione flessibile che permetta di progettare e valutare le varie proposte. iTETRIS mette inoltre a disposizione dati con elevato dettaglio, che permettono quindi di valutare i vari protocolli e tecnologie in simulazioni con scenari molto simili a quelli reali, che possono quindi produrre dati comparabili con quelli che si otterrebbero in test direttamente sul campo.

Avere strumenti aperti e condivisi permette un'analisi trasparente dei protocolli, e una loro valutazione in scenari realmente esistenti, con tracce di mobilità ricavate da analisi sul territorio. Simili tool sono fantastici per provare nuove soluzioni, poichè riducono le assunzioni che si fanno sugli scenari, progredendo

[6]iTETRIS - http://ict-itetris.eu

verso quello che si può realmente trovare nelle città.

SAFESPOT

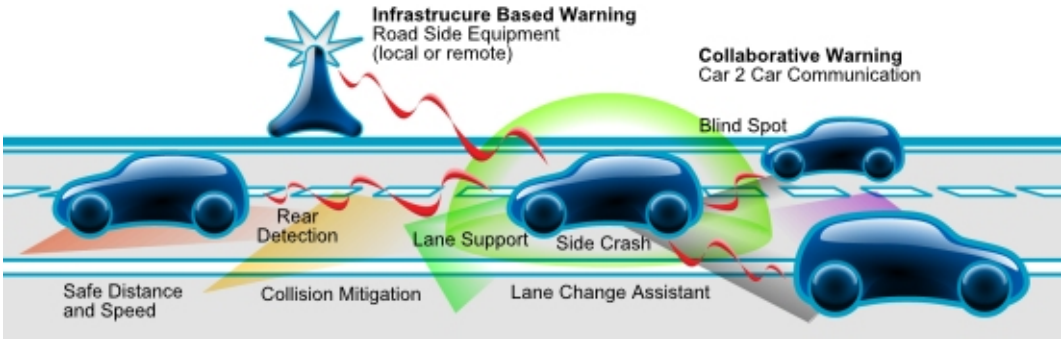


Figura 1.4: Immagine del progetto SAFESPOT che illustra le variabili prese in considerazione dal sistema

SAFESPOT crea reti dinamiche cooperative dove i veicoli (OBU) e le infrastrutture stradali (RSU) comunicano per condividere informazioni sulla presenza e la posizione di veicoli sulle strade. Cerca quindi di prevenire gli incidenti informando gli autisti di situazioni potenzialmente pericolose, come ad esempio la copertura dei cosiddetti punti ciechi, o l'informare se i veicoli stanno cambiando di corsia.

SAFESPOT è un progetto cofinanziato dalla commissione europea per le tecnologie di informazione.

Alcune delle applicazioni che il progetto ha portato a termine sono:

- **Notifica dei semafori rossi**: il veicolo viene messo a conoscenza che sta avvicinandosi a un semaforo rosso, e il guidatore può quindi conoscere lo stato dell'incrocio in anticipo, frenando e fermandosi aspettando il verde.

- **Notifica di traffico agli incroci**: quando il veicolo si avvicina a incroci con elevato traffico proveniente da altre direzioni, il guidatore è avvisato per prestare maggiore attenzione al tratto di strada che sta per percorrere. Inoltre, questo sistema funziona anche in presenza di pedoni che attraversano

la strada, poichè il guidatore ne viene informato in anticipo. Queste informazioni sono inoltre utili per i veicoli di emergenza, poichè i veicoli normali sono informati dell'avvicinarsi di un veicolo in emergenza, e possono quindi dare la precedenza anche se non hanno visto o sentito l'approcciarsi di questo pericolo.

- **Notifica di veicoli fermi**: il sistema avverte il guidatore che c'è un veicolo fermo sulla strada. Questo è molto utile se si ha davanti un camion, che impedisce di vedere oltre. Se si fosse invece informati di questa situazione, si avrebbe tutto il tempo per frenare in sicurezza ed evitare il veicolo fermo.

- **Notifica di individui**: in presenza di pedoni che attraversano la strada, magari coperti e quindi invisibili per un veicolo che si stia avvicinando all'attraversamento pedonale, il sistema ne notifica la presenza, permettendo al guidatore di fermarsi in tempo.

Tutte queste cose sono spiegate in un video presente nella homepage[7] del progetto SAFESPOT, in cui viene mostrato il sistema in funzione nelle casistiche appena citate.

Dei vari siti di test in cui il progetto SAFESPOT fa le sue prove, ve ne è anche uno presente in Italia, precisamente a Torino.

Altri progetti

Il progetto *Coopers*[8] *(CO-Operative SystEms for Intelligent Road Safety)* si focalizza sullo sviluppo di applicazioni telematiche per migliorare la collaborazione tra i veicoli, al fine di giungere a una gestione del traffico cooperativa tra veicoli e infrastrutture, per cercare di colmare la distanza che talvolta si crea tra questi due organismi.

Il progetto *eSafetySupport*[9] è un progetto della comunità europea, creato nel 2002 dopo l'obiettivo che la stessa commissione si era prefissato nel 2001, ovvero

[7] SAFESPOT - http://www.safespot-eu.org/

[8] http://www.coopers-ip.eu

[9] www.esafetysupport.org

quello di dimezzare il numero di incidenti stradali ad andare al 2010. Questo progetto coordina le industrie e la commissione al fine di promuovere tecnologie e metodi comuni per migliorare la sicurezza sulle strade.

GeoNet[10] (Geographic addressing and routing for vehicular communications) dovrebbe prendere i risultati del consorzio Car2Car, e portarli avanti implementando anche l'utilizzo di IPv6. L'obiettivo principale di GeoNet è quello di implementare e testare formalmente un meccanismo di rete che possa esistere come modulo software indipendente.

1.5 Problematiche di ricerca

La ricerca sulle VANETs è molto ampia, ma un grosso sforzo si sta facendo per rispondere alla domanda: *Come possiamo far comunicare i veicoli in modo rapido, garantendo una disseminazione veloce e affidabile dei messaggi?*

La risposta più semplice a questa domanda potrebbe essere: *Un veicolo che deve comunicare qualcosa manda il messaggio con l'informazione in broadcast, che viene poi replicata da ogni veicolo che riceve il messaggio originario*.

In questo modo, però, dobbiamo affrontare il problema del *broadcast storm* [20] [29] [28], ovvero uno scenario in cui tutti i veicoli cercano di trasmettere, e il canale wireless è saturo, generando un elevato numero di collisioni e quindi una propagazione lenta dei messaggi. Inoltre, un'applicazione di questo tipo occuperebbe per lunghissimi periodo di tempo il canale, impedendone ad altre comunicazioni l'accesso e quindi il loro funzionamento.

Occorre quindi una forma di organizzazione tra i veicoli, di modo che tutti gli interessati ricevano effettivamente il messaggio, ma che solo alcuni lo ritrasmettano proprio per evitare di cadere in queste problematiche.

In letteratura si possono trovare diverse proposte per risolvere questa problematica. Molti si basano sul clustering dei veicoli, cercando di eleggere alcuni veicoli come principali, di modo che le comunicazioni possano essere replicate solo da questi ultimi e non da tutti. In [9] [7] [13] [25] [17] [22] si cerca di risol-

[10] www.geonet-project.eu

vere il problema a livello MAC, in modo *reattivo*[5], ovvero ogni volta che viene prodotto un messaggio di allerta, il veicolo che lo deve replicare viene deciso al momento tramite biased contention [22], black-bursts [17] o bipartizione spaziale [25].

Anche se riducono l'overhead e limitano di molto il broadcast storm, i protocolli reattivi hanno bisogno di una fase di contesa per determinare il prossimo veicolo che deve replicare il messaggio, generando collisioni in scenari urbani densi dei veicoli, e aggiungendo ritardi nella consegna dei messaggi.

I protocolli proattivi invece, come ad esempio in [9], creano un'infrastruttura virtuale tra i veicoli, di modo che nel momento in cui presenta un messaggio da ritrasmettere, il veicolo designato sia gia individuato, e quindi possa ritrasmettere il messaggio in un tempo più rapido. Chiaramente questi protocolli possono essere più veloci di quelli reattivi nella ritrasmissione dei messaggi, ma introducono overhead per creare e mantenere la struttura.

1.5.1 Shadowing

Una problematica da tenere in considerazione nella progettazione di reti veicolari è lo shadowing provocato da palazzi e/o ostacoli, che rallentano o a volte addirittura impediscono la comunicazione tra veicoli, anche se vicini.

Nel progettare un protocollo in scenario urbano bisogna quindi non solamente tenere conto della posizione spaziale dei nodi, ma anche della possibilità o meno per loro di comunicare, vedendo ad esempio se il segnale è attenuato da ostacoli. In [26] gli autori fanno vedere come lo shadowing abbia effetto nella comunicazione tra veicoli, e propongono un modello per includere questo fenomeno nelle simulazioni di protocolli.

Lo shadowing può essere complesso da simulare, poichè bisogna conoscere oltre che l'infrastruttura stradale anche la posizione, la grandezza e il materiale con il quale sono costruiti gli ostacoli. Fortunatamente, alcuni progetti come ad esempio OpenStreetMap[11] tengono traccia anche dei palazzi, permettendo di importarli in strumenti di simulazione includerli e valutarli nella comunicazione.

[11] OpenStreetMap - http://www.openstreetmap.org

1.6 Protocolli cognitivi

Un'altra cosa da tenere in considerazione quando si progettano protocolli per reti veicolari è la frequenza che verrà utilizzata per le comunicazioni. Recentemente, la domanda per un numero maggiore di frequenze è cresciuta, fatto dovuto a un sempre più importante utilizzo di tecnologie wireless. Seguendo il paradigma *OSS (Opportunistic Spectrum Sharing)* [19], i dispositivi wireless hanno il permesso di utilizzare le frequenze disponibili, senza però interferire con i licenziatari di quelle bande, detti *PU (Primary Users)*.

Questa famiglia di protocolli prende il nome di protocolli cognitivi[2], ovvero protocolli che sono in gradi di utilizzare canali differenti a seconda di quelli che sono disponibili nella zona di percorrenza attuale.

Un modo per capire quali frequenze possono essere libere in una determinata zona è quello di fare *sensing* del canale. Chiaramente questo ha un costo temporale, poichè non si può capire quali frequenze sono libere in una zona non appena vi si entra. Così, in [10] gli autori propongono, oltre all'uso di tecnologie cognitive, anche una comunicazione tra i veicoli, di modo che un veicolo sappia in anticipo quali frequenze saranno libere in una certa area di strada.

Il *reinforcement learning*[27] in ambito cognitivo veicolare è fondamentale: l'alta mobilità e l'eterogeneità degli scenari impongono requisiti temporali molto limitati, dunque poter capire in anticipo quali possono essere le caratteristiche che si troveranno nel futuro è sicuramente un buon modo per migliorare le performance del protocollo, qualunque esso sia, se deve impiegare tecniche *OSS*.

1.7 Sicurezza

Un altro aspetto da non tralasciare nelle VANET è quello legato alla sicurezza e all'integrità dei messaggi.

Nell'ambito dei messaggi di allarme non è un problema se qualcuno legge un messaggio che non era inteso direttamente per lui, poichè generalmente queste

sono comunicazioni rivolte a chiunque e tendono a informare su eventi imminenti e di generale interesse.

Uno dei problemi può invece essere l'opposto, ovvero se un veicolo inizia a mandare messaggi di allerta falsi, perchè questo potrebbe portare a paralizzare il traffico o addirittura ad essere controproducente per la sicurezza dei passeggeri. Molti messaggi che segnalano ad esempio un blocco stradale, o la presenza di asfalto danneggiato, può portare i veicoli che sentono il messaggio a modificare drasticamente i propri percorsi, o a adottare misure di sicurezza non necessarie.

In [23] vengono elencati alcuni importanti punti da tenere in considerazione. In una rete veicolare, vogliamo che non ci siano veicoli che possano fingersi diverse centinaia invece che una unità, perchè vorrebbe dire dare una rappresentazione del traffico congestionato quando invece non lo è. Questo potrebbe essere risolto mediante un'autenticazione con chiave pubblica/privata presso un'entità centrale, ma i guidatori vogliono anche mantenere la privacy e l'anonimato. Infatti, molte persone probabilmente rifiuterebbero un sistema che gli permetta di avere notizie sul traffico, se questo sistema tracciasse anche continuamente la loro posizione. Occorre quindi trovare un punto d'incontro tra privacy e autenticazione, per garantire un certo livello di anonimato senza però cadere nei problemi citati in [23].

Sempre [23] introduce anche il problema del DoS in ambito veicolare. In effetti, molti dei messaggi di allerta che si possono propagare nelle VANET hanno requisiti temporali di consegna molto limitati, infatti avere la notifica di un veicolo fermo quando l'abbiamo superato è assolutamente inutile. Se quindi esistesse un veicolo che immette nel canale molti messaggi a raffica, congestionandolo, i reali messaggi di allerta potrebbero non essere trasmessi.

Stando a [12], ci sono più di 250 milioni di automobili negli Stati Uniti d'America, e ogni anno avvengono più di 5 milioni di incidenti. Viste le applicazioni proposte, che concentrano molti degli studi su soluzioni probabilistiche, la certezza nella consegna dei messaggi deve essere elevata.

Il riconoscimento di un attacco non è sufficiente[23], poichè il guidatore sarebbe

[12] U.S. transportational Statistics - http://www.bts.gov/publications/national_transportation_statistics

informato comunque di un messaggio fasullo, e il riconoscimento potrebbe avvenire troppo tardi per avere effetto. Si pensi infatti alla ricezione di un messaggio di strada intasata, che porti poi a un cambio di percorso. Il riconoscimento di un attacco qualche secondo in ritardo sarebbe inutile, poichè il cambiamento di percorso sarebbe comunque avvenuto. Quindi, il lavoro che c'è da fare è più sulla prevenzione che sul riconoscimento degli attacchi.

L'alta mobilità dei veicoli in una rete veicolare inoltre sconsiglia l'utilizzo di protocolli basati sulla reputazione, o che implichino un'intensa comunicazione tra due dispositivi. La velocità, le strade, gli incroci, sono tutte caratteristiche che portano i veicoli ad avere vicini che cambiano molto rapidamente nel tempo.

1.7.1 Avversari

Chi sono gli avversari in un contesto veicolare? Chi può volere che le informazioni non siano quelle reali, siano scarse o sovrabbondanti? In [23] ne vengono identificati 5:

- Guidatori: un guidatore che ha fretta potrebbe voler convincere i veicoli che percorrono la strada assieme a lui che più avanti cè un ingorgo, di modo che i rispettivi sistemi di navigazione propongano un'altra strada.

- Sniffer: in questa categoria ci possono essere una moltitudine di persone diverse. Passiamo dalle compagnie che vogliono tracciare i percorsi, e quindi le abitudini, dei guidatori per offrire pubblicità mirata, a rapinatori che tracciando i veicoli possono sapere quali case sono vuote.

- Pranksters: persone che vogliono giocare con questi dispositivi, e che quindi possono convincere alcuni veicoli a decelerare, altri a frenare, e alterare quindi anche pericolosamente l'andamento del traffico.

- Infiltrati: se ad esempio un'officina potesse fare gli aggiornamenti del software dei veicoli, basterebbe un meccanico corrotto per rendere inutili e pericolosi i sistemi degli altri veicoli.

- Terroristi: si potrebbero facilmente creare degli ingorghi per far detonare una bomba, o far deviare alcuni veicoli per avere una posizione migliore su potenziali bersagli.

Capitolo 2

Multihop Broadcast

In questo capitolo vengono presentati alcuni protocolli esistenti in ambito veicolare, che cercano di risolvere il problema della comunicazione broadcast multi-hop. La comunicazione broadcast multi-hop è una delle tipologie più utilizzate nell'ambito delle reti mobili ad hoc, in quanto permette di spedire messaggi ai vari nodi presenti nello scenario. La comunicazione unicast, difatti, richiederebbe di conoscere il destinatario del messaggio, cosa che in contesto veicolare a volte non è importante e altre volte è impossibile da determinare.

2.1 Protocollo di Flooding

Il *flooding* è sicuramente l'approccio più semplice per ottenere un broadcast multi-hop dei messaggi in ambito veicolare. Purtroppo però ha alcune limitazioni, come ad esempio il numero di ritrasmissioni inutili, il numero di collisioni a livello MAC e di conseguenza anche la scarsa performance nel delivery delay.

Il protocollo è molto semplice, e non richiede particolari modifiche a un ipotetico dispositivo esistente, e funziona come indicato dall'algoritmo 1.

Si vede chiaramente come il numero di ritrasmissioni tenda a crescere anche in un'area limitata. Questo avviene perchè i dispositivi continuano a ritrasmettere i messaggi anche qualora questo non fosse più necessario. Difatti, se sono presenti N veicoli in un'area, con ψ e η agli estremi, se ψ trasmette un messaggio e η lo

Algorithm 1: Algoritmo di flooding

```
Ricevo un messaggio da ritrasmettere
if non l'ho mai ritrasmesso o non ho mai sentito una sua ritrasmissione then
    lo ritrasmetto
else
    lo scarto
end if
```

riceve, avremo $N - 2$ trasmissioni inutili, ovvero tutti quei veicoli compresi tra ψ e η.

