Alma Mater Studiorum · Università di Bologna

SCUOLA DI SCIENZE
Corso di Laurea Magistrale in Scienze di Internet

Teoria dei Giochi e Multi-Criteria Decision-Making per Reti Mobili Ad-hoc: un Protocollo di Routing

Tesi di Laurea in Teoria dei Giochi

Relatore:
Chiar.mo Prof.
Giovanni Rossi

Presentata da:
Alessandro Di Teodoro

Sessione III
Anno Accademico 2012-2013

Ad Olga e Roberto,

per essere stati il punto di riferimento

sul quale ho sempre potuto affidare me stesso.

Indice

1 Introduzione **1**

2 Le Mobile Ad-hoc Networks **3**

2.1 Le Tipologie di Networking 7

2.2 Problemi e Sfide 10

2.3 Le Applicazioni Possibili 13

2.4 Lo Stack Protocollare 14

2.4.1 I Layer 15

2.4.2 La Sicurezza 17

2.5 L'Instradamento 20

2.5.1 Conoscere la Topologia 22

2.5.2 Inviare i Dati 28

3 Modellare Problemi a più Partecipanti **33**

3.1 I Giochi Strategici 36

3.1.1 La Rappresentazione 37

3.1.1.1 La Forma Normale 40

3.1.1.2 La Forma Estesa 41

3.1.2 I Giochi a Somma Costante 43

3.1.3 L'Informazione 45

3.1.4 Le Strategie 46
3.1.5 I Punti di Equilibrio 49
3.1.5.1 Problemi noti 53
3.2 I Giochi Cooperativi 53
3.2.1 La Funzione Caratteristica 55
3.2.2 Differenza tra Giochi NTU e Giochi TU 58
3.3 L'Integrale di Choquet 61

4 Il Prototipo di Protocollo Proposto 67

4.1 La Motivazione della scelta 68
4.2 Descrizione Introduttiva 69
4.3 La Terminologia 70
4.4 Le Strutture Dati 74
4.4.1 Tabella di Vicinanza Nodi 74
4.4.2 Tabella di Vicinanza Cluster 75
4.5 I Pacchetti Base 77
4.5.1 Pacchetti di Topologia 77
4.5.1.1 Hello Message 78
4.5.2 Pacchetti di Routing 80
4.6 Le Operazioni Basilari 81
4.6.1 La Clusterizzazione 81
4.6.2 Il Routing di Dati 90
4.7 Le Operazioni di Ottimizzazione 95
4.7.1 Stimolo della Partecipazione 95
4.7.2 Diminuzione dei Gateway 103

5 Conclusioni 109

A Formazione di Cluster in MANet's 111

Bibliografia **117**

Sitografia **121**

Elenco delle figure

2.0.1 Una generica MANet di Terza Generazione 5
2.1.1 Una Rete di tipo Infrastructure 8
2.1.2 Un Multihop a tre salti 9
2.3.1 Una esempio di VANet 14
2.4.1 Il Modello OSI 15
2.5.1 Distance Vectors di una generica MANet 23
2.5.2 Applicazione del Routing di Bellman-Ford 24
2.5.3 Count To Infinity in DSDV 25
2.5.4 Esempio di stato dei link in una MANet 26
2.5.5 Propagazione informazione in Link State con un nuovo link . . . 27
2.5.6 Classificazione di Protocolli in MANet 29

3.0.1 Città a confronto per criteri 34
3.1.1 Dilemma del Prigioniero 38
3.1.2 Dilemma del Prigioniero in Forma Estesa 42
3.1.3 Informazione Completa contro Informazione Incompleta 45
3.2.1 Dilemma del Prigioniero in Gioco Coalizionale 60

4.3.1 Direzionalità dei link wireless 73
4.4.1 Rappresentazione dei link protocollari in bit 75

4.4.2 Cluster in collegamento tramite L-Gateway 76
4.5.1 Parte di un Hello Message con Indirizzi a 32 bit 80
4.6.1 Rappresentazione grafica dello Choquet Score 85
4.6.2 Clusterizzazione di una MANet secondo lo Choquet Score 88
4.6.3 Collegamento bi-direzionale tramite links mono-direzionali 91
4.6.4 Procedura di Discovery per un determinato cammino 94
4.7.1 Paths di una MANet con relativo Grafo di Dipendenza 101
4.7.2 Eliminazione dei Gateway meno necessari 104

Elenco delle tabelle

2.1 Comparazione delle categorie principali di Protocolli MANet . . 30

2.2 Principali Protocolli di Routing per MANet's 31

3.1 Dilemma del Prigioniero in Forma Normale 41

3.2 Un esempio di Gioco a Somma Costante 43

3.3 Gioco della Morra Cinese in Forma Normale 44

3.4 Rappresentazione della Morra Cinese tramite Payoff Costante . . 44

3.5 Dominanza Debole e Stretta di una Scelta 48

3.6 Solution Concept in Battaglia dei Sessi 50

3.7 Solution Concept in Dilemma del Prigioniero 51

3.8 Gioco Coalizionale con Pagamenti Laterali a tre giocatori 57

3.9 Comparazione di Strumenti Multi-criterio 64

4.1 Ordinamento nodi per Score in supporto alla figura 4.6.2 87

4.2 Dilemma del Prigioniero con Punizioni 102

Capitolo 1

Introduzione

La *Teoria delle Decisioni*, disciplina relativamente recente, ha dimostrato già più volte la sua versatilità per quanto riguarda gli scenari ai quali essa può essere applicata. Ufficialmente, questa materia di studio, nasce attorno agli anni trenta con l'obiettivo di coadiuvare gli sforzi bellici degli Alleati dovuti alla seconda guerra mondiale. Con il passare del tempo e con l'intensificarsi dell'interesse verso di essa, come appena anticipato, la *Ricerca Operativa* (altro nome della Teoria delle Decisioni) ha tentato di abbracciare ambiti totalmente diversi, cercando di spendersi in modo tale da risolvere problemi assai eterogenei, a volte anche diametralmente opposti fra loro; tra questa grande raccolta ovviamente, non sarebbero mai potuti mancare quelli in veste informatica. La *RO* (Ricerca Operativa) si adopera nello scovare soluzioni di tipo simulativo e di ottimizzazione, questa duplice natura la rende appunto complice perfetta nell'informazione automatica; cosa può esserci di meglio infatti, di specifiche *routine*, descritte in maniera non ambigua, in grado di stimare e far fronte a circostanze dal futuro nebuloso?

In questa tesi si andranno ad analizzare alcuni argomenti della *Teoria dei*

Giochi e della *Multi-criteria Decision-making* (materie di studio della *RO* appunto); questo ci darà le competenze necessarie per osservare certi quesiti sotto un'ottica diversa. Più precisamente, nell'elaborato, l'ambito applicativo scelto è quello delle *Reti Mobili Ad-hoc*. Queste ultime risultano essere di elevato interesse sotto lo studio e la previsione delle scelte perché, essendo composte da nodi che possono attuare un comportamento *selfish*, costituiscono una buona base sulla quale operare in termini di ricerca.

La tesi sarà quindi organizzata in tre principali capitoli (contornati da questa introduzione e dalle conclusioni finali), essi riguarderanno:

- Una illustrazione delle *Mobile Ad-hoc Networks* (**MANet's**);
- Una presentazione degli "strumenti" *Teoria dei Giochi* (**TdG**) e *Multi-criteria Decision-making* (**MCDM**);
- Una proposta di protocollo per la tipologia di rete esaminata.

Capitolo 2

Le Mobile Ad-hoc Networks

In questo capitolo cercheremo di capire al meglio il funzionamento di particolari reti di comunicazione, più nello specifico analizzeremo le cosiddette MANet's.

Questi sistemi, come ci suggerisce l'acronimo (*Mobile Ad-hoc Network*), sono progettati per far dialogare tra loro un insieme di nodi, più o meno vasto, che non debba rispettare le due caratteristiche principali del networking classico. Ci aiutano a capire meglio questa affermazione le parole *Mobile* ed *Ad-hoc*, rispettivamente esse vanno ad indicare:

- **Presenza di device mobili**:

  Come noto, esistono dispositivi (es. laptop, gps, etc.) che massimizzano la propria utilità solo se utilizzati in movimento, si pensi ad esempio a strumenti per lo studio della telemetria o la semplice comunicazione tra più persone attraverso apparecchi non fissi. È necessario, in questi casi, adoperarsi per riuscire a far interagire nodi che non seguono pattern di movimento ben definiti, ma continuano a muoversi nello spazio senza alcuna regola. Questo rende particolarmente difficile tentare di raggiungere un'apparecchiatura con l'informazione ad essa destinata e, non meno ar-

duo, il compito di controllare se l'operazione di invio sia andata a buon fine;

- **Configurazione in assenza di infrastrutture definite a priori e staticamente**:

  Parlando di mobilità dei dispositivi, nelle MANet's, non ci riferiamo ad un piccolo insieme dei nodi della stessa rete, ma praticamente a tutti i device che fanno parte di essa. Ciò significa non avere mai né sicurezza sul fatto che un'apparecchiatura, partecipante alla maglia di collegamenti, possa trovarsi per un determinato tempo nella stessa posizione né, tantomeno, che rimanga in un range sferico delimitato da un certo raggio. Questo porta quindi all'impossibilità di definire, a priori ed in fissa maniera, uno smistatore di pacchetti.