In [28] sono state fatte alcune simulazioni per vedere effettivamente quanto incida il fenomeno del broadcast storm nelle comunicazioni wireless in ambito veicolare, e i risultati sono sintetizzati dal grafico 2.1.

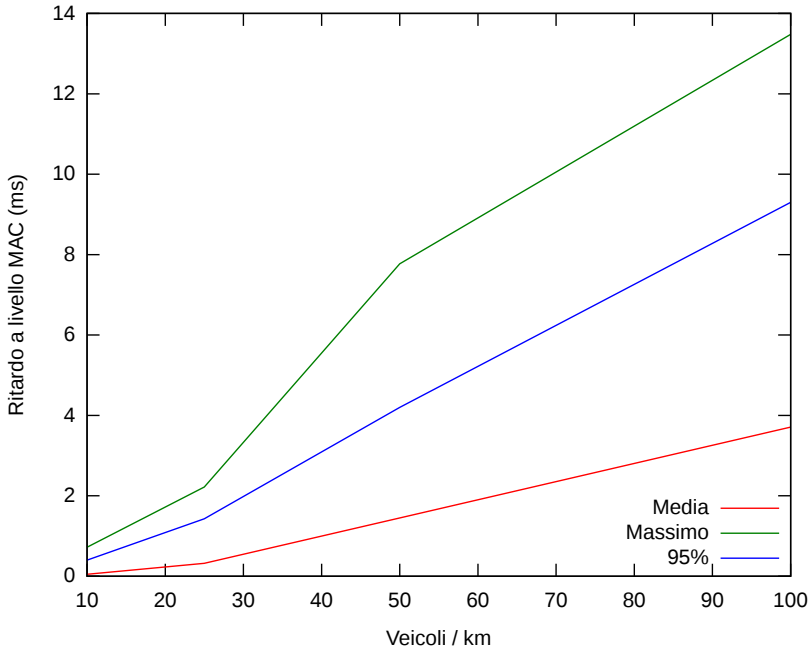


Figura 2.1: Il fenomeno del broadcast storm - il ritardo introdotto a livello MAC

È immediato notare come il crescere dei veicoli nell'area di interesse faccia anche crescere i ritardi a livello MAC.

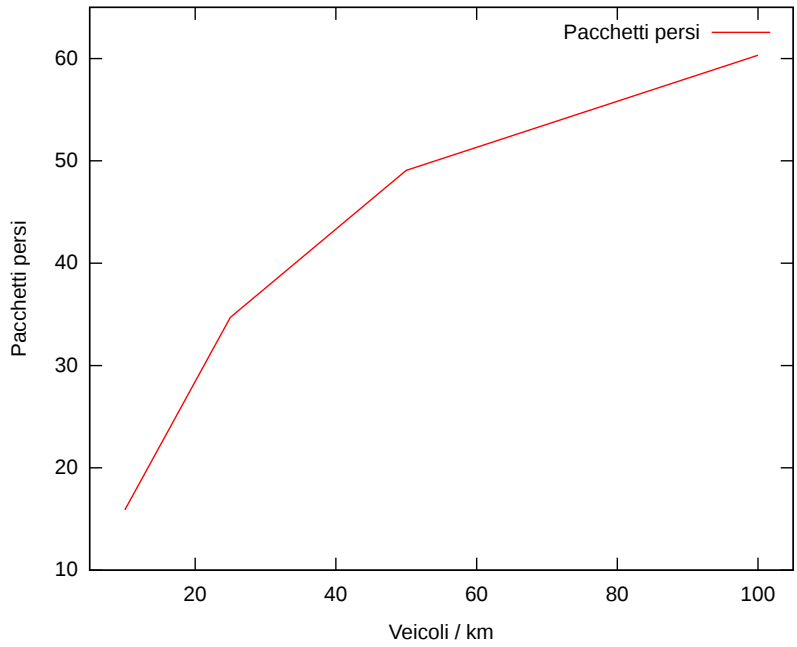


Figura 2.2: Il fenomeno del broadcast storm - la percentuale di pacchetti persi

La cosa ancora peggiore si nota dal grafico 2.2, che evidenzia come la percentuale di pacchetti persi raggiunga valori notevoli, oltre il 50% se si aumentano i veicoli. Non dovrebbe stupire un comportamento di questo tipo, ed è una delle motivazioni per la quale sono necessarie metodologie di organizzazione delle trasmissioni, per eliminare quelle non necessarie.

Con trasmissioni non necessarie intendo tutte quelle che non portano nuovi veicoli a conoscere l'informazione trasmessa, come sintetizzato dalla serie di figure seguenti: Nella figura 2.3(a) si vede un ipotetico scenario, con 12 nodi, tutti nel raggio trasmissivo degli altri. In un certo istante, il veicolo A decide di mandare in broadcast un messaggio, che quindi viene ricevuto dagli altri. Nella figura 2.3(b) si vede la situazione una volta ricevuto il messaggio dagli altri nodi: in verde c'è il veicolo A, che ha già trasmesso il messaggio e quindi resta in attesa, e in blu ci sono invece i veicoli che hanno ricevuto il messaggio ed entrano quindi in contesa per ritrasmetterlo. È chiaro che a questo punto una trasmissione di un qualsiasi veicolo blu non avrebbe nessuna utilità, poichè tutti i veicoli nel raggio

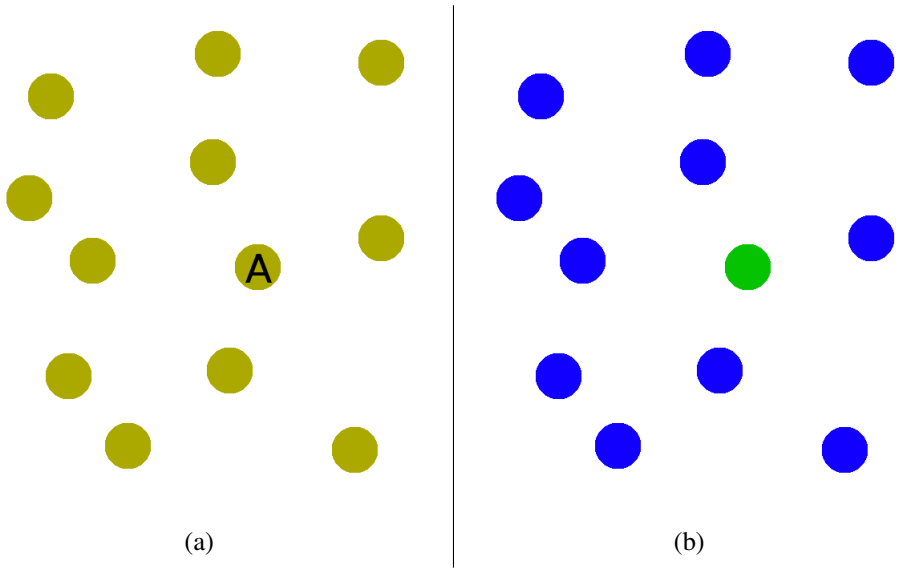


Figura 2.3: Il fenomeno del broadcast storm - graficamente

di trasmissione hanno già ricevuto l'informazione.

Vediamo quindi che la ritrasmissione del messaggio da parte di alcuni dispositivi non porta benefici, ovvero non raggiunge veicoli che non hanno ancora ricevuto l'informazione, ma crea solamente collisioni sul canale. Questo non permette a veicoli che potrebbero comunicare il messaggio a dispositivi che ancora non l'hanno ricevuto di farlo, e quindi i tempi di consegna crescono molto rapidamente.

Per cercare di ovviare a questo problema servono dei protocolli che riescano a identificare i veicoli che devono ritrasmettere il messaggio in modo che si cerchi di minimizzare le ritrasmissioni superflue, e quindi di minimizzare anche il numero di hop necessari per la consegna del messaggio.

2.2 MCDS

Una proposta di soluzione potrebbe essere quella di costruire un *MCDS (Minimum Connected Domain Set*, come proposto in [32] e [24]. Il MCDS è definito come segue:

Il MCDS di una rete è il sottoinsieme di nodi connessi fra di loro con cardinalità minima, tale per cui ogni nodo della rete è connesso con almeno un nodo del MCDS.

In effetti, in questo modo è possibile far comunicare solo i veicoli del MCDS in broadcast, e saremo certi che tutti i veicoli della rete riceveranno il messaggio.

In particolare, in [24], si propone di mantenere i messaggi in memoria finchè tutti i vicini non hanno mandato l'ACK per l'avvenuta ricezione del messaggio in broadcast. Difatti, un veicolo mantiene in memoria i messaggi di broadcast che ha mandato e la lista dei vicini che l'hanno ricevuto, di modo da non ritrasmetterlo nel caso questo non fosse necessario.

A prima vista il protocollo funziona, ed effettivamente i risultati di simulazione dimostrano che è molto affidabile e quasi tutti i nodi ricevono il messaggio di informazioni. Purtroppo il protocollo non è stato analizzato guardando le collisioni sul canale, che a mio avviso possono essere molte se la densità dei veicoli cresce. Infatti, non appena viene emesso un messaggio da un veicolo ψ, tutti i vicini di ψ devono mandare un ACK, e se l'area è densa questo potrebbe provocare diverse collisioni.

Invece in [32] gli autori fanno un'analisi teorica del limite inferiore per le comunicazioni AdHoc unilineari, partendo dal presupposto che il MCDS sia il miglior sistema per comunicare in ambito AdHoc, e derivando una serie di formule analitiche che stabiliscono il limite di performance per ogni protocollo di broadcast possibile. L'analisi è interessante, e permette di avere un riferimento per l'analisi di ogni altro protocollo che si può progettare.

Sfortunatamente, costruire un MCDS stabile in un contesto veicolare è molto arduo, poichè l'alta mobilità dei nodi ed effetti di shadowing portano la rete a cambiare molto rapidamente. Per questo ed altri motivi, in letteratura ci sono svariate proposte su come poter effettuare comunicazioni in modo rapido e affidabile in contesto veicolare. Possiamo raggruppare questi protocolli in due principali famiglie, quella dei protocolli reattivi, presentati nella sezione 2.3, e quella dei protocolli proattivi, presentati nella sezione 2.4.

2.3 Protocolli Reattivi

I protocolli *reattivi*, come anticipato, cercano di capire quale sia il veicolo che possa effettuare la miglior ritrasmissione *on-the-fly*, ovvero quando ne hanno bisogno, senza mantenere strutture virtuali. Si cerca quindi di pilotare la fase di contesa, in favore dei veicoli che possono garantire performances migliori. In particolare, in [13] e [21] si propone di modificare la contention window per favorire i veicoli più lontani, di modo da minimizzare gli hop. Il paradigma di scegliere il veicolo che deve ritrasmettere in base alla sua distanza dal nodo mittente è molto utilizzato, e in letteratura si trovano molte proposte che sfruttano questo meccanismo, in particolare [14]. In aggiunta a questo, in [16] gli autori propongono di aggiungere al meccanismo precedente anche una valutazione basata sugli angoli di percorrenza dei veicoli, per meglio scegliere *relay* che percorrano la strada nella stessa direzione del mittente. In [17] gli autori propongono di utilizzare anche i black-bursts. Infine, in [25] gli autori estendono il meccanismo di scelta dei nodi sulla base della distanza, partizionando l'area di ritrasmissione finchè non si trova il veicolo più lontano dal mittente.

In genere sono protocolli molto leggeri, che permettono di gestire reti veicolari introducendo un minimo overhead, talvolta nullo. La problematica sta nel tempo che occorre aspettare ogni volta che c'è da comunicare, poichè bisogna sempre ricalcolare il veicolo che garantisce la miglior ritrasmissione. Questo inoltre può provocare collisioni, quindi perdita di messaggi, che potrebbe quindi limitare i protocolli reattivi, sconsigliandone l'uso in applicazioni in cui ci siano requisiti *QoS* molto stringenti.

In [13] si propone *SB (Smart Broadcast)*, un protocollo progettato per richiedere una minima modifica ai già esistenti dispositivi IEEE 802.11. È pensato per scenari autostradali, e assume che la strada possa essere partizionata in settori e che ogni veicolo riesca a sapere a quale settore appartiene. La scelta del prossimo nodo che deve ritrasmettere il messaggio avviene come segue:

- Il nodo mittente trasmette un *RTB (Request-to-Broadcast)*, che contiene la propria posizione e altre informazioni.

- Solo i veicoli che percorrono la strada nella stessa direzione del mittente possono partecipare all'elezione. I veicoli scelgono un tempo di backoff casuale ma dipendente dal settore a cui appartengono, facendo in modo che i veicoli nei settori più lontani trasmettano per primi.

- Quando un nodo può trasmettere, manda un *CTB (Clear to Broadcast)*, che se ricevuto dal mittente da il permesso di trasmettere l'informazione.

Un altro contributo arriva da [22], in cui gli autori propongono un meccanismo diviso in due fasi, denominate *estimation phase* e *broadcast phase*. Nella *estimation phase* ogni veicolo cerca di conoscere il proprio raggio di trasmissione, utilizzando semplici messaggi di *hello*. Il tempo è diviso in turni, e vengono mantenute due variabili, CMFR e CMBR, che rappresentano rispettivamente la stima della distanza in avanti nella quale si riescono a sentire i veicoli (*Current-turn Maximum Front Range*) e la distanza alla quale è possibile trasmettere i messaggi all'indietro (*Current-turn Maximum Back Range*). Queste due variabili vengono aggiornate di continuo, e quando scade il turno il loro valore viene copiato in LMFR (*Latest-turn Maximum Front Range*) e LMBR (*Latest-turn Maximum Back Range*). Questo meccanismo viene meglio spiegato dagli algoritmi 2 e 3:

Algorithm 2: L'algoritmo di invio dei messaggi di hello di [22]

```
for each turn do
    sending_time = random(turn_size)
    wait(sending_time)
    if !(heard_hello_msg() or heard_collision()) then
        hello_msg.declared_max_range = max(LMFR,CMFR)
        transmit(hello_msg)
    end if
end for
```

Nella *broadcast phase* poi si usano questi valori per modificare la CW di ogni veicolo, e quindi dare priorità diverse a seconda delle caratteristiche trasmissive. In particolare, i veicoli con caratteristiche migliori avranno un CW maggiore, e di conseguenza potranno trasmettere per più tempo. Il protocollo è molto snello,

Algorithm 3: L'algoritmo di ricezione dei messaggi di hello di [22]

```
mp = my_position()
sp = hello_msg.sender_position
drm = hello_msg.declared_max_range
d = distance(mp, sp)
if received_from_front(hello_msg)) then
   CMFR = max(CMFR, d, drm)
else
   CMBR = max(CMBR, d, drm)
end if
```

ed è un qualcosa a metà tra i reattivi e i proattivi. Difatti, non mantiene una struttura virtuale per dare priorità diverse ai veicoli, ma comunque calcola dati che memorizza nel tempo per avere informazioni sui veicoli più idonei alla trasmissione dei messaggi. Le analisi dimostrano la bontà del protocollo, ma credo andrebbe provato tenendo conto anche di fenomeni come lo shadowing, che possono far variare i raggi di trasmissione dei veicoli in breve tempo e quindi potrebbero creare problemi al protocollo.

2.4 Protocolli Proattivi

I protocolli *proattivi* invece cercano di creare e mantenere una struttura virtuale, di modo che in presenza di un messaggio da ritrasmettere non si debba stabilire chi è il miglior veicolo che deve fare da relay, poichè è la struttura virtuale che lo determina. Alcuni esempi sono [9], dove si costruisce e mantiene una backbone virtuale di veicoli, oppure [3], [7], [8] e [12]. Ad esempio in [7] gli autori propongo la creazione di una backbone, stabilendo un membro all'interno di segmenti di uguale lunghezza tra di loro. Un'altra discussione sui benefici della backbone nella trasmissione di messaggi in ambito veicolare è spiegata in [12]. [3] e [8] discutono invece, tra le altre cose, di diverse metriche per la scelta dei membri della backbone, argomento cruciale per ottenere performance elevate in ambito veicolare.

Ad esempio VWCA[8] si prefigge come scopo quello di eleggere la testa del cluster in modo rapido, cercando quindi di ridurre l'overhead. Si basa inoltre sul mantenimento di due liste, una *white list* e una *black list*, in cui vengono inclusi i veicoli a seconda di una soglia di affidabilità. Si compone in tutto di 5 step:

- Determinare la lista dei vicini: i veicoli mandano periodicamente un messaggio di *hello*, comunicando le proprie caratteristiche, di modo che tutti possano creare la propria lista di vicini.
- Determinare l'affidabilità di ogni veicolo: all'inizio ogni veicolo è ritenuto affidabile, stato che può cambiare col tempo.
- Viene determinata la direzione dei veicoli.
- Calcolo dell'entropia usando l'algoritmo WCA[30]: questo passo cerca di capire l'eterogeneità nel tragitto dei veicoli, cercando di identificare quelli più affini.
- Verifica della somma dei valori: dopo aver ottenuto tutte queste informazioni, i veicoli possono calcolare una somma che determinerà l'elezione della testa del cluster.

Il protocollo è inoltre adattivo, per adattarsi meglio alle condizioni del traffico e dei vari scenari che si possono presentare.

Il protocollo presentato in [3] per eleggere la testa del cluster si basa sulla corsia di appartenenza. Fa l'esempio di una strada con quattro corsie, di cui 3 in una stessa direzione, e forza quindi l'elezione del cluster in una di queste tre corsie, presupponendo chiaramente che ogni veicolo conosca la propria corsia di appartenenza. Ogni veicolo computa il proprio CHL_i, definito come:

$$CHL_i = NCL(t)_i + ADL_i + ADV_i \tag{2.1}$$

dove $NCL(t)_i$ è una stima della connettività del nodo, ovvero quanti vicini ha e con quanti riesce a comunicare, mentre invece ADL_i e ADV_i sono le medie rispettivamente della distanza media da ogni nodo e della velocità del veicolo.

Una volta computato, ogni veicolo manda in broadcast il proprio CHL_i, e chi ha ottenuto il valore maggiore vince e diventa il *cluster head*. Con questo contributo gli autori sono voluti andare oltre il concetto di distanza fra nodi, per cercare di formare dei cluster basandosi sulle caratteristiche dei veicoli, privilegiando quelli che mantengono un andamento più costante nel tempo. È un approccio interessante, ma ritengo che con una elevata mobilità il broadcast continuo del CHL_i di ogni veicolo possa far decadere le performance del protocollo.

Capitolo 3

DBA-MAC

In questo capitolo viene presentato DBA-MAC (*Dynamic Backbone Assisted - MAC*), originariamente proposto in [4] e successivamente esteso in [9], un protocollo basato su backbone virtuale per la rapida disseminazione di messaggi su reti veicolari.

3.0.1 Definizioni

In questa sezione vengono presentate e spiegate tutte le definizioni che verrano poi utilizzate nella discussione e illustrazione del protocollo.

- **BM**: è il *Backbone Member*, ovvero un membro della backbone, eletto secondo la procedura di clustering.

- **NV**: è il *Normal Vehicle*, ovvero un veicolo che non fa parte della backbone attualmente, ma potrebbe diventare in futuro un *BM*.

- **RZ**: è la *Risk Zone*, ovvero quella zona che ha come centro la partenza del messaggio di allarme, e che si estende per un certo raggio. È cioè quell'area in cui i veicoli al suo interno devono essere informati del messaggio di allarme.

- $\mathbf{T_{thr}}$: è il *Time Threshold*, ovvero il limite temporale dopo il quale un messaggio di allarme si considera non più utile. Generalmente questo tempo in applicazioni reali è dell'ordine di qualche millisecondo[1,2]

3.0.2 Assunzioni

Per poter funzionare, il protocollo ha bisogno di conoscere l'esatta posizione del veicolo nello spazio, e questo è possibile grazie a un dispositivo GPS, ormai molto comuni al giorno d'oggi.

Inoltre, si assume anche che ogni veicolo abbia un'interfaccia wireless 802.11, che gli permetta di comunicare con altri veicoli.

3.1 Protocollo

DBA-MAC, come anticipato, è un protocollo per reti veicolari basato su backbone. Il suo funzionamento è riassunto dalla figura 3.1, in cui si nota come ci siano i veicoli che fanno parte della backbone (BM), quelli contornati di rosso, e dei veicoli normali (NV).

Il protocollo è composto da due componenti principali, uno che si preoccupa della creazione della backbone, implementato a livello MAC, e uno che gestisce l'accesso al canale.

La backbone in uno scenario può anche non essere unica, ma composta di più catene che non si sovrappongono tra di loro.

3.1.1 Creazione della backbone

La procedura di creazione della backbone viene iniziata da un NV ψ, non membro di una backbone, che non abbia ricevuto un messaggio di *BEACON* per un intervallo $Time_B$. In questo caso, il veicolo ψ manda in broadcast un messaggio

[1] EuroFOT EU project. http://www.eurofot-ip.eu. FP7

[2] PRE-DRIVEC2X EU project. http://www.pre-drive-c2x.eu. FP7.

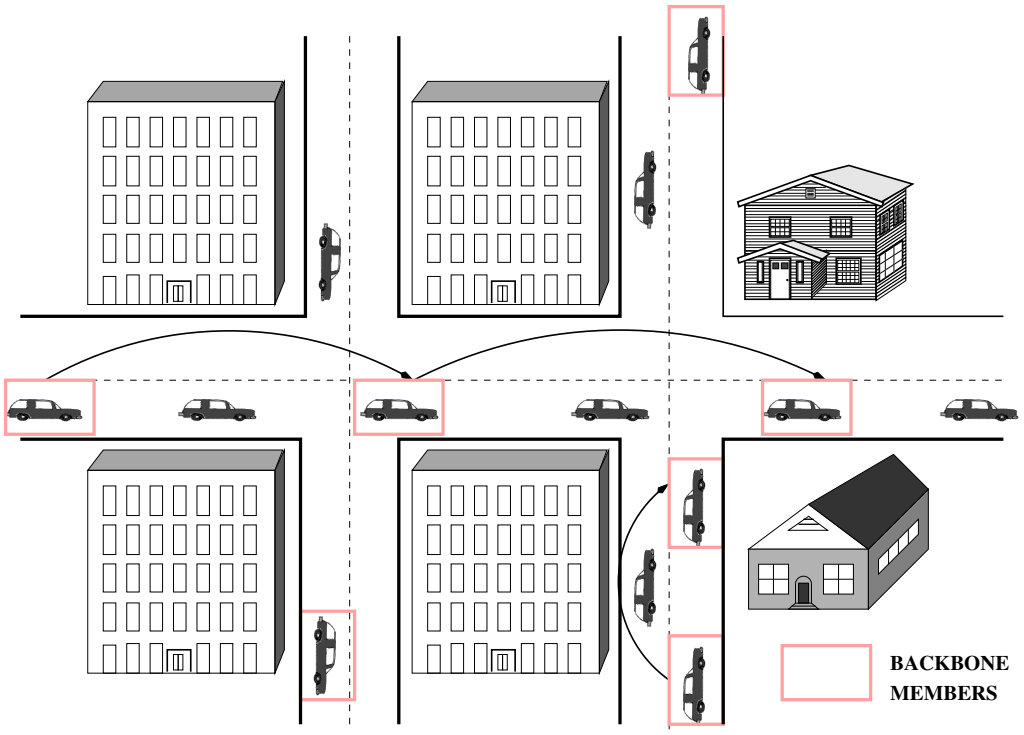


Figura 3.1: Lo scenario urbano del protocollo DBA-MAC

di BEACON per la creazione della backbone, contenente le seguenti informazioni:

$$< x, y, v, \overrightarrow{dir}, seq >$$

dove *(x,y)* sono le coordinate geospaziali di ψ, *v* è l'attuale velocità di ψ, $\overrightarrow{dir}$ è la sua direzione e *seq* è il suo attuale numero di sequenza (attualmente 1, visto che sta creando la backbone).

Per scegliere il prossimo veicolo da aggiungere alla backbone, viene usato una fase di contesa distribuita. Quando un NV η riceve il messaggio, verifica che possa essere eleggibile come candidato ad essere il prossimo BM controllando le seguenti condizioni:

$$\theta < \theta_{MAX} and |v_\psi - v_\eta| < v_{MAX} \tag{3.1}$$

dove θ è la differenza di angolazione tra le direzioni di ψ e η, v_ψ e v_η sono rispettivamente le velocità di ψ e η, e v_{MAX} e θ_{MAX} sono le soglie massime per la differenza di velocità e angolatura tra i veicoli.

Tramite l'equazione 3.1 si cerca di migliorare la stabilità della backbone, impedendo a veicoli con caratteristiche troppo differenti da ψ di potersi candidare per diventare *BM*.

Se una delle due condizioni dell'equazione 3.1 non è verificata, allora η scarta il messaggio di BEACON ed esce dalla procedura.

Prima metrica (minimizzazione degli hop)

Con questa metrica si vuole puntare a minimizzare il numero degli hop richiesti per la comunicazione. Si preferisce dunque scegliere come membri della backbone veicoli che siano posti alla massima distanza possibile di trasmissione.

Se l'equazione 3.1 è verificata, η calcola un *Fit Factor (FF)* [4], che definisce una metrica per stabilire la bontà dell'inserimento di η nella backbone. Definiamo quindi $FF(\eta, \psi)$ come segue:

$$FF(\eta, \psi) = \frac{d}{R_\psi} \tag{3.2}$$

dove R_ψ è il raggio di trasmissione del veicolo ψ. Basandosi sull'equazione 3.2, η modifica la sua CW calcolando $CW(\eta)$ come segue:

$$CW(\eta) = (1 - FF(\eta, \psi)) * (CW_{MAX} - CW_{MIN}) + CW_{MIN} \tag{3.3}$$

dove CW_{MAX} e CW_{MIN} sono rispettivamente delle soglie massime e minime per la CW definite in [1].

Chiaramente, avrà maggiori possibilità di essere inserito come membro della backbone un veicolo η che ha una distanza da ψ maggiore degli altri.

Seconda metrica (miglioramento della qualità dei link)

Con questa metrica si cerca di massimizzare la robustezza della comunicazione tra i veicoli membri della backbone. Per questo motivo, sono da preferire i veicoli che abbiano il più alto link budget tra quelli presenti, proprio per cercare di ottenere i collegamenti migliori possibili.

Una delle più grosse limitazioni della metrica precedente era quella di assumere lo stesso raggio di trasmissione per ogni veicolo, definito come R_ψ. Questo

può valere in un contesto autostradale, scenario nel quale era stato studiato il protocollo proposto in [4], ma non in un contesto urbano dove i raggi di trasmissione possono variare molto rapidamente, a causa di effetti di shadowing e attenuazione di edifici.

Inoltre, la metrica valutata nell'equazione 3.2 favorisce la creazione di backbone con veicoli alla massima distanza di trasmissione, senza però considerare velocità, sensi di marcia e altro. Secondo l'equazione 3.2, un veicolo che procede a una velocità molto inferiore rispetto al mittente alla massima distanza di trasmissione ha più probabilità di candidarsi a diventare un BM rispetto a un veicolo più vicino ma con una velocità più affine al mittente. Questo porta a backbone molto fragili, che possono crearsi e rompersi nel giro di qualche secondo, ma sicuramente aiuta ad abbassare il numero di hop nella comunicazione.

Con questa nuova metrica, invece, si tengono conto di tutti questi fattori, e i risultati di simulazione fanno vedere come la nuova metrica sia in effetti migliore per la stabilità della backbone.

Per prima cosa si definisce l'equazione per il link budget come:

$$LB(\psi, \eta) = P_{rx}^{\eta} - RS_{thr}^{\eta} \tag{3.4}$$

In seguito, si può definire $FF(\psi, \eta)$ come:

$$FF(\psi, \eta) = min\{\frac{LB(\psi, \eta)}{Power_{thr}^{\eta}}, 1\} \tag{3.5}$$

dove $LB(\psi, \eta)$ è il link budget calcolato tramite l'equazione 3.4, e $Power_{thr}^{\eta}$ è la differenza in dBm tra i valori massimi e minimi di trasmissione supportati dall'interfaccia di rete. Per esempio, per una scheda DCMA-86P2, 802.11p, $Rate_{Max}$ è 54Mbps se $P_{rx} >$ di -93 dBm, mentre invece $Rate_{Min}$ è 6Mbps se $P_{rx} >$ di -75 dBm. P_{rx} è invece $>$ di -93 dBm, di conseguenza $Power_{thr}$ è $Power_{thr} = |-93 + 75| = 18dBm$.

Terza metrica (massimizzazione di distanza e qualità del link)

Con questa metrica si cerca di ridurre il tempo di consegna dei messaggi, considerando allo stesso tempo sia la distanza percorsa che la velocità possibile ad ogni hop.

Sia quindi $E[delay]$ il ritardo medio richiesto per trasmettere un messaggio su una RZ di lunghezza h. Se si assume che il messaggio è spedito dai membri della backbone che hanno una distanza media $E[d]$, e che il ritardo introdotto da ogni ritrasmissione è $E[delay_{hop}]$, allora $E[delay]$ sarà:

$$E[delay] = \frac{h}{E[d]} * E[delay_{hop}] \tag{3.6}$$

Quindi, è immediato notare come per minimizzare $E[delay]$ sia necessario scegliere un veicolo della backbone che garantisca il minimo rapporto $\frac{E[delay_{hop}}{E[d]}$, essendo h fissata.

Si va quindi a definire $FF(\psi, \eta)$ come:

$$FF(\psi, \eta) = \frac{d}{R} * \frac{Rate(\psi, \eta)}{Rate_{MAX}} \tag{3.7}$$

dove $\frac{d}{R}$ deriva dalla prima metrica e rispecchia $E[d]$, mentre invece $\frac{Rate(\psi,\eta)}{Rate_{MAX}}$ da una stima di $E[delay_{hop}]$.

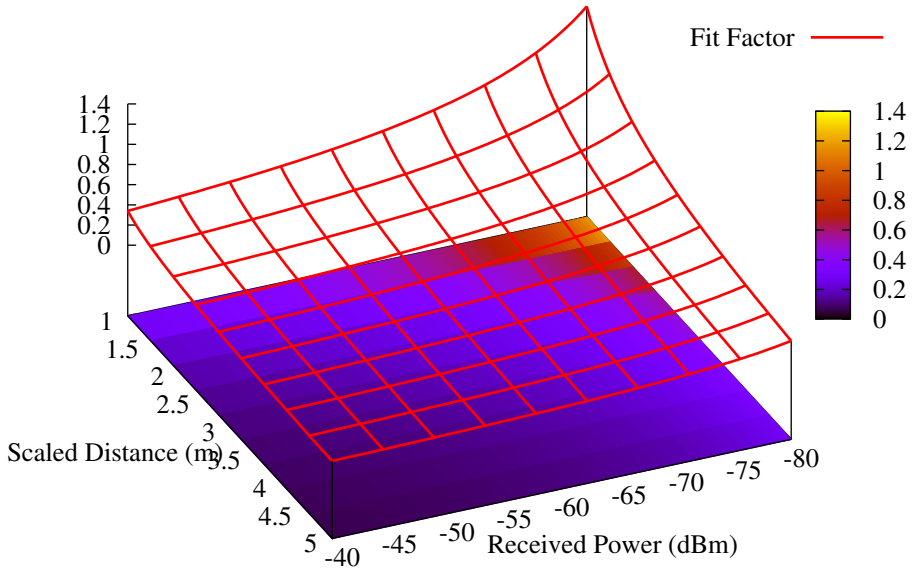


Figura 3.2: I valori di Fit Factoring calcolati secondo l'equazione 3.7

La figura 3.2 fa vedere la variazione di FF(ψ,η) a seconda della variazione di distanza e della potenza ricevuta, utilizzando per il calcolo l'equazione 3.7.

Si nota quindi come la prima metrica sia migliore per applicazioni di sicurezza, che richiedono una veloce disseminazione di brevi messaggi.

La seconda metrica invece garantisce la più alta probabilità di ricezione dei messaggi, dato che i membri della backbone saranno veicoli vicini nello spazio, e quindi ci sarà un elevato numero di ritrasmissioni. In questo modo, un veicolo che magari non ha sentito un messaggio da un membro della backbone, potrebbe risentirlo da un altro veicoli vicino.

Infine, la terza metrica cerca di bilanciare i due aspetti precedenti, e si vedrà che riduce i tempi di consegna per messaggi di media e grande dimensione, pur garantendo comunque una elevata percentuale di messaggi consegnati.

3.1.2 L'accesso al canale

Il protocollo DBA-MAC ha due politiche di ritrasmissione dei messaggi sul canale, denominate FMF (Fast Multihop Forwarding) e CBF (Contention Based Forwarding). L'FMF viene utilizzato quando è effettivamente presente una backbone, al fine di usarla e quindi ottenere le migliori performance, mentre invece il meccanismo CBF entra in gioco quando non è presente la backbone, di modo da garantire comunque la consegna dei messaggi.

FMF - Fast Multihop Forwarding

Il meccanismo FMF garantisce una veloce disseminazione dei messaggi quando è presente una backbone virtuale, in assenza di contesa poichè in questo caso solo i veicoli membri della backbone possono trasmettere. Supponiamo di avere due veicoli, BM_i e BM_{i+1}, rispettivamente con numero di sequenza i e i+1. Così, quando BM_i riceve un messaggio da ritrasmettere, lo manda in broadcast. Quando BM_{i+1} lo riceve, manda dopo un SIFS un ACK a BM_i, e senza rilasciare il canale rimanda il messaggio ad un eventuale BM_{i+2}. Tutti i NV che hanno ricevuto il messaggio ne processano il contenuto, ad esempio per avere informazioni di si-

curezza, ma non rimandano il messaggio e quindi non entrano nella ritrasmissione dello stesso.