Proprio queste due caratteristiche, coadiuvate al loro intrinseco legame, vanno a sancire la netta divisione tra l'*Infrastructure mode* e la tipologia di comunicazione MANet's (nel networking wireless). Naturalmente esistono anche tipologie di rete che ammettono dispositivi fissi e configurazioni senza instradatori ben definiti; anche queste ultime possono essere trattate, in alcuni casi e se lo si ritiene necessario, come *Mobile Ad-hoc Networks*.

Ma come può essere definito, in maniera più tecnica, un insieme di elementi che si scambiano dati secondo questo standard?

"*A mobile ad hoc network consists of wireless hosts that may move often. Movement of hosts results in a change in routes, requiring some mechanism for determining new routes*"[13]. Questa è una delle tante definizioni che possiamo utilizzare per descrivere i sistemi (un esempio in figura 2.0.1) a cui siamo interessati in questa tesi, comunque, qualsiasi spiegazione venga fornita, il concetto principe risalta in maniera chiara. Il problema è riuscire a creare un set di regole che permetta la comunicazione all'interno di un insieme, in cui gli elementi e ov-

viamente anche i collegamenti tra gli stessi sono inseriti ed eliminati in maniera dinamica e non predicibile.

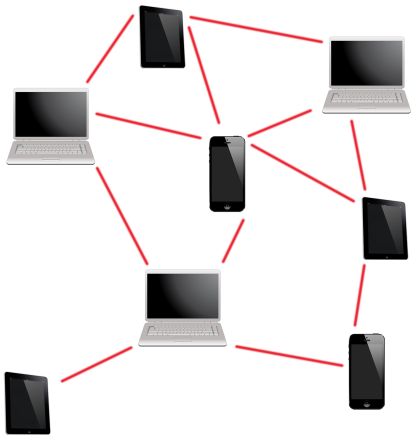


Figura 2.0.1: Una generica MANet di Terza Generazione

Definite le reti di nostro interesse, anche se in maniera non approfondita, è opportuno, prima di continuare verso il nostro obiettivo, incentrarle brevemente nella linea temporale, per riuscire a capire al meglio lo sviluppo che hanno subito. Riusciamo a classificare nettamente tre periodi storici basandoci sul progresso che le *Mobile Ad-hoc Networks* hanno avuto negli anni e, tramite questi, caratterizziamo tre generazioni, rispettivamente:

- *First Generation*
- *Second Generation*
- *Third Generation*

First Generation

Parlando della tipologia di rete a cui siamo interessati riusciamo a risalire addirittura agli anni '70, più precisamente al 1977, quando la *Darpa* (Defense Advanced Research Projects Agency) decise di creare una rete basata sul pac-

ket switching *Packet Radio*. Quest'ultimo non era altro che una tecnologia in grado di applicare lo scambio dei pacchetti alle reti point-to-point che si stavano sviluppando in quell'epoca, riusciva a garantire prestazioni migliori tramite accessi multipli al canale radio garantendo un numero di *subscribers* elevato.[12]

Utilizzando il protocollo *Aloha* prima e, successivamente, il *CSMA* (Carrier sense multiple access), assieme al *Packet Radio*, l'agenzia americana creò così la PRNet, una delle prime reti *multihop* ed *Ad-hoc* senza fili, considerata la madre delle presenti MANet's.

Second Generation

Per ciò che riguarda la seconda generazione troviamo due sottoinsiemi. Il primo indicativamente attorno agli anni '80 con il programma *SURAN* (Survivable Adaptive Radio Networks), che migliorò le prestazioni radio in termini di costo/sicurezza utilizzando un protocollo di packet switching in reti composte da dispositivi mobili senza un'infrastruttura di riferimento. Mentre il secondo, alla fine degli anni '90, non fu altro che l'avvento della modalità *Ad-hoc* nelle schede di rete dei laptop. Il termine per la prima volta veniva usato in concomitanza allo standard wireless 802.11 e la nuova tecnologia iniziava ad interessare anche le grandi industrie dato che ormai era sfruttata anche a fini commerciali.

Third Generation

L'ultima generazione che, in ordine di tempo, ritroviamo nelle MANet's è la terza, nonché quella in cui ora siamo immersi.

L'immissione, sempre più elevata, di dispositivi mobili nel mercato consumer ha reso il costo di quest'ultimi relativamente esiguo, permettendo la proliferazione di applicativi, e quindi annessi studi sulla parte comunicativa, da utilizzare nelle tipologie di rete alle quali ci stiamo interessando. Classici esempi sono le

reti mobili veicolari o semplicemente le reti che vengono create a scopo di invio pacchetti tra un laptop e una moltitudine di smartphone.

2.1 Le Tipologie di Networking

Se la caratteristica di mobilità inciderà, come è possibile intuire, nelle riconfigurazioni che la rete andrà ad operare a causa degli spostamenti, allora l'altro connotato delle *MANet's* (quindi la modalità *Ad-hoc*) imprimerà la sua egemonia anche in fase di *setting-up* del sistema. È bene quindi descrivere, in maniera più chiara possibile, le differenze che intercorrono tra una rete che fa riferimento ad un'infrastruttura ed un'altra priva di quest'ultima.

Distinguiamo quindi due principali tipi di networking nella scena di comunicazione di rete odierna, queste tipologie vanno sotto il nome di:

- *Infrastructure Networking*
- *Ad-hoc Networking*

Infrastructure Networking

Come già detto, in reti di questo genere (figura 2.1.1), si fa sempre riferimento ad un impianto centrale per quel che riguarda lo smistamento di informazione nei vari nodi. In questi sistemi, il device che vuole entrare a far parte dello scambio di dati è costretto inizialmente a fare una richiesta di accesso alla rete; questa richiesta serve, oltre che a fornire l'autorizzazione necessaria al terminale, a dare un *identificativo univoco* a quest'ultimo, in modo da poter essere raggiunto da un istanza a lui direzionata.

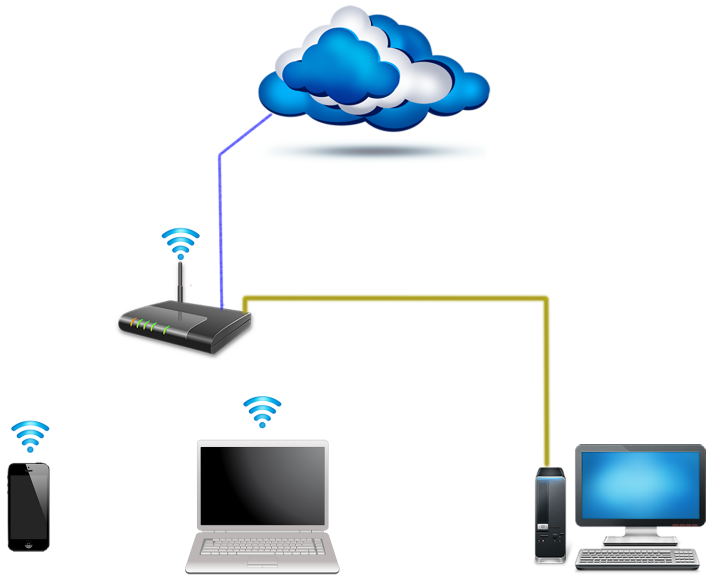


Figura 2.1.1: Una Rete di tipo Infrastructure

Cosa è però e come viene gestito questo identificativo univoco? Questo termine non identifica altro che un indirizzo con il quale è possibile fare riferimento ad una macchina, parliamo quindi di *indirizzo IP* o *Indirizzo Mac*. Nel primo caso siamo nell'ambito dei dispositivi di rete detti *router*, questi dispositivi inviano dati basandosi sul *Terzo Layer* del modello *OSI* che appunto corrisponde al livello *IP* dello stack protocollare *TCP/IP*. Mentre, nel secondo caso, viene utilizzato il *Secondo Layer* (quindi quello di *Collegamento*) attraverso dispositivi quali *Bridge* e *Switch*.[25]

Da ciò che abbiamo detto è ovvio immaginare che la topologia di rete, ad un certo istante, non sia nota ad alcun nodo; infatti ogni partecipante attivo è informato, tramite le infrastrutture, solo ed esclusivamente degli indirizzi che può raggiungere in richiesta (o risposta). La grande maglia viene tenuta in piedi quindi dai pochi dispositivi che abbiamo appena menzionato. Sono que-

sti infatti gli unici a poter "raggiungere geograficamente" un terminale voluto, interrogando le proprie tabelle (dove hanno fatto store degli indirizzi di tutti i collegamenti accettati).