Se BM_i non ha un nodo nel prossimo hop, ovvero BM_{i+1}, oppure non riceve un ACK dopo un SIFS, manda comunque il messaggio, che verrà poi replicato da un NV tramite il meccanismo CBF. È immediato quindi notare come il meccanismo FMF garantisca eccellenti tempi di consegna, poichè di fatto non introduce nessun delay aggiuntivo dovuto alla contesa del canale ad ogni hop, ma manda il messaggio nel più breve tempo possibile. Il meccanismo dell'ACK viene utilizzato per migliorare l'affidabilità nella consegna dei messaggi a differenza di altri protocolli reattivi come [7], [25] e [13].

CBF - Contention Based Forwarding

Il meccanismo CBF viene utilizzato quando un NV si accorge che non c'è un BM che possa ritrasmettere il messaggio, o il messaggio è stato rispedito una seconda volta.

A questo punto, tutti i NV entrano in una fase di contesa distribuita per decidere chi dovrà ritrasmettere il messaggio. Quindi, ogni NV modifica la sua contention window come segue:

$$CW(\eta) = \begin{cases} (1 - \frac{d}{R_\psi}) * \Delta CW + CW_{MIN} & Se\ \eta\ è\ un\ NV \\ 4 & Se\ \eta\ è\ un\ BM \end{cases} \tag{3.8}$$

dove d è la stima della distanza tra ψ e η, R_ψ è il raggio di trasmissione massimo di ψ e $\Delta CW = CW_{MAX} - CW_{MIN}$.

Dopo di questo, si comincia la procedura di backoff come definito nel protocollo 802.11p[1], ma cancella la sua ritrasmissione se riceve un'altra ritrasmissione del messaggio. Questo vorrebbe dire che c'è un veicolo λ con caratteristiche migliori di η, che quindi ha la priorità nel ritrasmettere il messaggio. L'equazione 3.8 fa vedere come i veicoli più lontani da ψ abbiano una maggior probabilità di ritrasmettere il messaggio, e che comunque i BM abbiano una CW molto piccola, di modo che il meccanismo FMF possa ripartire dopo ogni hop. Peraltro, questo tipo di approccio è anche utilizzato in [22].

3.1.3 L'algoritmo completo

Viene ora presentato l'algoritmo completo di creazione della backbone, unitamente a quello di spedizione dei messaggi di BEACON da un BM.

Questo algoritmo è presentato nel listato 4

Algorithm 4: La creazione della backbone di DBA-MAC per un veicolo η

if tempo dall'ultimo BEACON $>$ Time_B **then**
  stato$_\eta$ = BM
  Trasmetti in broadcast un `BEACON`
end if

Quando ricevo un messaggio di `BEACON` da ψ
if $\theta < \theta_{MAX}$ and $|v_\psi - v_\eta| < v_{MAX}$ **then**
  Calcola FF(η, ψ)
  Modifica CW_η
  vf = rand(0, CW)
  while faccio backoff **do**
    if se sento un messaggio di candidatura **then**
      cancella la trasmissione della candidatura
      esci dalla procedura
    end if
  end while
  Trasmetti u messaggio di `BEACON`
end if

if ricevo un `ACK_WINNER` da ψ **then**
  stato$_\eta$ = BM
  manda in broadcast un messaggio di `BEACON`
end if

3.1.4 Modello Analitico

In questa sezione viene presentato il modello analitico in una versione semplificata dello scenario, cioè quello autostradale.

Assumiamo che ci siano N veicoli nell'autostrada di lunghezza L, e che siano posizionati a una distanza tra di loro uniforme $\frac{L}{N}$, e che si muovano a una velocità costante v, e sia R il raggio di trasmissione di ogni veicolo.

Come prima cosa, ricaviamo un'espressione che mostri il ritardo nella trasmissione dei messaggi in una RZ di lunghezza h, sia per il meccanismo FMF[3] che per il CBF.

Possiamo vedere che sia nel meccanismo FMF che in quello CBF, i veicoli hanno una fase di contesa per l'accesso al canale. In FMF, per decidere il prossimo veicolo della backbone, e in CBF per decidere chi dovrà ritrasmettere il messaggio.

Assumiamo che un veicolo η trasmetta un messaggio di BEACON. Dati N, L e R, il numero di veicoli che si trovano a 1 hop di distanza da η è dato da:

$$n = \lfloor \frac{R * N}{L} \rfloor \tag{3.9}$$

Chiamiamo V_1,V_2,...,V_n, rispettivamente il primo, il secondo e l'ultimo vicino di η. Definiamo anche:

$$D_i = i * \frac{L}{N} \tag{3.10}$$

come la distanza tra il veicolo η e V_i.

Chiamiamo quindi CW(i) la contention window di V_i, calcolata tramite l'equazione 3.3, e con B_i il valore di backoff scelto casualmente tra [0,CW(i)[. La probabilità che V_i vinca la contesa e che quidi diventi il prossimo hop (P(NH = V_i)) è data dalla probabilità che V_i scelga un valore di backoff minore di V_j, $1 \leq j \leq n$:

$$P(NH = V_i) = \sum_{k=0}^{CW(i)} \prod_{j=0, j \neq i}^{n} P(B_i = k) * P(B_j > k) \tag{3.11}$$

[3] Secondo l'equazione 3.2

Assumendo che ogni veicolo scelga casualmente uno slot per la sua CW, allora $P(B_i = k)$ e $P(B_j > k)$ possono essere calcolati come segue:

$$P(B_i = k) = \frac{k}{CW(i)} \tag{3.12}$$

$$P(B_j > k) = max\left\{\frac{CW(j) - k}{CW(j)}, 0\right\} \tag{3.13}$$

Possiamo ora derivare la distanza media tra gli hop della catena con l'equazione 3.14:

$$E[d] = \sum_{i=1}^{n} P(NH = V_i) * D_i \tag{3.14}$$

Dall'equazione 3.6, il tempo medio richiesto per coprire una RZ di lunghezza h è definito come il prodotto del numero medio di ritrasmissioni per il ritardo introdotto a ogni salto. Chiaramente $E[delay_{hop}]$ dipende dallo schema di accesso al canale che viene considerato. Usando l'FMF, questo tempo è uguale al tempo richiesto per trasmettere il messaggio più il tempo per ricevere un ACK, ovvero:

$$E[delay_{hop}]_{FMF} = T_{SIFS} + T_{ALERT} + T_{SIFS} + T_{ACK} \tag{3.15}$$

Se invece consideriamo lo schema CBF, occorre anche tenere conto del ritardo di backoff introdotto a ogni salto:

$$E[delay_{hop}]_{CBF} = T_{DIFS} + T_{ALERT} + E[CW] * T_{\sigma} \tag{3.16}$$

Dove T_{σ} è la durata dello slot definita in 802.11p [1] e $E[CW]$ è la dimensione media della contention window del vincitore della fase di contesa, definita come:

$$E[CW] = \sum_{i+1}^{n} P(NH = V_i) * \frac{CW(i)}{2} \tag{3.17}$$

Infine, introduciamo p, ovvero la probabilità che venga usato lo schema FMF invece del CBF, e di conseguenza anche la probabilità complementare ovvero $1 - p$.

Quindi, possiamo definire il ritardo introdotto indipendentemente dallo schema utilizzato, utilizzando l'equazione 3.18:

$$E[delay_{hop}]_{DBA} = p * E[delay_{hop}]_{FMF} + (1 - p) * E[delay_{hop}]_{CBF} \tag{3.18}$$

Con $1 - p$ si prendono in considerazione tutti quei casi in cui la backbone fallisce, o per colpa del canale o perchè non cè un BM per ritrasmettere il messaggio. Partendo dall'equazione 3.18, possiamo anche trovare il caso ideale, ovvero quello in cui:

1. Tutti gli hop comunicano con FMF
2. I veicoli sono posizionati alla massima distanza di trasmissione
3. I veicoli sono fissi

Chiamo questo caso SBA-MAC, ovvero *Static Backbone Assisted - MAC*, per differenziarlo da DBA-MAC in cui le backbone vengono create a seconda della condizioni del traffico e del canale.

Quindi, il tempo di consegna dei messaggi di SBA-MAC può essere calcolato partendo dalle precedenti equazioni:

$$E[delay]_{SDA} = \frac{h}{R} * E[delay_{hop}]_{FMF} \tag{3.19}$$

Consideriamo uno scenario in cui:

- H = 10000m
- T_{ALERT} = 0.004 s
- T_{σ} = 0.00002 s
- CW_{MIN} = 32
- CW_{MAX} = 1024

e vediamo i risultati ottenuti dal confronto tra il modello analitico e alcune simulazioni. In queste analisi prendiamo anche in considerazione il protocollo denominato *SBA-MAC*, ovvero *Static Backbone Assisted MAC*, che è il caso in cui i nodi non si muovono e sono posti alla massima distanza di trasmissione l'uno dall'altro. Ovviamente in questa configurazione l'unico meccanismo di trasmissione utilizzato è l'FMF, poichè la backbone esiste grazie al posizionamento iniziale dei

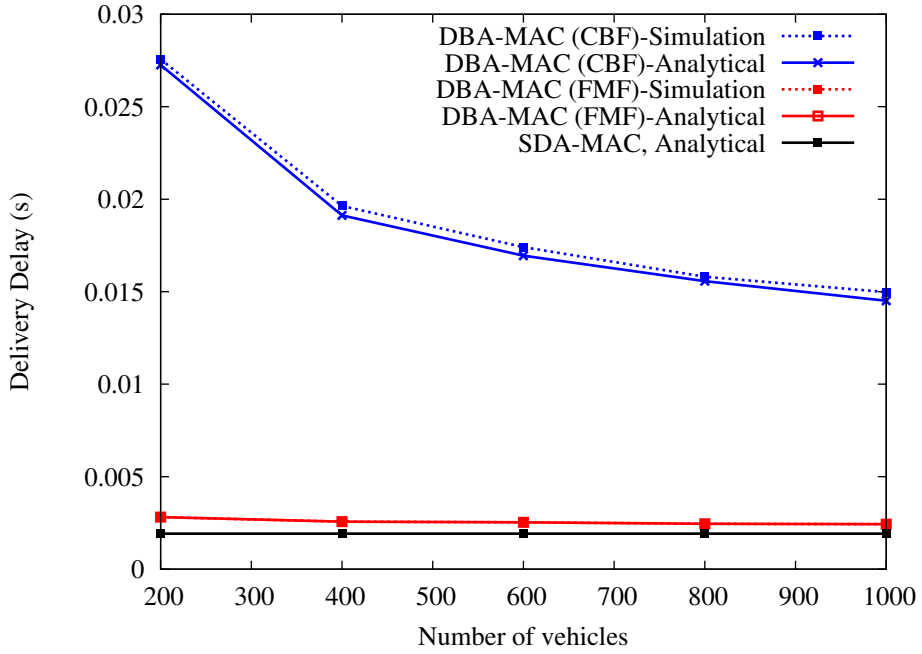


Figura 3.3: Il ritardo nella consegna dei messaggi

nodi. *SBA-MAC* ci può dare una stima inferiore del protocollo *DBA-MAC*, in altre parole ci fa vedere le performance migliori ottenibili.

Si nota dalla figura 3.3 come il modello analitico rispecchi fedelmente quanto ottenuto con le simulazioni. Vediamo anche grazie a questo grafico come DBA-MAC, utilizzando il meccanismo FMF, sia molto vicino a SBA-MAC (*Static Backbone Assisted MAC*. Anche DBA-MAC con il meccanismo CBF fa segnare buoni risultati, ma solo se i veicoli aumentano. Ad ogni modo, il grafico 3.3 valida il modello analitico presentato in questa sezione, dato che i risultati di simulazione sono molto simili a quelli ottenuti utilizzando le formule presentate.

Nella figura 3.4 invece vediamo sempre il ritardo nella consegna dei messaggi, ma al variare di p. Si ricorda che p è il parametro che permette di introdurre la probabilità di utilizzare il meccanismo FMF o quello CBF. Se $p = 1$ allora utilizziamo sempre il meccanismo FMF, mentre invece la probabilità complementare, ovvero $p = 0$, indica sempre l'utilizzo del meccanismo CBF. In altre parole, p ci rappresenta la mobilità e le caratteristiche dello scenario: con $p = 1$ abbiamo

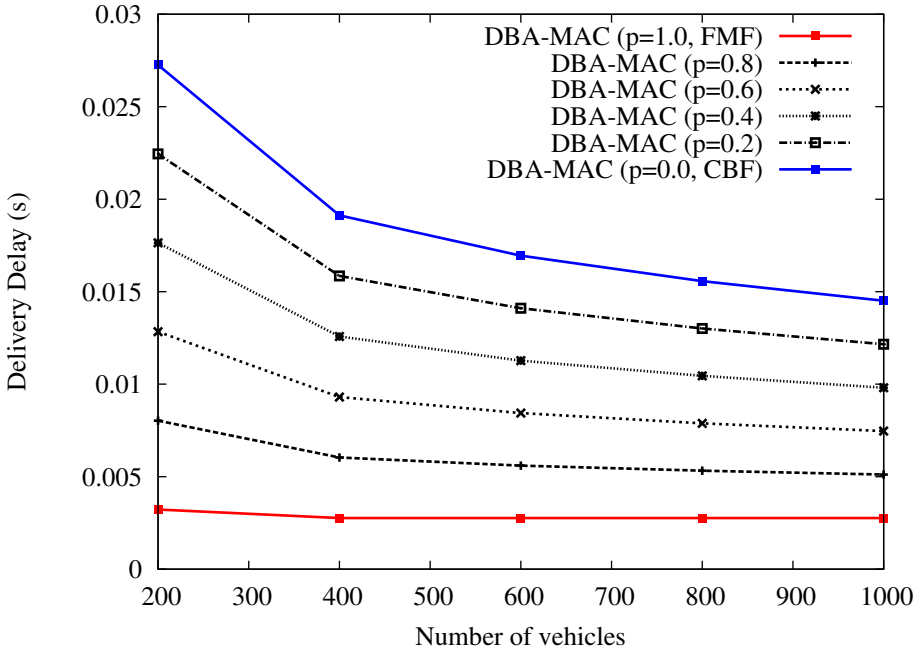


Figura 3.4: Il ritardo nella consegna dei messaggi al variare di p

infatti il caso ideale, poichè riusciamo sempre a trasmettere il messaggio grazie ai nodi della backbone, senza dover ricorrere a un *NV*, mentre invece con $p = 0$ i messaggi vengono sempre trasmessi da *NV*, che è ovviamente il caso peggiore. Il grafico 3.4 mostra come effettivamente FMF sia molto più rapido di CBF, e in mezzo tra i due estremi ci stiano, chiaramente, tutti gli altri casi, in cui FMF e CBF si alternano tra di loro.

Un'altra cosa che si nota è come i dati non convergano a uno stesso punto, ma si stabilizzino piuttosto a un certo livello al variare di p. Questo avviene poichè comunque, anche con molti veicoli, il meccanismo CBF garantirà sempre e comunque performance peggiori rispetto all'FMF. $E[delay_{hop}]_{FMF}$ è minore di $E[delay_{hop}]_{CBF}$, e la grossa differenza la fa proprio il tempo di attesa dovuto alla contesa del canale. Meno abbiamo questa contesa, più i messaggi verranno rispediti in modo rapido e senza aspettare.

Nella figura 3.5 si vede come le differenti metriche abbiano impatto sulle performance del protocollo. È evidente come la metrica numero 1, volta alla mini-

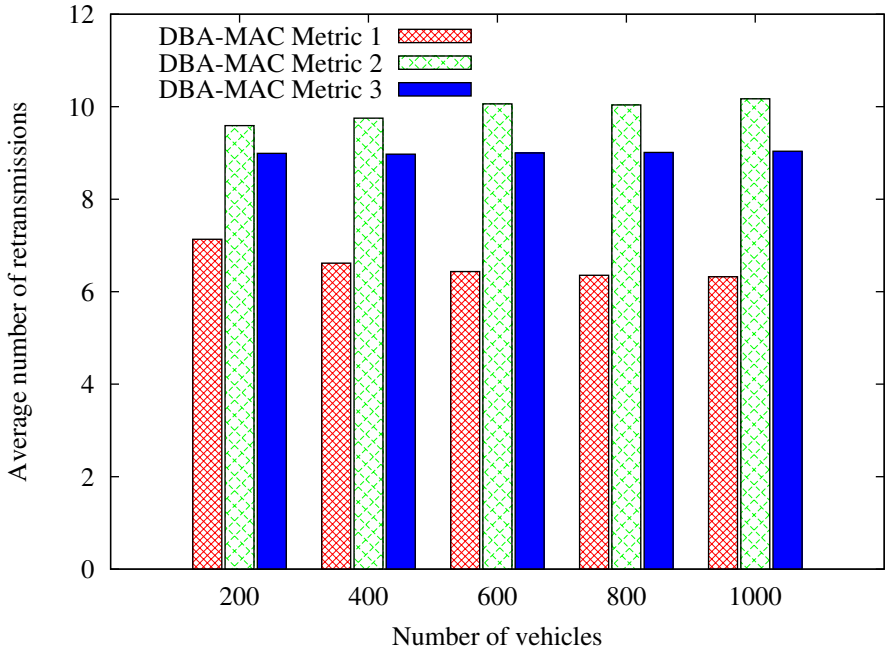


Figura 3.5: Il numero medio di ritrasmissioni al variare delle metrica

mizzazione degli hop, sia effettivamente quella che dimostri il minor numero di ritrasmissioni. La seconda metrica invece è sempre quella con il numero di ritrasmissioni più elevato, fatto derivante dalla scelta di veicoli vicini tra di loro. La terza metrica infine cerca di bilanciare la distanza e la qualità del link, e in effetti rimane sempre in mezzo ai due estremi.

Nel grafico 3.6 si vede il ritardo nella trasmissione dei messaggi agli estremi della RZ al variare della metrica utilizzata per formare la backbone. Anche in questo caso i risultati confermano quanto ci si aspettava, ovvero che la prima metrica, minimizzando il numero degli hop, garantisce anche il ritardo minore nella consegna dei messaggi. La seconda metrica invece è la peggiore da questo punto di vista: scegliendo infatti veicoli vicini tra di loro, i tempi di consegna crescono di molto. La terza metrica, bilanciando, si pone in mezzo come nel grafico 3.5.

Diventa quindi di cruciale importanza formare backbone solide, che permettano la disseminazione rapida di messaggi senza collisioni. Il grafico 3.4 ne è la

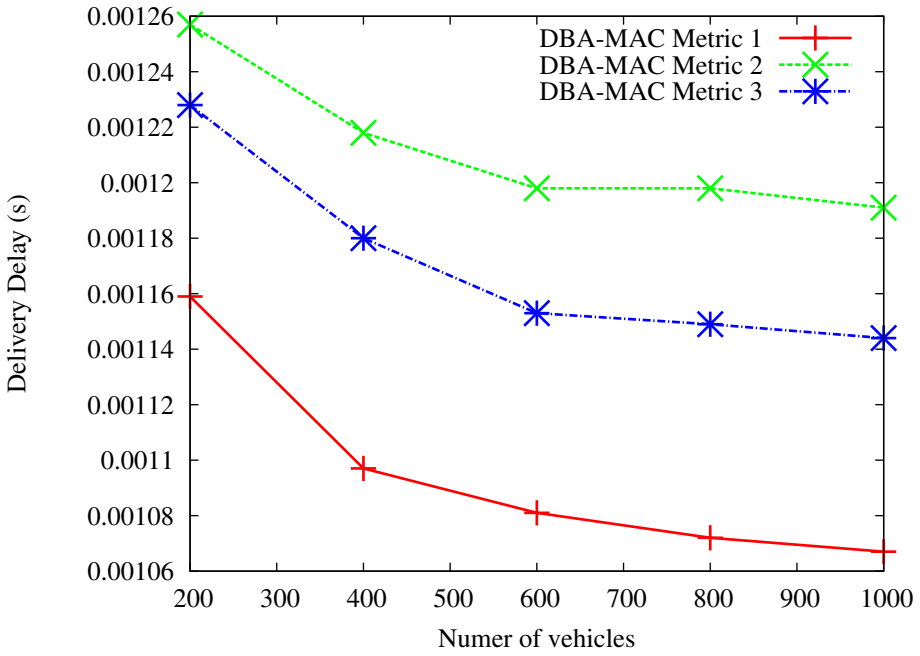


Figura 3.6: Il ritardo nella trasmissione

prova: se trasmettiamo solo in base alla distanza, quindi con il metodo *CBF* con $p = 0$, abbiamo dei tempi di trasmissione molto maggiori rispetto a quando usiamo il metodo *FMF*, che permette di trasmettere con tempi di attesa molto minori rispetto al meccanismo CBF.

Capitolo 4

Performance Evaluation

Simulare un contesto veicolare non è cosa semplice, ci si è quindi avvalsi di strumenti avanzati che permettessero di realizzare simulazioni il più possibile realistiche, introducendo anche il fenomeno di shadowing.

4.1 Tools

4.1.1 Omnet++

Omnet++[1] è un simulatore di reti molto utilizzato, con un IDE grafico che offre anche la generazione di grafici in modo automatico.

La scelta è ricaduta su Omnet++ poichè il recente aggiornamento, dalla versione 3 alla 4, ha fatto fare un grosso passo avanti al programma, che ora offre molte funzionalità evolute utilissime per la simulazione di reti veicolari.

Inoltre, Omnet++ fornisce un modello di programmazione a blocchi simile a quello ISO/OSI, e quindi molto intuitivo per progettare protocolli di rete cross-layer.

Omnet++ è attivamente sviluppato, e vanta una moltitudine di moduli aggiuntivi creati dalla comunità, alcuni dei quali sono anche stati utilizzati in questo lavoro di tesi.

[1] Omnet++ Network Simulation Framework. Website: http://www.omnetpp.org

Omnet++ è disponibile con due licenze separate: per un utilizzo all'interno di università il prodotto è gratuito, se invece l'utilizzatore è un'azienda occorre pagare una somma all'azienda produttrice.

Per progettare un modulo in Omnet++ occorrono (almeno) 2 file: un .ned, che definisce la struttura del modulo, quindi eventuali interfacce di input/output, il numero e il tipo di parametri di configurazione del modulo e così via.

```
simple DbaMac
{
    parameters:
        string address = default("auto");
         // MAC address as hex string , or
            // "auto". "auto" values will be replaced by
            // a generated MAC address in init stage 0.
        int maxQueueSize;
        bool rtsCts = default(false);
        double bitrate @unit("bps");
        int broadcastBackoff;
        int mtu = default(1500);
        @display("i=block/layer");
        // Dba-mac
        bool doublenexthop;
        int BB_REFR;
        int BACKBONE_BEACON_TIMER;
        bool parReset;
        int frequency;
        double probability;
        int fitFactorKind = default(0); // 0 -> Normal fitfactor
         // 1 -> Link budget
    gates:
        input uppergateIn @labels(Ieee802Ctrl);
        output uppergateOut;
```

```
        input lowergateIn @labels(Mac80211Pkt);
        output lowergateOut @labels(Mac80211Pkt);
}
```

Ciò che si vede nel precedente listato è proprio la definizione del modulo DBA-MAC all'interno di Omnet++. Vediamo infatti una serie di parametri (*parameters*) e una serie di interfacce di input/output (*gates*). Questi ultimi rappresentano proprio il modo per passare dati tra un modulo ed un altro. Mandando infatti un dato ad esempio sul gate *lowerGateOut*, un qualsiasi altro modulo che si legherà a questo *gate* riceverà il dato, simulando appunto quello che è lo stack ISO/OSI.

Questo modello di programmazione dei moduli di simulazione permette un elevato riutilizzo del codice e la possibilità di simulare i protocolli cambiano anche diversi livelli dello stack. Ad esempio, se si volesse simulare DBA-MAC con una interfaccia fisica diversa, variando ad esempio la potenza del segnale, e in generale tutti i parametri che stanno al di sotto di DBA-MAC, questo sarebbe possibile senza cambiare una singola riga di codice al protocollo. Semplicemente, nel modulo della macchina, che viene presentato di seguito, occorrerà collegare un'altra interfaccia di rete a DBA-MAC.

```
module CarDbamac {
    parameters:
        @display("bgb=424,541");
    gates:
        input radioIn;

    submodules:
        notificationBoard: NotificationBoard {
            parameters:
                @display("p=140,462;i=block/control");
        }
        interfaceTable: InterfaceTable {
```

```
            parameters:
                @display("p=140,326;i=block/table");
        }
        mobility: TraCIMobility {
            parameters:
                @display("p=60,459;i=block/cogwheel");
        }
        routingTable: RoutingTable {
            parameters:
                IPForward = true;
                routerId = "";
                routingFile = "";
                @display("p=60,326;i=block/table");
        }
        udp: UDP {
            parameters:
                @display("p=384,146;i=block/transport");
        }
        networkLayer: NetworkLayer {
            parameters:
                proxyARP = false;
                @display("p=304,327;i=block/fork;q=queue");
            gates:
                ifIn[1];
                ifOut[1];
        }
        wlan: DbaNic {
            parameters:
                @display("p=304,461;q=queue;i=block/ifcard");
        }
    connections allowunconnected:
```

```
        udp.ipOut --> networkLayer.udpIn;
        udp.ipIn <-- networkLayer.udpOut;

        wlan.uppergateOut --> networkLayer.ifIn[0];
        wlan.uppergateIn <-- networkLayer.ifOut[0];

        radioIn --> wlan.radioIn;

}
```

In questa definizione si vede infatti la costruzione di un modulo più complesso, ovvero quello di una macchina. Anche qui si hanno i parametri (*parameters*), le interfacce (*gates*), una lista di sottomoduli (*submodules*) da cui è composto il veicolo e le connessioni (*connections*) che ci sono tra questi moduli. Cambiando queste connessioni, collegando cioè moduli diversi, si riesce a comporre delle macchine costruite in modo diverso, simulandone quindi l'eterogeneità.

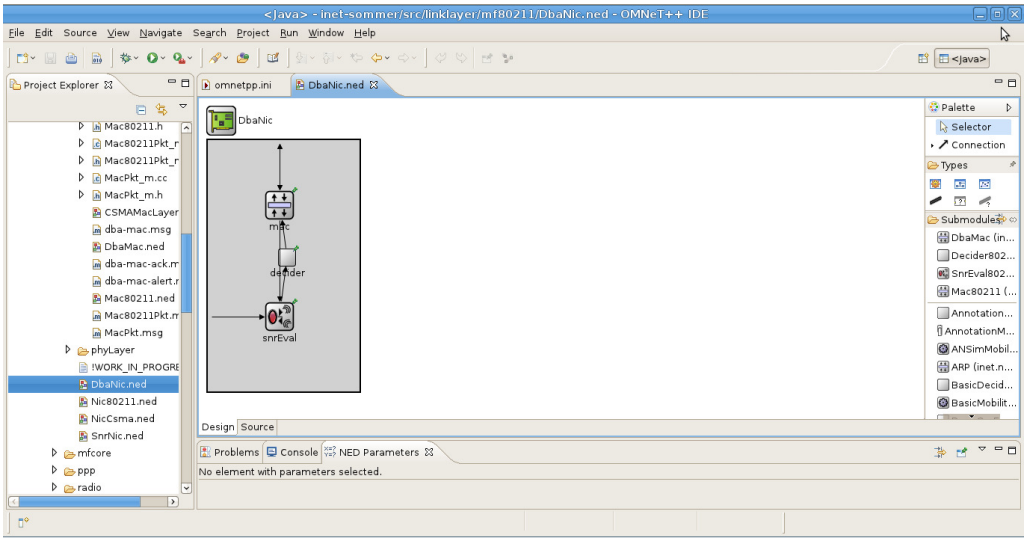


Figura 4.1: La schermata dell'IDE Omnet++ con aperto il file che mostra la composizione del protocollo DBA-MAC

Un altro esempio lo possiamo vedere nella figura 4.1, che mostra graficamente il modulo di una scheda di rete compatibile con il protocollo DBA-MAC. Si no-

ta chiaramente come la filosofia di Omnet++ sia quella di comporre dei singoli moduli, che si vanno poi a unire assieme mediante l'utilizzo dei già citati *gates*.

Nello specifico, si notano il canale wireless che arriva dall'esterno, che manda i propri dati al modulo *snrEval*, modulo dedicato a scartare i messaggi che presentano troppo rumore per essere utilizzabili. Ogni altro canale può essere collegato con vari moduli, e ogni modulo complesso, come quello presentato in figura 4.1, può poi essere trattato ed usato nella composizione di ulteriori moduli, permettendo una scalabilità e una profondità nella programmazione eccellente.

Infine, il modulo vero e proprio viene programmato in C++, e si può avvalere di primitive proprie di Omnet++ per comunicare con gli altri, creare messaggi, ottenere informazioni sulla simulazione e altro.

Si passa quindi a configurare le singole simulazioni, attraverso il file *omnetpp.ini*, in cui si possono specificare i parametri dei moduli già programmati, oltre al numero di ripetizioni, e a diverse opzioni per controllare le esecuzioni.

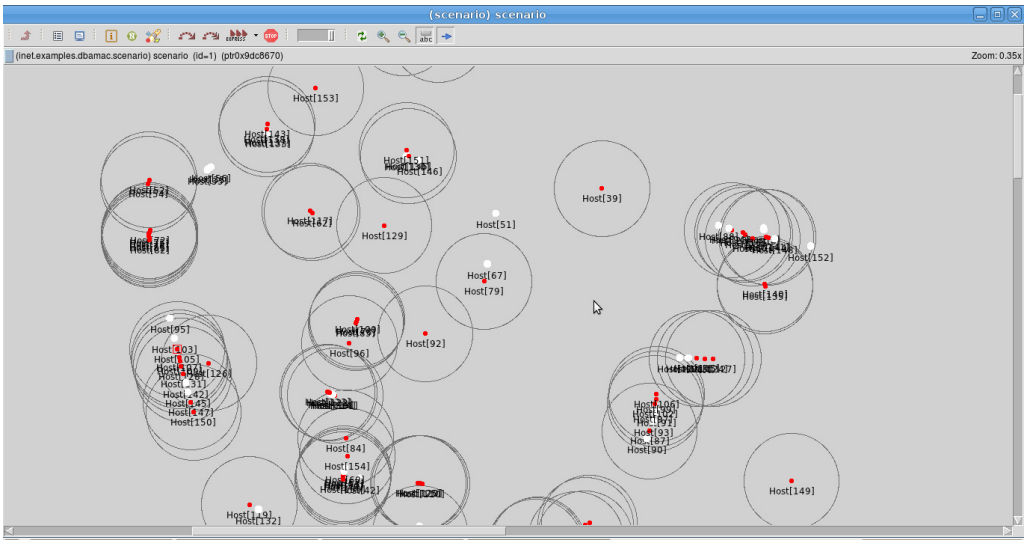


Figura 4.2: La schermata di Tkenv con una run di simulazione con il protocollo DBA-MAC

Infine, tutti i moduli vengono compilati assieme in un unico eseguibile, che può poi essere fatto partire in due ambienti diversi, uno grafico (*Tkenv*), mostrato nella figura 4.2, e uno testuale (*Cmdenv*). Generalmente *Tkenv* si utilizza in fase

di debug del modulo, poichè permette di vedere i singoli messaggi che vengono mandati, come i nodi si muovono nello spazio e in genere tutte le interazioni che avvengono. *Cmdenv* invece si usa quando i moduli sono già stati testati, e quindi devono essere ripetute le simulazioni più volte per ottenere valori più affidabili o comunque per testare diverse configurazioni in modo semi-automatico.

4.1.2 SUMO

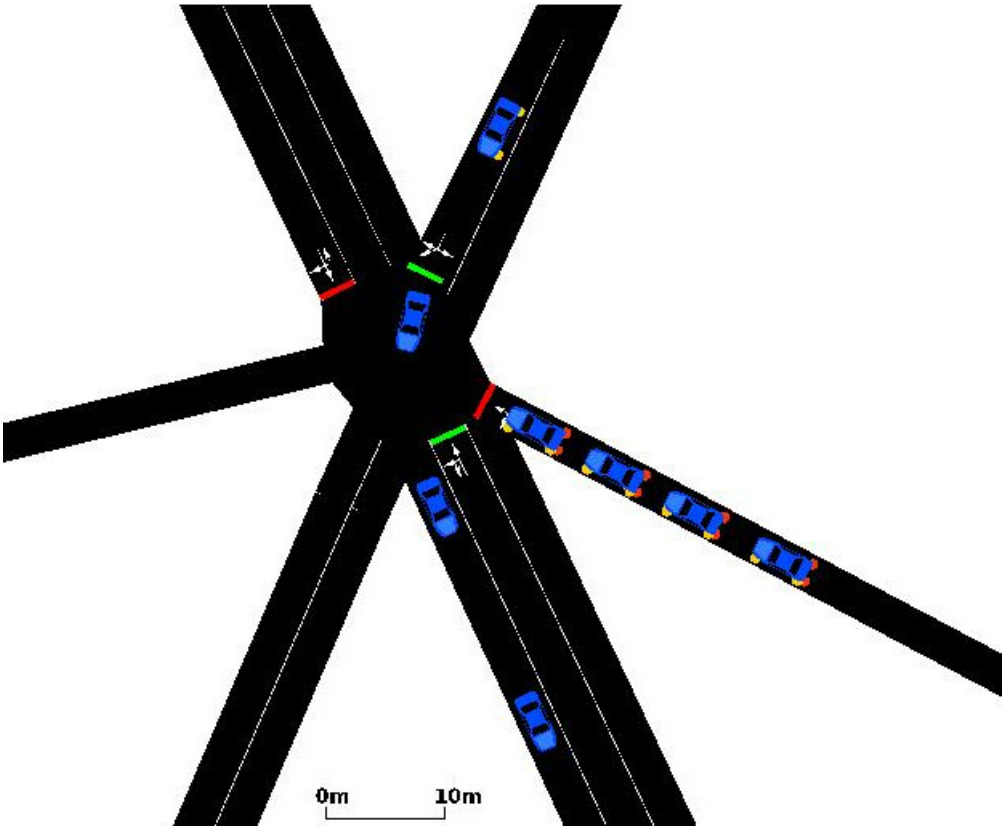


Figura 4.3: sumo-gui, il tool grafico di SUMO che permette di vedere le strade e i veicoli che le percorrono

SUMO[18][2] è un simulatore veicolare sviluppato in Germania, all'interno dell'istituto di ricerca sui trasporti presso il centro aerospaziale tedesco.