Ad-hoc Networking

Limitarsi a definire un sistema di questo genere affermando semplicemente che esso è decentralizzato può essere fortemente riduttivo. Dobbiamo specificare come avvengono trasmissione e ricezione dei pacchetti. In queste reti (solitamente non molto estese) si ragiona in termini di *path*, ogni nodo conosce la posizione dei suoi vicini e utilizza i percorsi per arrivare a contattare un qualsiasi partecipante. Il concetto centrale dunque è l'*hop*, nient'altro che la nozione di "salto" applicata alle reti.

Immaginando di vedere il sistema come un gigantesco grafo che si compone di nodi (terminali connessi) ed archi (collegamenti tra i terminali), definiamo l'*hop* come un intero che ci informa sul numero di archi che dobbiamo percorrere per far sì che un *packet* vada dal mittente al destinatario. Ne consegue, in maniera del tutto ovvia, che queste reti ammettono il *multihop* (nel esempio mostrato in figura 2.1.2 è raffigurato questo concetto, il primo nodo a sinistra non è nel range wireless del primo a destra ma, attraverso tre archi, riesce comunque a far giungere a destinazione l'informazione voluta).

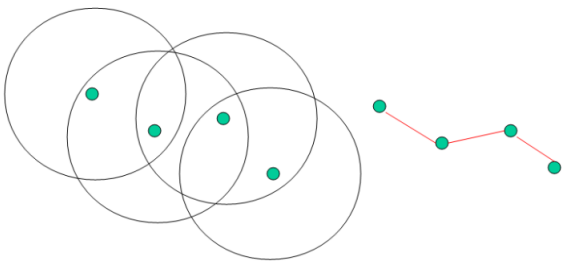


Figura 2.1.2: Un Multihop a tre salti

Non è difficile comprendere che le *challenges* maggiori che questa rete pone sono: la *discovery* (con eventuale abbattimento del suo costo) e la raggiungibilità dei vari terminali che ne fanno parte. L'esempio più banale che si può fornire è il classico *algoritmo di Flooding*[1] (magari modificato in maniera da arrestarsi al verificarsi di alcune clausole).

2.2 Problemi e Sfide

Adesso che, almeno in via teorica, abbiamo la padronanza sul concetto e sul funzionamento base di una *Mobile Ad-hoc Network*, possiamo addentrarci nei problemi e nelle *challenges* che l'impianto di comunicazione, del quale stiamo nutrendo interesse, ci pone. Come ovvio che sia, qualsiasi hardware o software informatico deve essere valutato secondo due tipologie di problematiche che possono emergere in base alla sua realizzazione. Esistono infatti *challenges* di tipo critico e altre di ottimizzazione. Le prime sono necessarie affinché l'organismo progettato sia in grado materialmente di funzionare, le seconde invece garantiscono che questo "risponda" alle richieste pervenute rispettando alcuni standard qualitativi (logicamente facciamo riferimento agli scontatissimi *costo* e *velocità di risposta*). Le classi che andremo a tracciare saranno quindi quella delle *critic features* e quella delle *qualitative feature*s. Andiamo ad elencarle per farci un'idea degli scogli che le *MANet's* pongono ogni giorno a chi tenta di creare, o migliorare, le operazioni di *routing* e *discovery*.

[1] Algoritmo di Flooding - È un particolare algoritmo con il quale si raggiungono tutti i nodi di un grafo a partire da una radice scelta.

Critic Features

Join La dinamicità in queste reti non è da intendersi solo ed esclusivamente nella possibilità di movimento dei nodi che ne fanno parte. Dobbiamo considerare anche i terminali che desiderano entrare a far parte del sistema quando questo risulta già attivo. Ciò apre tutta una serie di problemi riguardanti le possibili riconfigurazioni che dovranno essere operate in un certo istante di tempo;

Depart-Settling Per far iniziare la comunicazione dovremmo fornire un setting iniziale alla rete. Questo non va creato con algoritmi di riconfigurazione, che hanno un numero di variabili relativamente piccolo, ma analizzando l'impostazione migliore che può essere data a tutti gli attori che vanno a formare l'ecosistema (in modo tale da partire con la configurazione migliore possibile);

Recovery Una *MANet* deve avere una ripresa adeguata, ovvero deve riuscire a riconfigurarsi in maniera tale da servire tutti i nodi che rientrano nel range di raggiungibilità, quando un terminale esce da questa o si muove da una posizione nella quale funge da tramite (ovviamente anche al costo di aumentare il numero di *hops*);

Updates Considerando che questo sotto-insieme è costituito dagli aggiornamenti di rete per l'ottimizzazione degli *hops*, si potrebbe tendere ad inserire questa caratteristica nelle *qualitative*, nulla di più sbagliato. Un numero elevato di salti, coadiuvato ad un utilizzo massivo di *flooding*, potrebbe portare ad un eccessivo sovraccarico che mina la già instabile natura della rete;

Scalability Creare uno standard di comunicazione tra pochi terminali non ha alcun senso di esistere. Con le *MANet's* non abbiamo intenzione di riuscire a servire una rete grande potenzialmente come Internet ovviamente,

ma vogliamo comunque una buona scalabilità dei protocolli in modo da utilizzare la decentralizzazione per scopi utili nel mondo reale.

Qualitative Features

Location-Knowledge Riuscire a tenere immagazzinati i *path*, anche magari solo dopo aver applicato un protocollo di tipo *on demand*, significa riuscire a "georeferenziare" un nodo in una rete. Senza bisogno di alcuna spiegazione è lampante come questa caratteristica possa fruttare benefici. Essa si va a tradurre in divulgazione o ricezione fulminea di informazione, ciò perché, riuscendo a "targettizzare" il messaggio secondo il suo destinatario, il flusso non è costretto a seguire l'*algoritmo di flooding*;

Power-Knowledge Parliamo di dispositivi mobili; ovviamente questi sono dotati di una batteria. Dobbiamo considerare la possibilità che questa vada ad esaurirsi. Prendere "per tempo" queste situazioni può alleggerire non di poco i complessi calcoli intrinsechi alla configurazione della rete. Riuscire a capire quale nodo sta per "cedere" da la possibilità di calcolare percorsi alternativi prima che questi si spenga, iniziare a pensare ad un *update*, in un momento in cui la rete non sta trasmettendo e può utilizzare la sua forza-calcolo liberamente, risulta essere la scelta migliore;

Priority Nel sistema viaggiano messaggi. Ovviamente alcuni di questi sono di gran lunga più importanti di altri (ad esempio è più urgente la segnalazione di un nuovo terminale piuttosto che comunicare l'esistenza di *multipath*), questi avvisi degni di più nota, se marcati con priorità elevata, aiutano le rete a fornire un servizio nettamente migliore;

RealTime Alcuni tipi di messaggio ad alta priorità (ad esempio la segnalazione di un nodo sconnesso), se consegnati a *Real Time* fanno sì che la rete esegua migliorie nella gestione della nuova topologia;

Security Per quel che riguarda la sicurezza, decidiamo di inserire questa *feature* nelle qualitative. Questo perché le reti sono solitamente composte da un numero di nodi relativamente esiguo, quindi il pericolo è dato solitamente da chi fa già parte della *MANet*, terminale che comunque è stato vagliato a tempo di *join*.

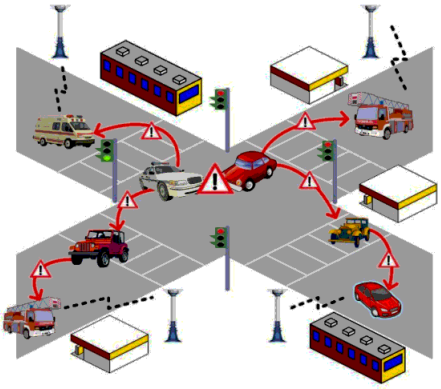
2.3 Le Applicazioni Possibili

Avendo presentato, se pur in via generale, le *Mobile Ad-hoc Networks* e le caratteristiche che esse devono rispettare, possiamo farci un'idea dei campi reali in cui queste possono essere utilizzate. Per la sua mobilità e non centralità la tecnologia *MANet* viene utilizzata, o si tenta di utilizzare, soprattutto nelle operazioni militari, nelle *VANet's* o per operazioni di soccorso dopo disastri. Per quel che riguarda il primo impiego, possiamo immaginare noi stessi l'importanza di queste tipologie di reti (sia in scenari bellici che in operazioni pacifiche).

Più interessanti comunque sono *in primis* le *Vehicular Ad-hoc Networks* (figura 2.3.1), un particolare tipo di rete composta da device mobili dove i nodi sono proprio i veicoli[19]. L'interesse qui è mosso da fatto che, a generare cambiamento, partendo logicamente dal movimento, sono terminali che si muovono a volte anche in gruppo; andiamo quindi ad essere messi davanti a situazioni in cui l'elemento più piccolo del sistema non è un nodo ma un *insieme*. Questo, sotto il punto di vista del *routing*, cambia le carte in tavola permettendo lo studio di protocolli finalizzati a tecniche quali il *clustering*.

In secundis, a livello di interesse, troviamo questo "organismo" di soccorso composto da hardware particolare (ad esempio dei droni) che cerca di dare aiuto in situazioni impredicibili. L'attrattiva in questo caso è data dal fatto che la stessa rete va a disegnare l'adeguato pattern di movimento di un suo oggetto, in modo tale da poterlo sempre raggiungere, almeno con il massimo numero di

hops concessi, oppure da schedulare in maniera precisa il tempo in cui questo scomparirà dal range radio (magari per andare a "servire" una zona inaccessibile, o scomoda, alla totalità della rete).



© University of California - Irvine

Figura 2.3.1: Una esempio di VANet

È d'obbligo specificare che un'applicazione possibile delle *MANet's* può essere la comunicazione e lo sharing *Peer-To-Peer*. L'affermazione potrebbe essere scontata ad una prima lettura, visto che la definizione che abbiamo utilizzato fino ad ora ci costringe a considerare come dei *pari* i nodi della rete. Quando si parla di *P2P* però, non ci si limita ad immaginare una sistema di nodi medesimi, ma si spazia anche verso tipi di networking più particolari come quelli non *flat*, che appunto sono implementabili attraverso la tecnologia mobile *Ad-hoc*.

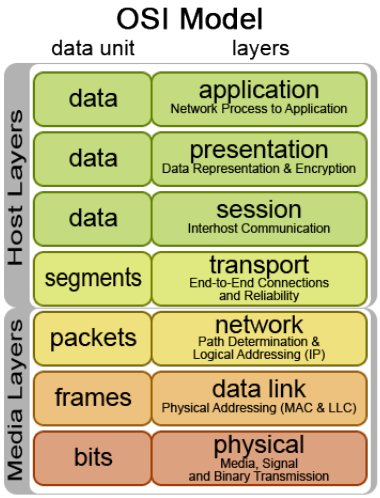
2.4 Lo Stack Protocollare

In questa sezione andremo ad ispezionare il funzionamento dello stack protocollare nelle reti mobili *Ad-hoc*, cercando di imprimere importanza alla divisione dei layers e alle problematiche che l'architettura esposta andrà a presentare. Ciò ci permetterà di comprendere al meglio il sistema in analisi, potremo poi adden-

trarci nello studio della parte comunicativa a livello algoritmico-protocollare, che è ciò che consente materialmente lo scambio di informazioni in questi complessi organismi.

2.4.1 I Layer

Come avremo ormai capito, in questo elaborato, si sta parlando di metodi per la diffusione di dati tra sistemi informatici. Naturalmente se proferiamo la parola "diffusione" in connubio alla parola "informatica", o a qualsiasi altra che esuli dalla commutazione di circuito, subito deve venire in mente lo standard *Open Systems Interconnection* (il modello ISO/OSI in figura 2.4.1)



© Wikipedia

Figura 2.4.1: Il Modello OSI

Come possiamo vedere dall'immagine, nel nostro caso, riteniamo molto importante suddividere il modello in due gruppi di layers, rispettivamente: *media layers* e *host layers*. La motivazione è semplice, dei due insiemi solo uno è degno di interesse per la nostra causa, ovvero il primo. Livelli come quelli *application*, *presentation* o *session*, come si può pensare, non servono alla descrizione tec-

nica di una *MANet*, al più ad attirare la nostra attenzione potrebbe essere il layer numero 4. Questo perché potremmo avere una rendita in termini di rete utilizzando al meglio i protocolli di trasporto che possono essere forniti, come i classici TCP (*Transmission Control Protocol*) e UDP (*User Datagram Protocol*) o i più "mirati" RTMP (*Real Time Messaging Protocol*) e SMB (*Server Message Block*), questi ultimi due rispettivamente utilizzati per: streaming real time e condivisione file/periferiche.

Come però dicevamo poche righe sopra, a rapire il nostro interesse è il *media layer*, potremmo essere tentati di focalizzarci soprattutto allo smistamento di pacchetti, quindi al livello *network*, ma questo sarebbe un errore. Se è vero che è possibile vedere il livello fisico come una *black box* che fa il suo "sporco lavoro" (tecniche di modulazione e trasmissione come *FHSS*, *DSSS*, *OFDM* e comunque l'utilizzo diretto di antenne) e fornisce informazioni ai layer superiori, non è altrettanto possibile avere una visuale di tipo "compartimento stagno" per quel che riguarda il layer *data link*.

"*Routing in* MANet's *must consider both Layer 3 and Layer 2 information, while traditional protocols rely on Layer 3 information only*"[16]; come sottolinea la frase citata, nei protocolli tradizionali si considera, per un corretto funzionamento di diffusione dei dati, solo il terzo livello. Questa affermazione appare più che sensata infatti, chiedersi come potrebbero essere utili in fase di routing informazioni residenti all'interno del livello *Mac,* quali dati derivanti dall'applicazione di tecniche come la *CSMA/CD* (*Carrier Sense Multiple Access with Collision Detection*) o la *RTS/CTS* (*Request to send / Clear to send*), ha una risposta relativamente semplice. La natura delle *Mobile Ad-hoc Networks*, per come esse sono state descritte, ci porta a dover controllare in maniera più approfondita alcune informazioni sull'attuale situazione di connettività all'interno delle rete (problemi di interferenza e raggiungibilità) per poter rispondere

alle *challenges* prima esposte. Aiuti informativi, in questo caso, posso venire solo ed esclusivamente da chi si occupa della gestione del flusso in termini di controllo degli errori e conferme di ricezione. Ad avere l'onere di portare a termine questi compiti è appunto il *data link*, che quindi verrà interrogato spesso nell'applicazione dei protocolli di *routing* per le reti mobili decentralizzate.

Finalmente arriviamo a parlare del nucleo, o cuore che dir si voglia, delle *Mobile Ad-hoc Networks*. Questo è costituito dal complesso insieme di operazioni che vengono svolte al terzo layer, corrispondente al *Network*, nell'*OSI Model*. Come tutti sappiamo il compito principe di questo livello è l'applicazione del protocollo di *routing* definito per il sistema in oggetto, per fare ciò questo dovrà:

- Conoscere la topologia di rete;
- Gestire dispositivi non omogenei;
- Considerare le debolezze in ambito wireless;
- Considerare i fattori critici dei dispositivi (grado di computazione e livello di batteria);
- Scegliere il *path* migliore per il viaggio di un pacchetto.

Oltre a questo, come detto in precedenza, il layer *Network* dovrà essere abile nel riadattare la rete cercando di recuperare informazioni direttamente dal livello sottostante, in maniera tale da creare un protocollo in grado di fornire un servizio di scambio più ottimizzato possibile.

2.4.2 La Sicurezza

Fattore classificato non critico a livello di *features* visto che, a parte in rari casi, non è in grado di intaccare tutta la rete, è la sicurezza che la *Mobile Ad-hoc Network* dovrebbe essere in grado di offrire. Vale comunque la pena spendere due parole per descrivere questo aspetto, quanto basta per riuscire a rendersi conto di chi può muovere un attacco e come riesca nell'impresa; fissare bene

quindi quali sono le *weakeness* è un buon modo per poter tenere la situazione sotto controllo.

In prima battuta possiamo dire che le *MANet's*, data la loro essenza, sono soggette a tutti e due i principali tipi di attacco, cioè: il *passivo* e l'*attivo*[20]. Rispettivamente intendiamo:

1. Un tipo di attacco non finalizzato all'intrusione o alla corruzione dei dati in un sistema. Si utilizzano attacchi di questo genere quando si vogliono "sniffare" informazioni che successivamente potranno essere utilizzate per alcuni scopi;
2. Un attacco che mina l'integrità dei dati oppure la stessa rete. Gli utenti malevoli, che tentano questo tipo di attacco, sono interessati solitamente a far collassare il sistema: in maniera diretta con la loro entrata o, per via indiretta, inducendo i terminali, camuffandosi da nodi *trusted*, a comportarsi in maniera avversa alla vita dell'ecosistema.