[2]Simulation of Urban MObility (SUMO) Project. Website: http://sumo.sourceforge.net

È un simulatore di mobilità microscopico, ovvero può arrivare a modellare e simulare le più piccole caratteristiche dei tragitti, come ad esempio i semafori, le corsie, i sensi unici.

SUMO rimane comunque abbastanza leggero e scalabile, è infatti possibile simulare svariate migliaia di veicoli contemporaneamente su un normale PC desktop.

SUMO si compone di svariati moduli e programmi, ma in particolare ce ne sono due principali che sono poi quelli che si usano normalmente nelle simulazioni, ovvero *sumo* e *sumo-gui*. I nomi già chiariscono il tipo di programma, con il primo che è testuale e viene sovente utilizzato assieme a TRACI, che verrà presentato nella prossima sezione, mentre invece il secondo viene utilizzato per capire i flussi e vedere graficamente il traffico stradale, come mostra la figura 4.3.

Traci e VEINS

Traci come anticipato è un modulo di SUMO che permette ad altri programmi esterni di comunicare col simulatore, al fine di conoscere statistiche, modificare alcuni parametri, variare il percorso di veicoli. VEINS[3] è invece un modulo di Omnet++ che permette di comunicare con SUMO dal simulatore.

Tramite questo modulo è possibile ottenere informazioni sui veicoli, quali il consumo di carburante, la posizione nello scenario, la strada percorsa, la velocità e molti altri. VEINS inoltre introduce anche un modello di shadowing[26] abbastanza leggero, che ha mostrato buoni risultati anche in prove sperimentali, e che quindi si presta bene a simulare il comportamento dei protocolli anche in presenza di ostacoli.

Unendo VEINS con SUMO è quindi possibile utilizzare quest'ultimo all'interno di Omnet++.

[3] Vehicles in Network Simulation (VEINS) Project. Website: http://veins.car2x.org.

4.2 Scenario

Lo scenario per le simulazioni del protocollo è di tipo urbano. Utilizzando SUMO, e importando mappe realistiche di scenari urbani da OpenStreetMap[4] si sono costruiti dei flussi veicolari. Tramite Omnet++ è anche stato possibile modellare anche lo shadowing e l'attenuazione introdotte degli edifici, grazie al modulo VEINS.

Senza perdere di generalità, consideriamo un'applicazione installata sui veicoli che manda messaggi generici di allarme, ad esempio per incidenti o brusche frenate, ogni 100 secondi. Non tutti i veicoli manderanno questi messaggi di allarme, ma solo una parte, ovvero quelli che si accorgono del pericolo imminente. Ogni messaggio è rispedito fino a coprire l'area della RZ, definita in 1km, partendo dal veicolo che ha generato l'allarme.

Vengono quindi valutati tre protocolli:

- *Flooding*: è il caso di base, quello presentato nell'algoritmo 1, in cui viene utilizzato il protocollo 802.11p per ritrasmettere il messaggio. Come detto in precedenza, questo porterà a numerose collisioni, visibili nei grafici.
- *DBA-MAC (CBF)*: è il caso in cui la backbone non viene creata, e ogni veicolo ritrasmette basandosi sull'equazione 3.8.
- *DBA-MAC (CBF + FMF)*: è il protocollo vero e proprio, in cui viene utilizzato il meccanismo FMF quando questo è possibile, e si passa al CBF quando l'FMF presenta dei problemi, come ad esempio una backbone non presente.

Per quanto riguarda invece le metriche valutate per l'analisi, queste sono:

- *Delivery Delay (DD)*: è il tempo medio che impiega il messaggio a percorrere una distanza prefissata dal mittente.
- *Delivery Ratio (DR)*: è la percentuale di veicoli nella RZ che hanno ricevuto il messaggio di allarme, a fronte di tutti quelli che avrebbero dovuto riceverlo.

[4]OpenStreetMap Project. Website: http://www.openstreetmap.org

- *System Load (SL)*: è il carico del sistema, inteso come i Bps impiegati per ritrasmettere i messaggi di allerta.

4.3 Risultati

Sono stati considerati due scenari urbani, uno è quello di Manhattan (un'area di circa 2,5km x 5km) e uno è quello di Bologna (un'area di circa 1,5km x 2,5km).

4.3.1 Delivery delay

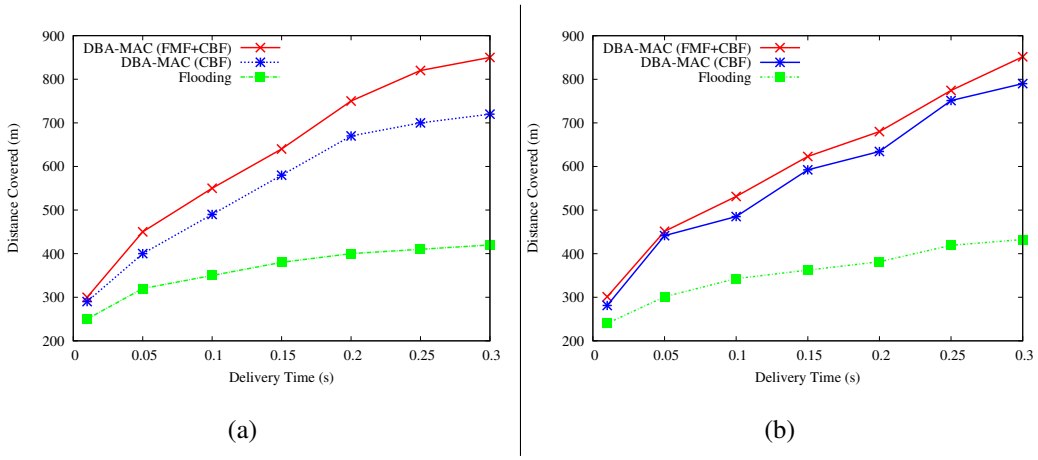


Figura 4.4: Il fenomeno del broadcast storm - graficamente

Il grafico 4.4(a) mostra i tempi di consegna sull'asse delle ascisse, e la distanza coperta sull'asse delle ordinate, nello scenario di Manhattan. Si nota chiaramente che per tempi piccoli le distanze coperte sono abbastanza simili, questo perchè non sono molti gli hop da percorrere e quindi le collisioni non penalizzano molto il protocollo di flooding. Quando però andiamo a vedere tempi maggiori, notiamo come DBA-MAC abbia performances nettamente migliori, arrivando addirittura a doppiare la distanza rispetto al protocollo di flooding a parità di tempo.

Il grafico mostra anche come il meccanismo FMF abbia delle performance migliori, fatto dovuto a una minor attesa per l'accesso al canale, come sintetizzato dall'equazione 3.8. È evidente come più veicoli ci siano nello scenario, più la

distanza in termini di prestazioni tra DBA-MAC e il protocollo base di *flooding* sia evidente. Banalmente, per DBA-MAC più veicoli vuole dire poter compiere delle scelte migliori come BM, mentre invece per il protocollo di flooding vuol dire più collisioni.

Il grafico 4.4(b) mostra invece la stessa metrica, ma su scenario di Bologna, quindi particolarmente differente rispetto a quello di Manhattan. Lo scenario Manhattan infatti presenta strade spesso perpendicolari tra di loro, con incroci molto simili come forma gli uni agli altri. A Bologna, invece, le strade e gli incroci sono di tipologie molto diverse con strade differenti anche in aree limitate, sia come forma che come velocità. Altra cosa da tenere presente sono invece le angolature degli incroci, come già detto spesso perpendicolari nello scenario di Manhattan, e senza uno schema in quello di Bologna.

I valori sono molto simili tra il grafico 4.4(b) e il 4.4(a), e questo conferma la validità di DBA-MAC anche su scenari urbani molto diversi tra di loro. Questo risultato non è scontato, poichè i fenomeni di shadowing potrebbero far peggiorare le performance in uno scenario come Bologna, fatto dovuto appunto alla forma delle strade estremamente eterogenea. A Manhattan, invece, un incrocio non vuole per forza dire interrompere la comunicazione, poichè spesso si riesce a comunicare da un lato all'altro dell'incrocio in linea retta.

4.3.2 Delivery delay su 400m

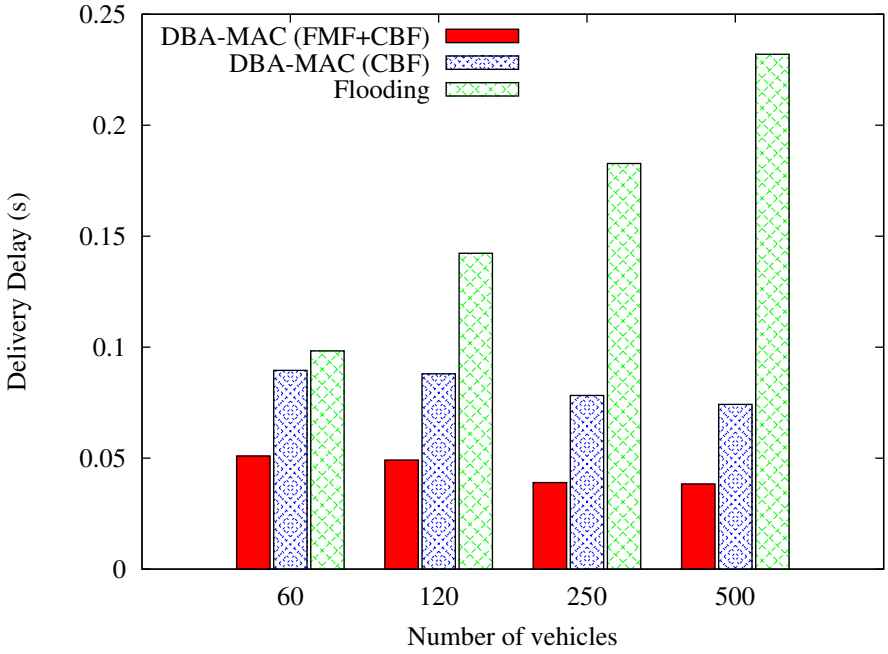


Figura 4.5: I tempi di consegna medi per percorrere 400m, variando il numero dei veicoli nello scenario

Il grafico 4.5 mostra i tempi di consegna dei messaggi per percorrere una distanza fissata di 400m. È significativo vedere come le performance di DBA-MAC migliorino all'aumentare dei veicoli nello scenario, a differenza del protocollo di flooding che invece peggiora sensibilmente.

Sebbene questo possa sembrare strano a prima vista, è invece un naturale comportamento di DBA-MAC. Un numero maggiore di veicoli vuole infatti dire poter scegliere veicoli migliori come membri della backbone, ovvero posizionati alla distanza ottimale e con caratteristiche il più possibile simili a quelle degli altri membri della backbone.

Sono stati presi i 400 metri come distanza di riferimento poichè è una distanza alla quale è plausibile pensare che tutti, in presenza di un messaggio di allarme, possano reagire tempestivamente e in sicurezza. Questo grafico può anche essere

visto come una stima della velocità raggiunta e del tempo impiegato per coprire la distanza minima di informazione.

4.3.3 Delivery ratio

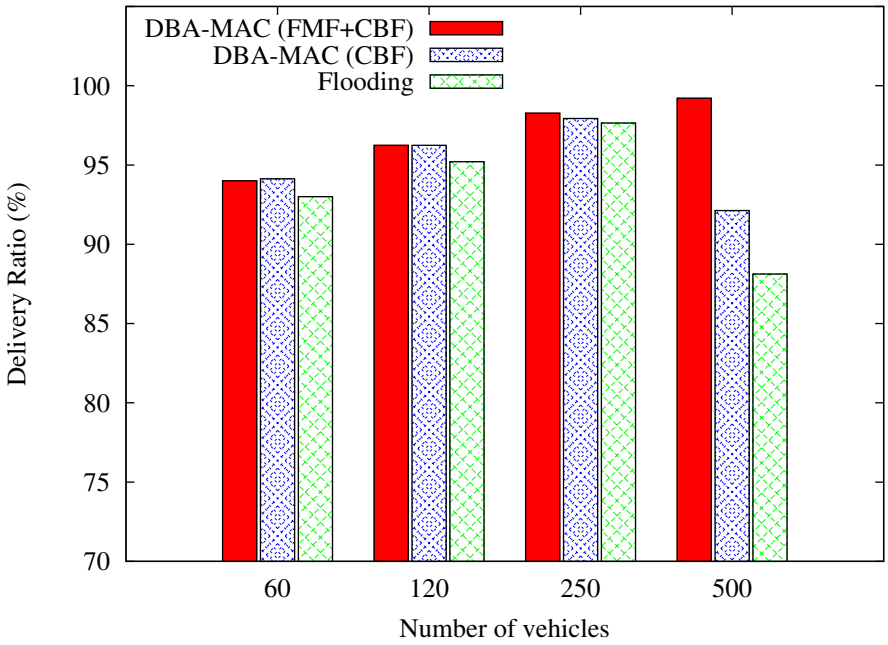


Figura 4.6: Le percentuali di consegna dei messaggi

La figura 4.6 mostra la percentuale di messaggi recapitati ai veicoli rispetto a tutti quelli presenti nella RZ. Risponde all'equazione:

$$DR = \frac{\#\ veicoli \in RZ\ che\ hanno\ ricevuto\ il\ messaggio}{\#\ veicoli \in RZ\ all'istante\ della\ prima\ trasmissione} \tag{4.1}$$

Si nota come fino ad un numero relativamente basso di veicoli, ovvero 250, le percentuali dei protocolli rimangono ottime, seppure con numeri lievemente migliori per DBA-MAC. La grossa differenza si nota invece con 500 veicoli, dove DBA-MAC(FMF+CBF) rasenta il 100%, mentre invece DBA-MAC (CBF) è circa su un 92%, e il risultato peggiore è fatto segnare dal protocollo di Flooding, che scende all'87,5 %.

Anche in questo caso, si vede come DBA-MAC abbia un comportamento sempre migliore all'aumentare dei veicoli. In particolare, per 500 veicoli nello scenario, ovvero circa 40 per km^2, le percentuali di consegna sono vicine a quelle ideali del 100%.

La percentuale di messaggi consegnati è un parametro importante da tenere in considerazione, poichè ci fa capire la propagazione del messaggio, e mostra quanto il messaggio sia conosciuto dai veicoli. Percentuali basse possono implicare la non segnalazione di un pericolo per alcuni veicoli, e quindi una utilità sicuramente inferiore delle applicazioni di supporto alla guida.

4.3.4 Channel load

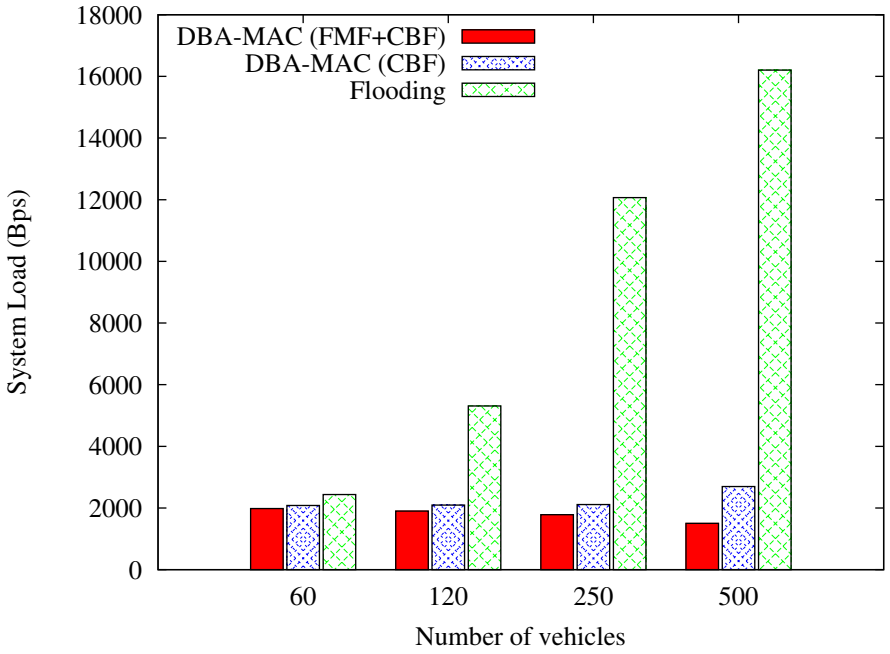


Figura 4.7: Il carico di lavoro del canale

La figura 4.7 mostra il carico del canale, espresso in Bps, al variare dei veicoli. Si nota chiaramente come DBA-MAC utilizzi molto meno il canale rispetto al protocollo di Flooding. Addirittura, l'utilizzo diminuisce all'aumentare dei veicoli nello scenario. Anche in questo caso, come in quello precedente, DBA-MAC dimostra la sua eccellente scalabilità, migliorando addirittura le performance all'aumentare dei veicoli. Questo comportamento è spiegabile con la stabilità della backbone, ovvero meno veicoli abbiamo, più è difficile trovarne due con caratteristiche simili, che quindi riescano a mantenere una backbone attiva per un certo periodo di tempo. Se invece aumentiamo il numero di veicoli, si formano backbone con membri molto più affini, riducendo quindi la necessità di mandare messaggi per la creazione della BACKBONE, e quindi lasciando libero il canale per altri messaggi, come quelli di allarme.