In secondo luogo possiamo soffermarci ad indicare quali tipi specifici di attacchi possiamo trovare nei vari livelli, in maniera tale da comprendere quali parti della comunicazione devono essere irrobustite. Ci imbatteremo:

- Nel *Transport* con:
  - **Session Hijacking** - Il classico dirottamento cercato per poter accede a zone senza avere l'autorizzazione necessaria;
  - **SYN Flooding** - In questa tipologia di attacco si fanno continue richieste ma non si inviano gli *acknowledgment* al server tanto da creare negazioni di servizio per utenti futuri.
- Nel *Network* con:
  - **Wormhole** - Si tratta di creare una rete che capti informazioni da un'altra attraverso un nodo che funge da collegamento;
  - **Location Disclosure** - Non è altro che la rivelazione e la divulgazio-

ne della posizione di alcuni nodi nella rete, è ovviamente un attacco utilizzato assieme ad altre tecniche;

  - **Tunneling** - Quando in una rete si utilizza un tunnel per una trasmissione criptata ma non decifrabile ai nodi di passaggio (utilizzata solitamente da nodi che sono entrati nella rete senza avere l'autorizzazione e vogliono scambiarsi informazioni non essendo l'uno nel range dell'altro);
  - **Impersonification** - Classico attacco in cui si cerca di intercettare informazioni fingendosi un certo nodo della rete.

- Nel *Data Link* con:
  - **Monitoring** - Classico attacco passivo con il quale si va, solitamente, a scegliere il nodo più papabile per scagliare un futuro attacco attivo;
  - **Disruption Mac** - Metodo con il quale si tenta di far cessare una determinata connessione di un certo nodo, è logicamente usata se l'utente malevolo si accorge che nella rete ci sono terminali con importanza maggiore.
- Nel *Physic* con:
  - **Jamming** - Questo è l'attacco più banale che si possa avere, si tratta in semplice maniera di disturbare la comunicazione andando ad operare direttamente sulle onde radio;
  - **Eavesdropping** - Una sorta di *sniffing*, un'intercettazione che viene fatta però direttamente a livello fisico.

Come notiamo, gli attacchi peggiori risiedono proprio nel livello *Network*, cioè il layer che ha il compito di gestire il *routing*. Questa situazione non è un caso, in precedenza abbiamo specificato come il terzo layer sia il cardine delle *MANet's* ed è ovvio aspettarsi che i pericoli peggiori arrivino proprio da questa zona.

È bene quindi riuscire a descrivere una bozza di comportamenti che queste

reti dovrebbero avere per innalzare il proprio livello di affidabilità, essi sono [6]:

- I segnali di tipi *path* non devono poter essere falsificati;
- Un nodo deve sapere l'origine di un messaggio di *routing*, non può limitarsi ad accettarlo solo perché inviato da un nodo *trust*;
- Anche se un *path* cambia non devo aggiornarlo se il pacchetto non è più nelle mani del mittente. Se necessario distruggerò il pacchetto e lo rinvierò;
- Non devo implementare algoritmi che scelgano sempre e solo il *path* più corto, significa esporsi agli attacchi in facile maniera;
- I nodi non autorizzati possono rimanere nella rete utilizzando apposite funzioni, mai però devono entrare a far parte delle funzioni di *routing* o *discovery*.

Si noti comunque che il terzo livello non gestisce in maniera diretta la sicurezza. Per raggiungere quest'ultima è quindi indispensabile creare algoritmi protocollari che si incentrino nella collaborazione tra i nodi esistenti in un determinato momento (nel quale possono certificarsi come terminali affidabili tra loro).

2.5 L'Instradamento

Instradare qualcosa, un pacchetto nel nostro caso, significa riuscire a far recapitare l'oggetto al destinatario desiderato, cercando ovviamente di impiegare il minor tempo possibile; in questa sezione andremo ad analizzare le tecniche di routing sfruttate dalle *Mobile Ad-hoc Networks* nelle operazioni di *delivery*.

Per addentrarci al meglio in questo discorso è utile differenziare il routing classico con quello utilizzato nelle *MANet's*. Le difformità sostanziali non arrivano neanche alla decina, ma sono talmente profonde da far sì che i tradizionali protocolli per reti *wired* non possano essere utilizzati su reti *Ad-hoc*. Sicuramente la prima discrepanza che possiamo intuire è la classificazione di instradatore; nelle classiche reti, a fungere da guida per i dati, c'è un dispositivo detto router,

nelle *MANet's* invece, a compiere questa missione, possono essere potenzialmente tutti i nodi connessi; infatti ogni terminale ha facoltà di rimbalzare pacchetti ad un suo pari in maniera diretta, senza dover utilizzare un apparecchio-tramite che svolga le operazioni legate alla gestione degli indirizzi. Questa architettura porta alla seconda differenza che ritroviamo nei sistemi che stiamo trattando, cioè la forte ridondanza. Avere una rete nella quale una macchina, alla cui viene garantito l'accesso, può collegarsi a tutti i nodi entro il suo range di dialogo si traduce in avere una miriade di collegamenti futili, un esempio classico è la presenza di molti terminali ad un solo *hop* e successivi dispositivi a due *hops* (questi ultimi saranno raggiunti in tanti modi quanto è il *numero di dispositivi ad un hop* dal nuovo apparecchio aggiunto).

Altra differenza basilare da tener conto è l'*interfaccia singola*. Solitamente, nel reticolato ad infrastruttura, i terminali che effettuano l'instradamento hanno tante interfacce quante sono le sottoreti che devono servire, tutto ciò per curare al meglio "l'indicizzazione" dei facenti parti del sistema. Nelle *Mobile Ad-hoc Networks* però non esistono le *stub net*[2], tutto è visto nello stesso piano. Neanche in questo caso sarà possibile applicare protocolli esistenti, essi sarebbero uno spreco di computazione.

Continuando ad elencare le disuguaglianze ci imbattiamo nei connotati non più statici. Esistono varie caratteristiche che ritroviamo in entrambi i tipi di rete, la differenza qui sta nel fatto che per ciò che riguarda il mondo *wired* i *boundary limit* sono sempre gli stessi e tendono a rimanere fissi durante l'attuazione del protocollo; nelle *wireless Ad-hoc* invece i limiti non sono noti prima che l'ecosistema si sia formato e comunque vanno ad evolversi in maniera continua, cercando di supportare il movimento attuale. Da notare è anche l'interessamento ad alcune caratteristiche non utili al mondo dei terminali fissi (un esempio è il livello di energia rimasta, quindi il problema dell'esaurimento della batteria).

[2]Stub Network - Una particolare sottorete con un unico NAT.

Ultimo motivo per il quale è fortemente sconsigliato l'utilizzo di protocolli già esistenti per supportare le reti in oggetto è la sicurezza, qui inutile dilungarsi; come abbiamo detto in precedenza questo problema deve essere affrontato con uno spirito completamente diverso da quello tradizionale, tutto ciò per non incorrere in problematiche quali *intercettazioni*, attacchi attivi di tipo *man-in-the-middle* o affini.

Detto questo, un *algoritmo di routing* dovrà[23]:

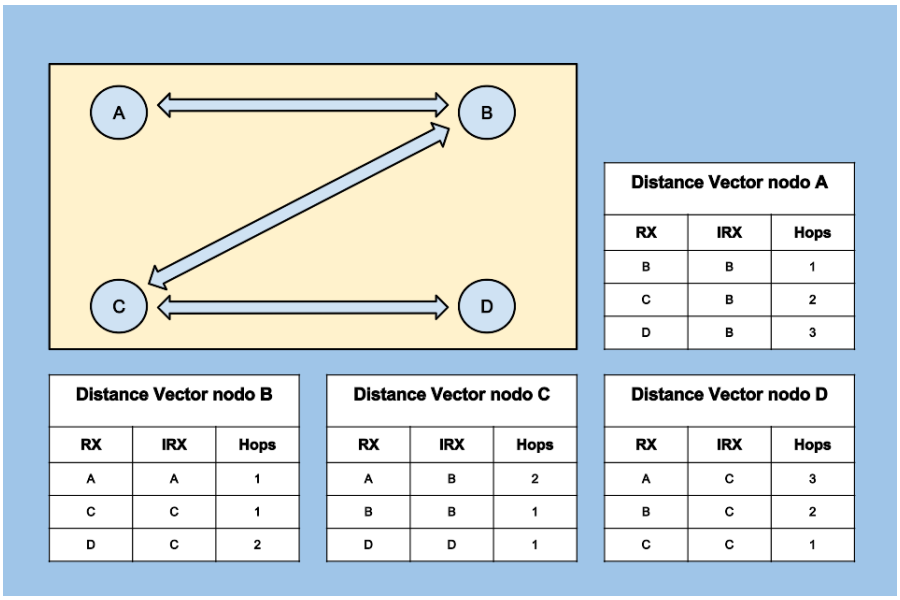
- Creare *tabelle di routing* descriventi la topologia;
- Riuscire ad aggiornare in maniera efficiente (tempo e quantità di informazioni) le *tabelle di routing*;
- Scegliere il *path* migliore date le tabelle salvate;
- Ottimizzarsi, a livello topologico, in breve tempo;
- Utilizzare *multipath* per salvaguardarsi da situazioni di indisponibilità-nodi su specifici percorsi.