I fatti che entrano in gioco in questo caso sono principalmente due: la lunghezza della backbone e la sua stabilità. Più una backbone è lunga, più viene coperta

un'area grande, e quindi un numero maggiore di veicoli non entra in fase di contesa poichè nella propria area è gia presente una backbone. La stabilità è inoltre un altro parametro fondamentale per caricare meno il canale. Difatti, più una backbone è stabile, meno ci saranno fase di contesa per la creazione di nuove backbone, con l'invio di messaggi di BEACON. Di conseguenza, il canale è lasciato libero per altri utilizzi, come l'invio di messaggi di allarme o altre applicazioni.

4.3.5 Connettività

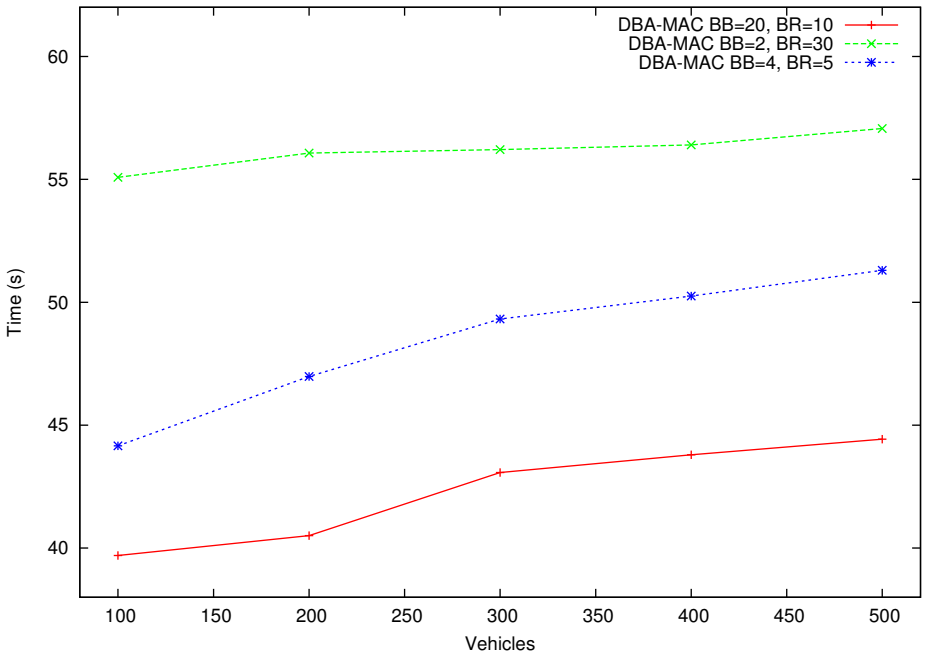


Figura 4.8: La durata media della backbone

La figura 4.8 fa vedere la durata massima della backbone al variare dei veicoli, per diversi tempi di beacon e di refresh. Si nota chiaramente com i tempi cambino drasticamente tra le varie configurazioni, privilegiando quella in cui il tempo di `BEACON` è di 2 secondi, mentre invece il tempo di `REFRESH` è di 30 secondi. Un tempo di `REFRESH` così alto fa entrare nella backbone solamente veicoli che siano ottimi candidati, tralasciando quelli che presentano valori non eccellenti. Questo si traduce, nel tempo, in maggiore stabilità, e di conseguenza in una durata media maggiore. Nel caso invece con un tempo di `BEACON` uguale a 4 secondi, e un `REFRESH` uguale a 5 secondi, le performance sono leggermente peggiori, ma comunque migliori di quando invece il `BEACON` è a 20 secondi. In questo caso infatti il tempo è troppo elevato, e i veicoli che facevano parte della backbone possono essere a una distanza troppo elevata per trasmettere. Dei `BEACON` più frequenti possono invece migliorare la stabilità, cercando di anticipare casi critici come quelli che si possono presentare a distanza di 20 secondi l'uno dall'altro.

Questo grafico ci dice molto di più: ci dice che se è possibile, è sempre meglio scegliere veicoli con caratteristiche simili a quelle del mittente, ma che se i veicoli sono pochi, è sempre meglio sceglierne comunque uno, anche se non con caratteristiche eccelse. Questo potrebbe far pensare che la strada di un protocollo adattivo, che si modifichi in base alle caratteristiche di densità e tipologia di traffico potrebbe migliorare ulteriormente queste performance.

4.3.6 Performance sul monitoraggio

Si vogliono studiare le performance di DBA-MAC nel caso in cui ci sia sopra di esso un'applicazione che faccia monitoraggio del traffico stradale (*TMIS* - Traffic Monitoring and Information System). Ogni veicolo periodicamente manda in broadcast un messaggio, per informare i vicini delle condizioni del traffico e di eventuali punti di interesse. Si considera quindi un pacchetto di 500 bytes e uno scenario con 400 veicoli presenti.

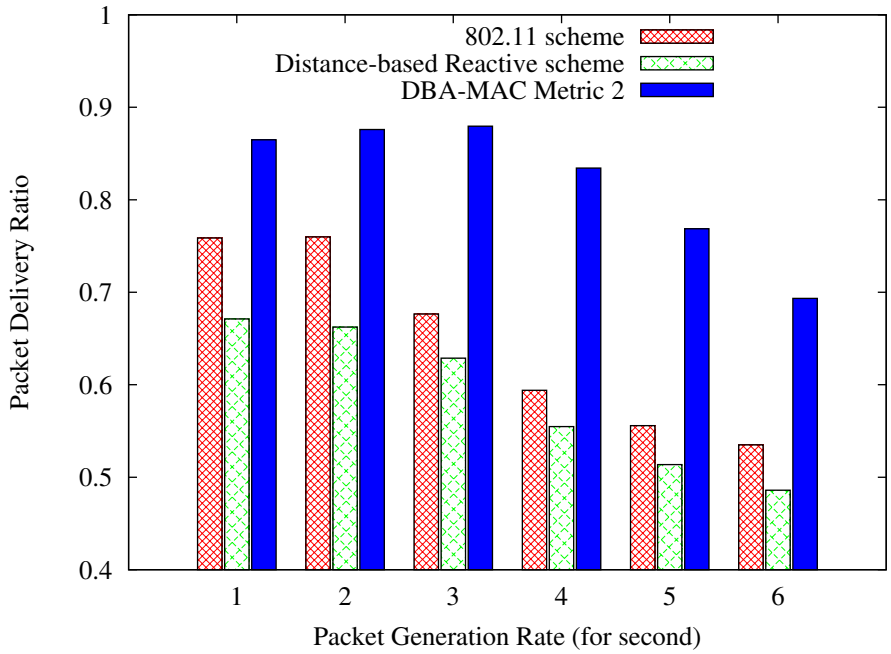


Figura 4.9: La delivery ratio dei pacchetti per l'applicazione di monitoraggio

La figura 4.9 mostra la percentuale di messaggi consegnati rispetto a quelli da consegnare, al variare della frequenza di generazione dei messaggi. Si nota come DBA-MAC con la seconda metrica garantisca performance migliori rispetto agli altri protocolli. La frequenza di generazione dei pacchetti è un'importante parametro per questa valutazione, poichè chiaramente più messaggi vengono generati, più ci potranno essere collisioni e quindi mancate consegne. Si deve comunque tenere presente che per un'applicazione di monitoraggio del traffico, anche una bassa generazione dei messaggi può essere sufficiente per garantire la fruibilità del servizio e la sua utilità.

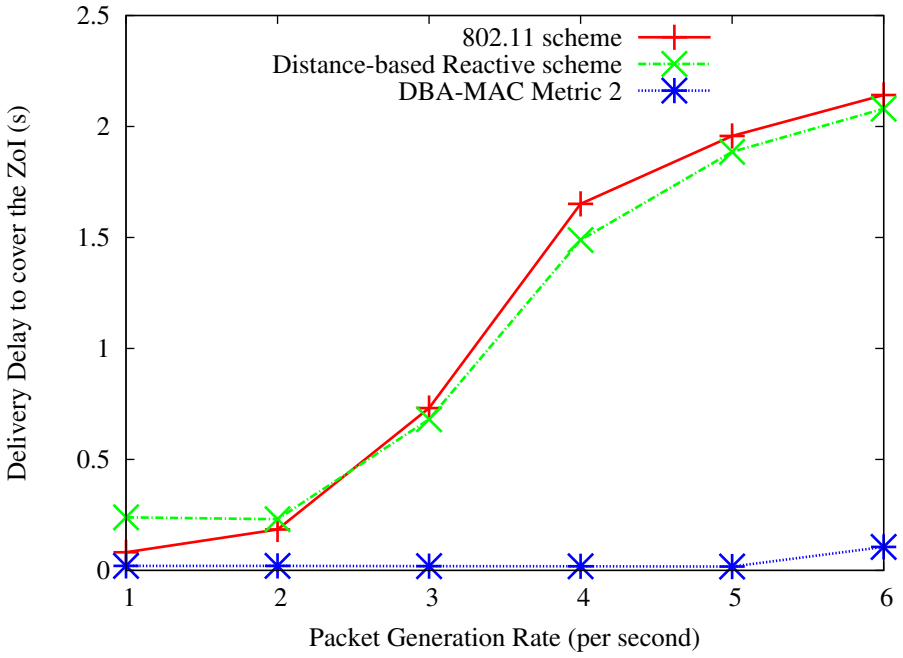


Figura 4.10: Il delivery delay dei pacchetti per l'applicazione di monitoraggio

Il grafico 4.10 mostra invece il ritardo nella consegna dei messaggi per l'applicazione TMIS. È lampante vedere come ne il protocollo di flooding con l'802.11, ne il sistema basato sulla distanza garantiscano performance accettabili. Performance che invece sono garantite da DBA-MAC con l'utilizzo della seconda metrica, in cui si nota come il ritardo sia sempre bassissimo, anche con frequenze di generazione dei messaggi particolarmente elevate. La seconda metrica, infat-

ti, cercando di massimizzare la qualità del link ad ogni hop, riesce a garantire performance eccellenti nella comunicazione.

4.3.7 Performance audio

Si consideri un'applicazione audio, che ha quindi bisogno di trasmettere stream audio ai veicoli della RZ. Si consideri quindi un'applicazione VOIP, che trasmette a una velocità di 64 kbps, con ogni singolo messaggio che ha un payload di 160 bytes, che corrisponde a un durata vocale di 20ms.

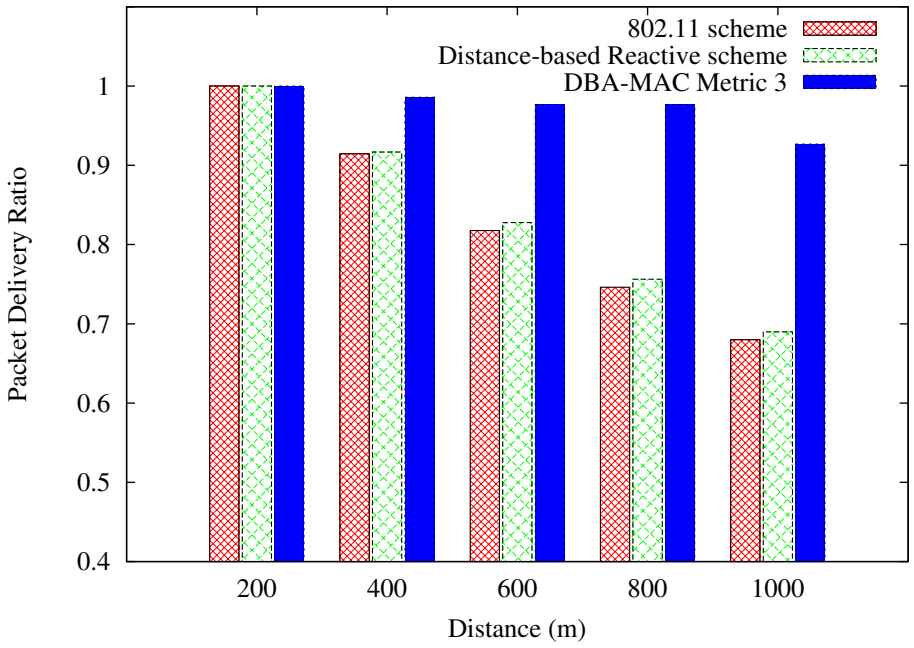


Figura 4.11: La delivery ratio dell'applicazione audio

La figura 4.11 mostra la percentuale di messaggi consegnati rispetto a quelli da consegnare. Si nota chiaramente come DBA-MAC con la terza metrica garantisca eccellenti performance, a differenza del protocollo basato sulla distanza o dell'802.11 con il flooding. Al massimo infatti DBA-MAC con la terza metrica perde il 10% dei messaggi a una distanza di 1000m. Questi fatti sono dovuti ad una assenza di contesa grazie al meccanismo FMF, e grazie anche a un bi-

lanciamento delle ritrasmissioni che permette elevate percentuali di consegna dei messaggi.

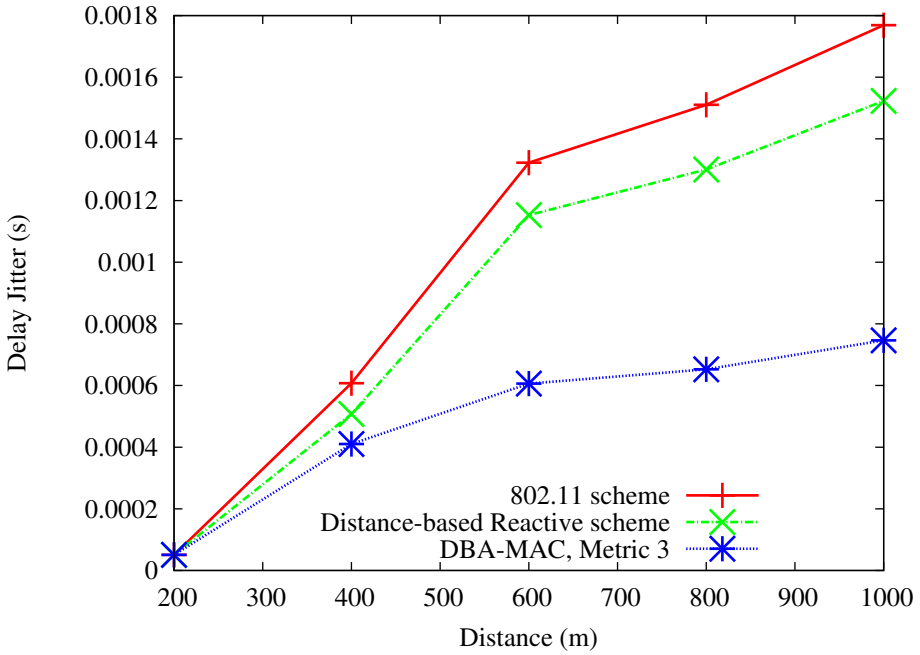


Figura 4.12: Il delay jitter dell'applicazione audio

La figura 4.12 mostra il delay jitter per l'applicazione studiata, ovvero il ritardo tra i pacchetti. Questa è una misura molto importante, poichè permette di capire se ci saranno o meno interruzioni nella comunicazione, particolarmente fastidiose per l'audio. La figura 4.12 mostra come DBA-MAC abbia il minor delay jitter tra i protocolli studiati, questo poichè i messaggi sono spesso spediti dagli stessi BM, e quindi hanno circa gli stessi ritardi. Per il meccanismo a distanza e per il flooding invece questo ritardo cresce parecchio, dato che all'aumentare del numero dei veicoli aumentano anche le potenziali collisioni e quindi i ritardi.