2.5.1 Conoscere la Topologia

Prima di affrontare la parte riguardante la suddivisione dei protocolli, è necessario introdurre le due tipologie principali sul quale un network può basarsi per inviare e ricevere pacchetti tra i terminali che lo compongono. Distingueremo quindi algoritmi di tipo *Distance Vector* e altri di tipo *Link State*.

Distance Vector

Come dice la categoria, questi algoritmi si basano sui vettori di distanza (esempio in figura 2.5.1). Questi non sono altro che particolari strutture che vanno ad indicare informazioni solo ed esclusivamente riguardanti i nodi tra di essi adiacenti (come si può intuire, queste strutture saranno residenti nelle *routing table*).



Distance Vector nodo A

RX	IRX	Hops
B	B	1
C	B	2
D	B	3

Distance Vector nodo B

RX	IRX	Hops
A	A	1
C	C	1
D	C	2

Distance Vector nodo C

RX	IRX	Hops
A	B	2
B	B	1
D	D	1

Distance Vector nodo D

RX	IRX	Hops
A	C	3
B	C	2
C	C	1

Figura 2.5.1: Distance Vectors di una generica MANet

Con questo *routing*, le istanze di tabella saranno composte da tre informazioni principali: la **destinazione**, il **passo successivo** e il **numero di *hops***. Le funzioni del primo e del terzo dato sono abbastanza intuitive, il secondo invece serve ad evidenziare il primo passo che l'eventuale pacchetto farà verso la sua *destination*, l'intera colonna **IRX** quindi indica tutti i vicini utilizzabili, in fase di instradamento, da quel terminale. Si noti comunque che nel mondo reale le tabelle devono essere integrate con i tempi di *delivery* di pacchetto tra nodo e nodo (per semplicità, nelle immagini, stiamo considerando un tempo identico in qualsiasi *hop*).

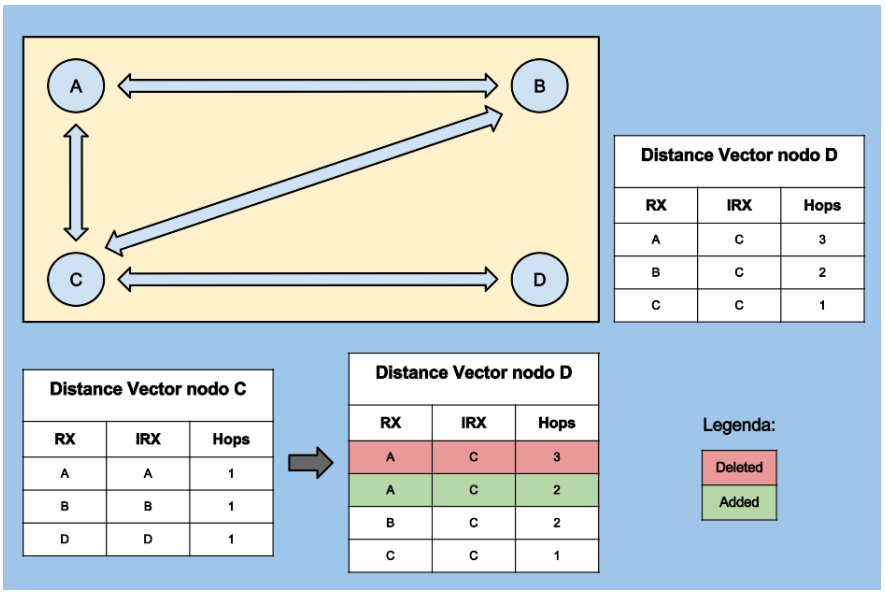


Figura 2.5.2: Applicazione del Routing di Bellman-Ford

Materialmente gli algoritmi basati su vettori di distanza funzionano con *Bellman-Ford*. "Ragionano" quindi su grafi pesati cercando di estrapolare il cammino minimo tra due nodi. Possiamo apprezzarne il funzionamento in figura 2.5.2; ammettiamo che la situazione di partenza sia quella della figura 2.5.1, allora nella tabella di **D** sarà indicato un *path* lungo 3 per arrivare ad **A**. A questo punto però si crea un collegamento tra **A** e **C** ed entrambi aggiornano immediatamente il loro *distance vector*, successivamente poi **C** spedirà il suo verso **D**, adesso quest'ultimo aggiornerà la sua *table* e ci sarà quindi un'ottimizzazione della rete. Dobbiamo però evidenziare una problematica di questo *modus operandi*, ovvero il **count to infinity**. Questo (figura 2.5.3) è un problema che affligge tutti i protocolli che ammettono cicli nei grafi. Consiste nell'avere cammini che incrementano il loro numero di *hops* a slot temporali fissi quando un *path* reale, tra i due nodi in esame, non esiste.

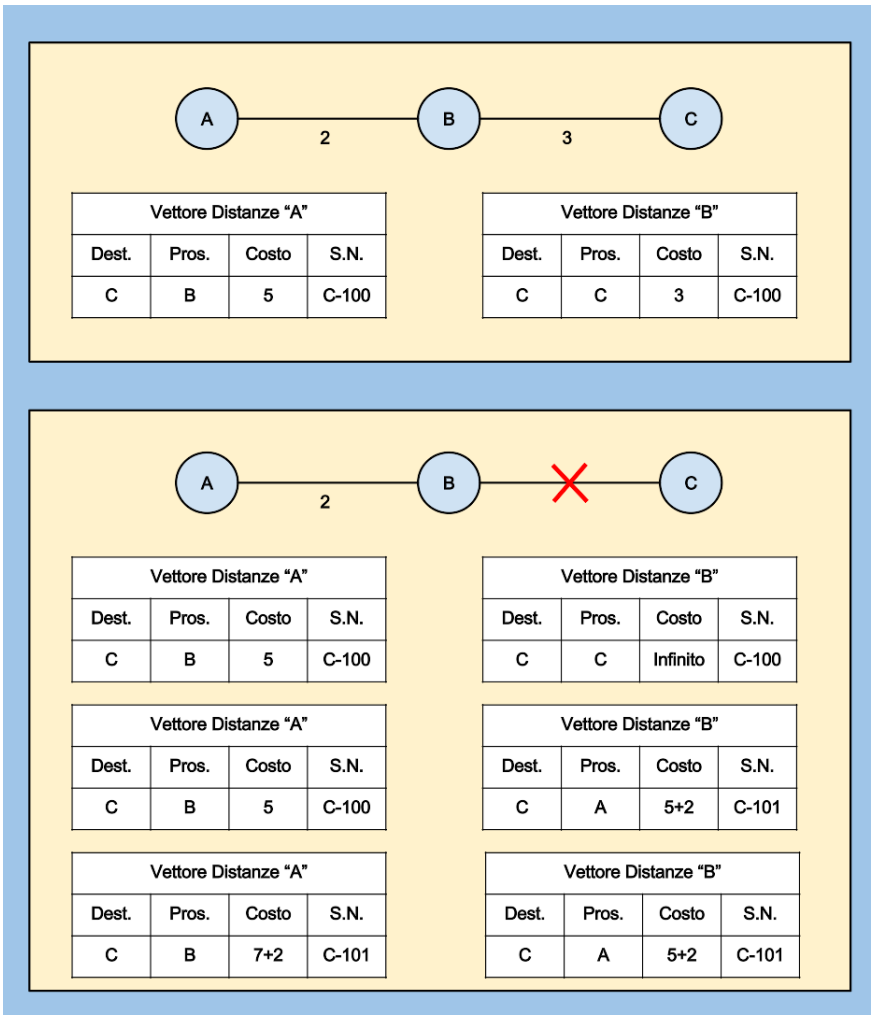


Figura 2.5.3: Count To Infinity in DSDV

Si pensi alla classica situazione nella quale un nodo si allontana da un congrega di simili, il link verso di lui non esisterà più; un suo vicino quindi non potrà più raggiungerlo, ma il terminale adiacente a quest'ultimo avrà ancora in tabella l'informazione di percorso a due *hops* di distanza. Nel caso questi non venisse avvertito in tempo, potrebbe inviare il suo vettore a chi rimaneva ad un solo *hop* dallo scomparso. Se così fosse, tra questi due nodi, inizierebbe uno

scambio reciproco di informazioni sul raggiungimento del device non facente più parte della zona rete; ciò non farebbe altro che portare ad innalzare il valore di *costo* nelle rispettive tabelle senza però avere una meta percorribile.

Risolvere questo problema è possibile, basta improntare un meccanismo che invii l'informazione di link-cassato, in maniera istantanea, al suo vicino. Una sorta di notifica *push* anziché una richiesta di tipo *pull*.

Link State

Altro metodo che permette di sapere quale è l'attuale topologia di rete è il *Link State*. Questo algoritmo cerca, al contrario del *Distance Vector*, di comunicare la struttura della rete a tutti i nodi che ne fanno parte, così da avere un sistema che, a regime, calcoli i suoi cammini minimi in un tempo irrisorio e possa gestire un elevato numero di terminali. Possiamo vedere un esempio in figura 2.5.4.