4.3.8 Performance Video

Si considera un'applicazione video, che debba fare lo streaming di un video a tutti i veicoli della zona di interesse. È stato usato come per i test un video

con codifica standard MPEQ QCIF [6], a 384 Kbps e 30 frame per secondo. I frame ricevuti sono bufferizzati, e visualizzati con il ritardo di un secondo sul video del veicolo. Per la valutazione della qualità è stato usato Evalvid [15], che offre un metodo per dare una stima della qualità del video ricevuto sulla base della valutazione PSNR dei frame. Si valutano quindi le differenze tra il video ricevuto e quello mandato, per dare una stima della bontà dei dati in possesso del ricevente.

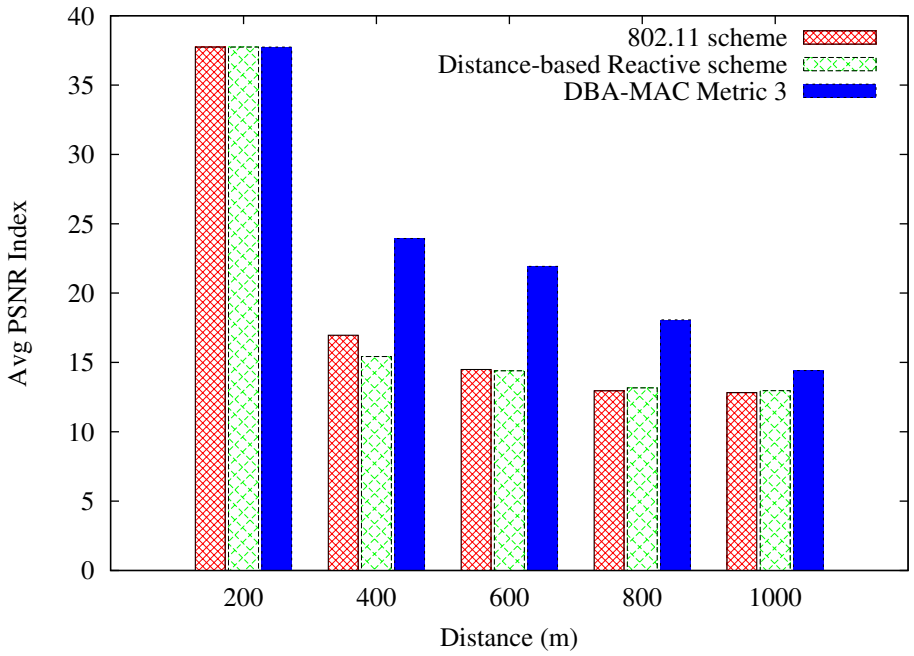


Figura 4.13: Il PSNR valutato per i tre diversi protocolli

La figura 4.13 mostra appunto i livelli calcolati di PSNR per i tre protocolli presi in esame, ovvero quello base ottenuto mediante l'802.11, quello basato su distanza e DBA-MAC con la terza metrica. Mostra quindi come, variando la distanza dal mittente, varino anche i livelli di PSNR relativi. A una distanza relativamente bassa, ovvero 200 metri, tutti e tre i protocolli si comportano bene, offrendo un'ottima qualità video, ma più ci si allontana dalla sorgente e più DBA-MAC ottiene performance migliori degli altri.

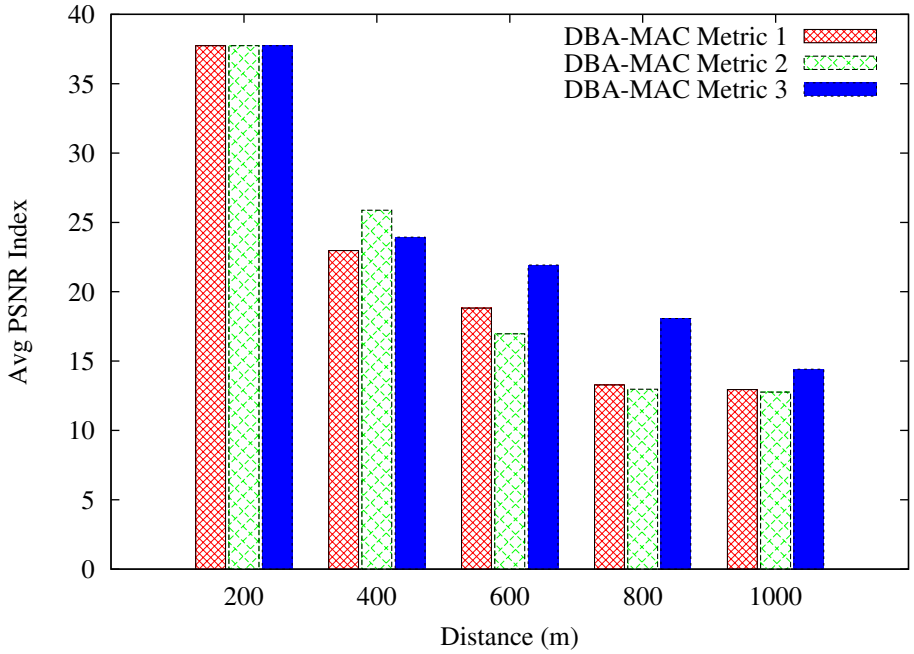


Figura 4.14: Il PSNR valutato per le tre diverse metriche possibili di DBA-MAC

La figura 4.14 mostra la variazione del PSNR a seconda della metrica utilizzata per DBA-MAC. Il grafico conferma come la terza metrica offra performance migliori delle altre, anche per veicoli vicini al limite spaziale considerato di 1 km di distanza dalla sorgente.

Si può quindi dire che DBA-MAC, con la terza metrica presa in esame, supporti le comunicazioni multimediali meglio degli altri protocolli presi in esame, migliorando la qualità del video ricevuto e offrendo quindi un'esperienza migliore per chi lo guarda.

Capitolo 5

Conclusioni e sviluppi futuri

In questo lavoro di tesi sono state presentate le VANET, spiegandone le caratteristiche e quali possono essere le loro potenzialità per migliorare la qualità della guida su strada. È stato poi presentato DBA-MAC, un protocollo che si basa su backbone per permettere un veloce relay dei messaggi in ambito veicolare, confrontandolo con altre soluzioni anche tramite l'utilizzo di simulazioni.

I risultati fanno vedere come DBA-MAC permetta di ottenere una migliore velocità di consegna dei messaggi, unitamente a un utilizzo minore del canale, fatto dovuto al minor numero di collisioni a livello MAC e all'attesa minore per la spedizione dei messaggi.

Nel futuro si pensa di estendere DBA-MAC, per valutare altre politiche di scelta dei membri della backbone, al fine di migliorare ancora di più la stabilità e la durata della stessa. La stabilità e la bontà della backbone sono infatti fondamentali per mantenere le performance del protocollo a un buon livello, come mostrato dal grafico 4.8. Trovare un buon bilanciamento tra stabilità della backbone, veicoli più adatti a farne parte e overhead sarebbe un passo avanti per quanto riguarda le performance.

Penso che si potrebbe anche valutare l'impatto e l'utilità di inserire un meccanismo grazie al quale un nodo, se capisse di essere troppo vicino a un altro, possa togliersi dalla backbone, evitando relay altrimenti poco distanti nello spazio. Se si pensa infatti all'effetto provocato dai semafori, in cui veicoli lontani possono

avvicinarsi, si capisce come gli eventi esterni possano far decadere l'utilità di una specifica backbone.

Inoltre, penso possa essere utile far vedere come si comporta DBA-MAC in presenza di altri tipi di messaggi oltre a quelli di allerta, inserendo anche un meccanismo di priorità dei messaggi. Questo permetterebbe di avere comunicazioni di diverso tipo sulla rete, lasciando però spazio a messaggi di maggiore priorità come ad esempio indicazioni di veicoli fermi qualora ce ne fosse bisogno. Le applicazioni veicolari infatti sono ancora agli albori, ed è facile pensare che in un prossimo futuro ne esistano di diverse tipologia, ognuna con una priorità diversa rispetto alle altre. Fornire un meccanismo che permetta una rapidità di consegna differenziata rispetto alla tipologia di messaggio permetterebbe anche l'introduzione di queste applicazioni senza far peggiorare le performance di altre con priorità maggiori.

Capitolo 6

Ringraziamenti

Alla fine di un percorso, ci sono sempre persone da ringraziare. Persone che ti hanno aiutato materialmente in quello che hai fatto, persone che non ti hanno aiutato materialmente ma sapevi che erano comunque sempre presenti per te, persone che ti sono vicine.

Fortunatamente, ho molte persone da ringraziare, ognuna per un motivo particolare che mi ha permesso di arrivare fin qui.

Ringrazio quindi il Prof. Luciano Bononi, che ritengo mi abbia aiutato e spronato a migliorarmi e a raggiungere gli obiettivi descritti nella tesi. La sua disponibilità e le sue indicazioni mi hanno fatto capire come affrontare un lavoro di questo tipo, e correggere il tiro ogniqualvolta ce ne fosse bisogno.

Ringrazio il Dott. Marco Di Felice, che assieme al Prof. Luciano Bononi mi ha seguito dall'inizio alla fine della tesi, rispondendo a qualunque domanda mi potesse venire in mente e dandomi nuovi spunti per progredire nel lavoro. Grazie a loro da questo lavoro di tesi sono uscite due pubblicazioni, una approvata e una in fase di pubblicazione, e quindi li devo ringraziare anche per avermi dato la possibilità di fare questo. Spero di poter continuare questo ed altri lavori con loro durante il periodo di dottorato di ricerca.

Ringrazio la mia famiglia, che mi ha sempre supportato in qualunque scelta prendessi, anche se non la ritenevano la migliore. Ringrazio quindi Serena, che mi ha supportato (e sopportato!) durante tutto il periodo della Laurea Magistrale,

aiutandomi a prendere scelte complesse e nella vita di tutti i giorni. Ringrazio mio papà Claudio e mia mamma Stefania, che hanno l'onere di avere un informatico in casa da 26 anni, ma che mi hanno sempre spronato a fare ciò che più mi piacesse, anche se magari non poteva sempre sembrare la scelta migliore.

Ringrazio i miei parenti, che nonostante gli argomenti da me studiati non fossero i più comprensibili, mi hanno sempre domandato cosa facessi all'università, quali fossero i miei obiettivi, le mie aspirazioni. Li ringrazio perchè si interessavano, e si interessano tutt'ora, ai progressi che faccio.

Ringrazio tutti i miei amici, che hanno saputo farmi staccare dai problemi di tutti i giorni con una risata, e hanno saputo aspettarmi quando i ritmi dell'università si facevano più pressanti togliendomi del tempo libero. Grazie a Cecco e Sara, a Zambo e Ari, a Munna e Matti.

Ringrazio tutti i compagni dell'università che mi hanno accompagnato dall'inizio fino ad oggi, condividendo passioni, soddisfazioni e delusioni. Grazie quindi a Uovo, Toshi, Bigo, Tappo, Jmondi, Sem, Matte, GG, Mone, Bagna, Lara, Giulio, Bea, Silvia, AP, Cruz, Ax, e a tutti quelli che non sono in questa lista, voi che lo capite vi dico *.grazie();.

Bibliografia

[1] Wireless lan medium access control (mac) and physical layer (phy) specifications: Wireless access in vehicular environment, 2009.

[2] Ian F. Akyildiz, Won-Yeol Lee, and Kaushik Roy Chowdhury. Crahns: Cognitive radio ad hoc networks. 2009.

[3] Mohammad S. Almalag and Michele C. Weigle. Using traffic flow for cluster formation in vehicular ad-hoc networks. 2010.

[4] Luciano Bononi, Marco Di Felice, and Sara Pizzi. Dba-mac: Dynamic backbone-assisted medium access control protocol for efficient broadcast in vanets. *Journal of Interconnection Networks*, 10(4):321–344, 2009.

[5] A. Casteigts, A. Nayak, and I. Stojmenovic. Communication protocols for vehicular ad hoc networks. *Wireless Communication and Mobile Computing*, 11(5):567–582, May 2011.

[6] Chih-Ming Chen, Chia-Wen Lin, Hsiao-Cheng Wei, and Yung-Chang Chen. Robust video streaming over wireless lans using multiple description transcoding and prioritized retransmission. *Journal of Visual Communication and Image Representation*, 18(3):191 – 206, 2007.

[7] Mohamed oussama Cherif, Sidi Mohammed Senouci, and Bertrand Ducourthial. A new framework of self-organization of vehicular networks. 2009.

[8] Ameneh Daeinabi, Akbar Ghaffar Pour Rahbar, and Ahmad Khademzadeh. Vwca: An efficient clustering algorithm in vehicular ad hoc networks. 2011.

[9] Marco Di Felice, Luca Bedogni, and Luciano Bononi. Dynamic backbone for fast information delivery in vehicular ad hoc networks: An evaluation study. In *Proc. of ACM PEWASUN*, 2011.

[10] Marco Di Felice, Kaushik Roy Chowdhury, and Luciano Bononi. Analyzing the potential of cooperative cognitive radio technology on inter-vehicle communication. In *Proc. of IFIP Wireless Days , pp. 1-6, Venice*, 2010.

[11] ACI e ISTAT. Studi e ricerche. dati sugli incidenti stradali rilevati nel 2009.

[12] Peng Fan. Improving broadcasting performance by clustering with stability for inter-vehicle communication. In *Proc. of IEEE VTC-Spring*, 2007.

[13] Elena Fasolo, Andrea Zanella, and Michele Zorzi. An effective broadcast scheme for alert message propagation in vehicular ad hoc networks. In *ICC 2006*, 2005.

[14] Roberta Fracchia and Michela Meo. Analysis and design of warning delivery service in intervehicular networks. 2008.

[15] C. H. Ke, C. K. Shieh, W. S. Hwang, and A. Ziviani. An evaluation framework for more realistic simulations of mpeg video transmission. *Journal of Information Science and Engineering*, 24(2):425–440, 2008.

[16] Maria Kihl, Mihail L. Sichitiu, and Harshvardhan P. Joshi. Design and evaluation of two geocast protocols for vehicular ad-hoc networks.

[17] Gokhan Korkmaz, Eylem Ekici, Fusun Ozguner, and Umit Ozguner. Urban multi-hop broadcast protocol for inter-vehicle communication systems. In *Proc. of ACM VANETs, pp. 76-85*, 2004.

[18] Daniel Krajzewicz, Michael Bonert, and Peter Wagner. The open source traffic simulation package sumo. 2005.

[19] Cuiran Li and Chengshu Li. Opportunistic spectrum access in cognitive radio networks. In *IJCNN*, pages 3412–3415. IEEE, 2008.

[20] Sze-Yao Ni, Yu-Chee Tseng, Yuh-Shyan Chen, and Jang-Ping Sheu. The broadcast storm problem in a mobile ad hoc network. *Mobicom*, 1999.

[21] C. E. Palazzi, M. Roccetti, and S. Ferretti. An intervehicular communication architecture for safety and entertainment. 11(1):90–99, 2010.

[22] Claudio E. Palazzi, Stefano Ferretti, Marco Roccetti, Giovanni Pau, and Mario Gerla. How do you quickly choreograph inter-vehicular communications? a fast vehicle-to-vehicle multi-hop broadcast algorithm, explained. 2007.

[23] Bryan Parno and Adrian Perrig. Challenges in securing vehicular networks. 2006.

[24] Francisco J. Ros, Pedro M. Ruiz, and Ivan Stojmenovic. Reliable and efficient broadcasting in vehicular ad hoc networks. 2009.

[25] Jagruti Sahoo, Eric Hsiao Kuang Wu, Pratap Kumar Sahu, and Mario Gerla. Bpab: Binary partition assisted emergency broadcast protocol for vehicular ad hoc networks. In *Proc. of IEEE ICCCN*, 2009.

[26] Christoph Sommer, David Eckhoff, Reinhard German, and Falko Dressler. A computationally inexpensive empirical model of ieee 802.11p radio shadowing in urban environments. Technical report, Friedrich-Alexander-Universität Erlangen-Nürnberg, Department Informatik, 2010.

[27] Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

[28] Ozan K. Tonguz, Nawaporn Wisitpongphan, Jayendra S. Parikht, Fan Bait, Priyantha Mudaliget, and Varsha K. Sadekart. On the broadcast storm problem in ad hoc wireless networks. 2006.

[29] Yu-Chee Tseng, Sze-Yao Ni, Yuh-Shyan Chen, and Jang-Ping Sheu. The broadcast storm problem in a mobile ad hoc network. *Wireless Networks*, 8:153–167, 2002.

[30] Yuxuan Wang and Forrest Sheng Bao. An entropy-based weighted clustering algorithm and its optimization for ad hoc networks. In *WiMob'07*, pages –1–1, 2007.

[31] Theodore L. Willke, Patcharinee Tientrakool, and Nicholas F. Maxemchuk. A survey of inter-vehicle communication protocols and their applications. 2009.

[32] Andrea Zanella, Gianfranco Pierobon, and Simone Merlin. On the limiting performance of broadcast algorithms over unidimensional ad-hoc radio networks. 2004.