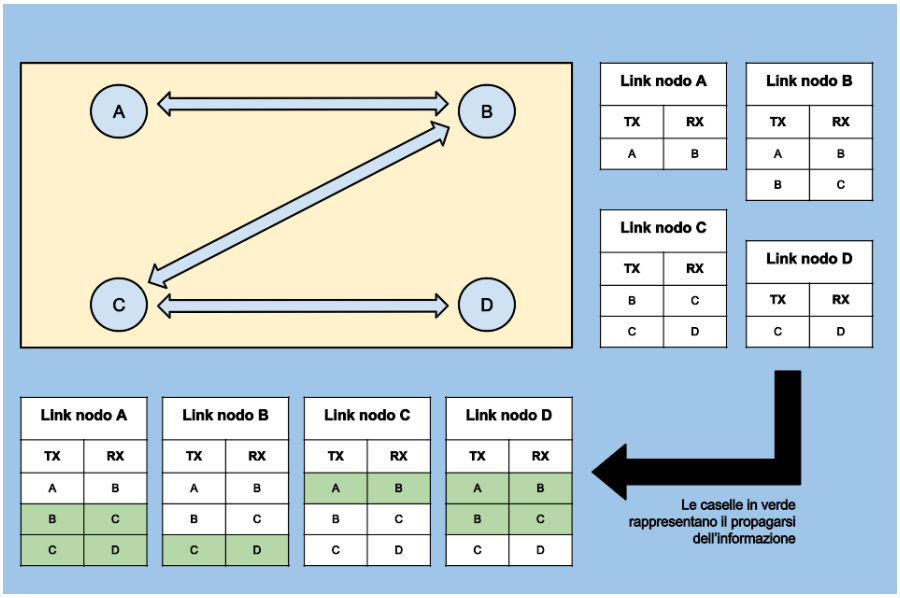


Figura 2.5.4: Esempio di stato dei link in una MANet

Come notiamo, inizialmente, i nodi conoscono solo ed esclusivamente i loro

link immediati; al momento di *join* infatti, ogni terminale riempie il suo *database* con i collegamenti diretti che si vanno a creare. Successivamente, con il passare del tempo, l'algoritmo fa sì che ogni nodo dell'organo conosca tutti i link presenti in rete, questo rende possibile tutta una serie di calcoli su *path* fatta a priori.

A garantire che il tutto si "regga in piedi" ci pensa il campo *Sequence Number* inserito nel *Packet Link State* (pacchetto contenente informazioni sui link). Questo numero consente di confrontare se il pacchetto ricevuto è più o meno recente di quello che in precedenza ha aggiornato il *database*, con questo meccanismo siamo sicuri che i dati relativi ai collegamenti siano sempre aggiornati e possano essere utilizzati per la comunicazione.

Un classico esempio di scambio di *Packet Link State* per aggiornamento di *DB* è riportato nella figura 2.5.5,

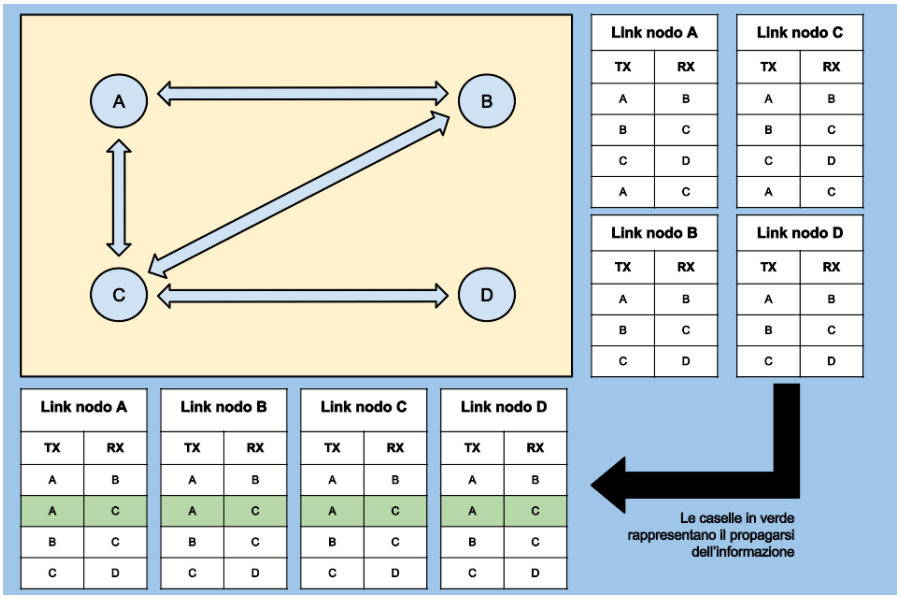


Figura 2.5.5: Propagazione informazione in Link State con un nuovo link

Qui non abbiamo un aggiornamento che salverà il *path* migliore, tutti i percorsi continuano ad essere tenuti in memoria; si favorisce così il *multipath* a di-

scapito del risparmio di computazione (si ha un conteggio dinamico e continuo dei cammini).

2.5.2 Inviare i Dati

"*Routing is the mechanism of forwarding packet towards its destination using most efficient path. Efficiency of the path is measured in various metrics like, Number of hops, traffic, security, etc.*" [20]. Questa frase raccoglie in pochissime parole quale sia il compito dei *router*, funzione che nelle *MANet's* può essere svolta potenzialmente da qualsiasi nodo; come possiamo leggere, siamo davanti a moltissimi criteri da dover valutare per determinare il miglior *path*. Naturalmente non potremo mai massimizzarli tutti per arrivare ad una soluzione che sia ottimale in una qualsiasi configurazione ottenuta.

Molti ricercatori, università o aziende, nel corso degli anni, hanno proposto il loro miglior modello protocollare di instradamento, ma ancora oggi il dibattito risulta aperto; questo perché non esiste, e forse mai esisterà, una distribuzione di "pesi" oggettiva per tutti i criteri da considerare, si pensi addirittura che possono esistere utilizzi che non considerano minimamente criteri definiti fondamentali per altri.

La figura 2.5.6 ci fornisce la divisione dei protocolli per categorie, in questo caso in numero pari a cinque (rettangoli gialli). In realtà possiamo comunque unire queste regole nei due primi gruppi e cioè quello dei *proattivi* e quello dei *reattivi*. La constatazione è giustificata dal fatto che qualsiasi protocollo residente nei *gerarchici* o nei *position aware* sarà comunque proattivo o reattivo, fanno però eccezione gli *ibridi* che costituiscono una mescolanza tra le due tipologie principe ma comunque presentano una predominanza dell'una o dell'altra tecnica.

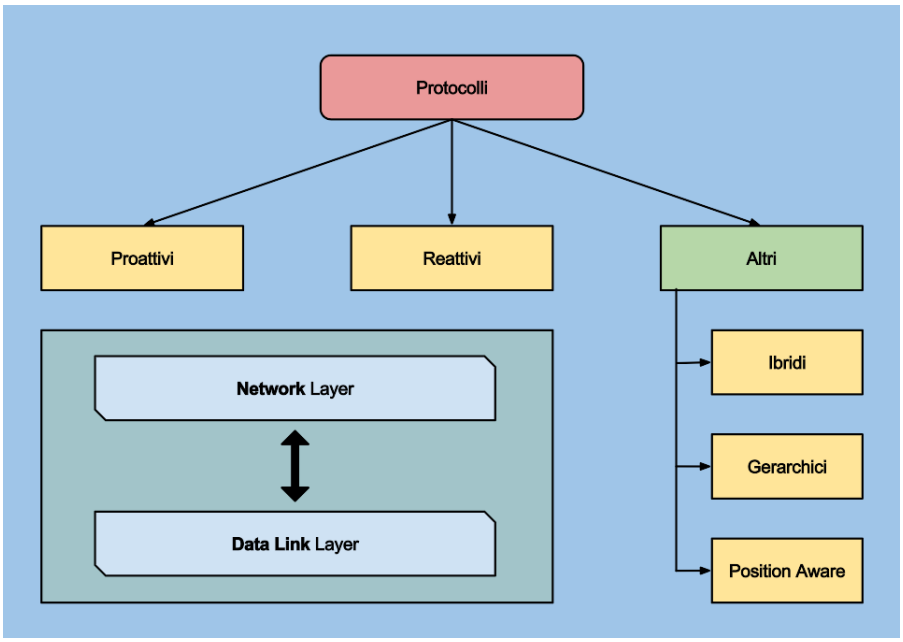


Figura 2.5.6: Classificazione di Protocolli in MANet

Detto questo, cerchiamo appunto di comprendere i due protocolli base delle *Mobile Ad-hoc Networks*. Le tecniche proattive sono anche conosciute con l'appellativo di *Table Driven*, questo nome è giustificato dal comportamento dei terminali, esso è infatti teso a mantenere in memoria la topologia di rete anche quando non è stata avanzata alcun tipo di richiesta. Praticamente ogni *router* cerca di inviare le proprie informazioni ai vicini e allo stesso tempo tenta di riceverne da questi, in modo da "sapere" a priori tutti i *path* che potranno essere richiesti. Le tecniche reattive invece sono dei servizi *On Demand*, i terminali fanno sì store di informazione di *path,* ma mai prima che questi ultimi vengano richiesti per una *delivery*.

La descrizione data fa subito immaginare quali siano i limiti attribuibili alle due categorie. I *proactive* avranno bisogno di fare computazione continua ed avere capacità di memoria elevata, allo stesso tempo la loro latenza, quando

i nodi si saranno oramai "conosciuti" reciprocamente, sarà quasi inesistente. I *reactive* saranno invece caratterizzati da una lentezza in invio, lentezza data dalla funzionalità di *discovery* e non dall'occupazione di risorse. Ne conseguirà, considerando anche l'*overhead* (si veda tabella 2.1), che i primi protocolli saranno preferiti in reti molto dinamiche e magari nelle quali si tende a sfavorire il *multipath* (data il formarsi di cammini ciclici); i secondi invece saranno utilizzati in reti che tendono a non mutare velocemente, semplificano quindi la vita di tecniche che "studiano" le situazioni in cui sono immerse per arrivare alla soluzione ottimale.

	Approccio Proattivo	Approccio Reattivo
Latenza	**BASSO**	**ALTO**
Overhead	**ALTO**	**BASSO**

Tabella 2.1: Comparazione delle categorie principali di Protocolli MANet

Da simulazioni effettuate[17] è stato possibile constatare che solitamente i reattivi sono migliori dei proattivi, questo sia perché la loro natura permette una migliore ottimizzazione dei parametri, data l'abbondante potenza di calcolo (e memoria) non utilizzata, sia perché i *pattern di movimento* usati, che cercano di rispecchiare la realtà, non sono caratterizzati da profonde modifiche alla topologia.

Per quel che riguarda le due categorie rimanenti accenniamo al fatto che i *gerarchici* tenderanno ad avere un'architettura a supernodi, mentre i *position aware* si baseranno in forte maniera su posizione e movimenti attorno a terminali interessati.

Riportiamo in tabella 2.2 alcuni protocolli divisi per le varie tipologie.

Proattivi	**Reattivi**	**Ibridi**
OLSR (Optimized Link State Routing)	AODV (Ad-hoc On demand Distance Vector)	ZRP (Zone Routing Protocol)
DSDV (Destination Sequenced Distance Vector)	DSR (Dynamic Source Routing)	OORP (Order One Routing Protocol)
WRP (Wireless Routing Protocol)	ABR (Associativity Base Routing)	
STAR (Source-Tree Adaptive Routing)	TORA (Temporary Ordered Routing Algorithm)	
	PA (Power Aware)	
	SSR (Self Selective Routing)	
	CEDAR (Core Extraction Distributed Ad-hoc Routing)	
	SSA (Signal Stability Adaptive)	
	DBR (Distance Based Routing)	

Gerarchici	**Position Aware**
FSR (Fisheye State Protocol)	GPSR (Greedy Perimeter Stateless Routing)
HSR (Hierarchical Segment Routing)	LAR (Location Aided Routing)
CGSR (Clusterhead Gateway Switch Routing)	GRA (Geographical Routing Algorithm)

Tabella 2.2: Principali Protocolli di Routing per MANet's

Capitolo 3

Modellare Problemi a più Partecipanti

In ogni momento della vita siamo chiamati ad effettuare delle scelte che si ripercuotono nel nostro futuro. Il modo solito, con il quale cerchiamo di approcciare i nostri problemi, è ovviamente quello con il quale abbiamo la possibilità di trarre il miglior vantaggio possibile, sia questo riferito ad una somma di denaro, ad una prossima posizione lavorativa o anche, perché no, ad una decisione di vita che scaturisce dalle nostre emozioni. Più in generale vogliamo intendere che la nostra indole ci porta, sempre e comunque, a cercare di massimizzare il profitto vagliando l'assortimento di alternative che abbiamo davanti. Questo *modus operandi* non incontra nessun ostacolo, nella sua messa in atto, fino a quando a "pescare" dal recipiente delle decisioni siamo solo noi stessi; iniziamo però ad avvertire problemi se a partecipare alla costruzione del futuro si inseriscono altri personaggi con le loro rispettive scelte, ovviamente quando queste ultime sono in grado di interferire con la nostra attesa.

Per fare un esempio banale, se vivessimo in un paese organizzato in città e

per i sindaci ci fosse la possibilità di ottenere dallo stato un finanziamento a scelta tra due opzioni:

opzione-uno: 200,00 € per ogni cittadino con regolare residenza;

opzione-due: 100,00 € per ogni m^2 di estensione del territorio.

Inverosimilmente, nel caso esistesse una sola città, il primo cittadino di questa non esiterebbe ad effettuare la scelta basandosi esclusivamente su una semplice moltiplicazione, indicando come strada da perseguire quella che porta ad un introito superiore per le casse del suo comune. Nella realtà però esistono una moltitudine di città in uno stato ed ovviamente questo chiama le prime ad accordarsi verso una scelta identica (tutte le città dovranno sottostare o alla **opzione-uno** o alla **opzione-due**, in altre parole alcuni sindaci dovranno accontentarsi della selezione che non massimizza il valore monetario in entrata).

Un semplice esempio è riportato in figura 3.0.1. Come notiamo abbiamo due città ("A" e "B"), la prima che si estende per 10.000 m^2con un numero di residenti pari ancora a 10.000, mentre la seconda ha una superficie di 20.000 m^2 in cui abitano solo 5.000 anime.

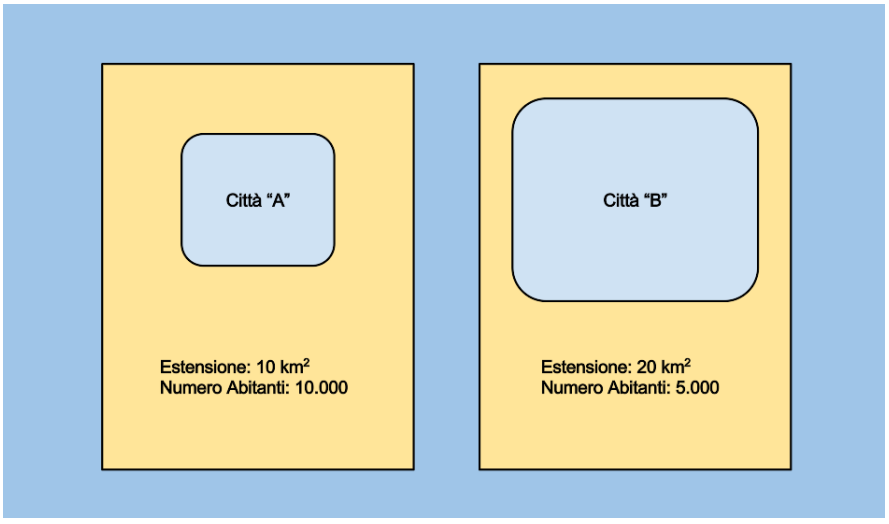


Figura 3.0.1: Città a confronto per criteri

I conti sono estremamente semplici da effettuare ed i risultati ottenibili possono essere:

- Scelta della **opzione-uno**:
  - Introito Città "A": 2000000;
  - Introito Città "B": 1000000.
- Scelta della **opzione-due**:
  - Introito Città "A": 1000000;
  - Introito Città "B": 2000000.

Come è possibile immaginare, i sindaci delle Città "A" e "B", tenderanno a scegliere rispettivamente il primo l'**opzione-uno** e il secondo l'**opzione-due**; in caso non saltasse fuori un accordo in grado di far convergere le opinioni, nessuna delle due urbe incasserebbe nulla e si decreterebbe quindi la soluzione pessima al problema.

È proprio in casi come questo, quindi in situazioni dove la formulazione di un sistema puramente matematico risulta insufficiente per delineare una soluzione, che ci accorgiamo di dover trovare metodi alternativi per descrivere cosa andrà ad accadere nel sistema preso in atto. Ad accorrere in nostro aiuto è la disciplina matematica della quale *John von Neumann* è considerato il padre grazie alla pubblicazione dell'articolo "*Zur Theorie der Gesellschaftsspiele*" sulla celebre rivista tedesca *Mathematische Annalen* [15], stiamo ovviamente parlando della **Teoria dei Giochi**. Questa branca delle scienze si occupa di porgere strumenti che permettono di trovare quella che viene chiamata la *Solution Concept*, cioè una proposta di soluzione con la quale si è in grado di intuire l'atteggiamento che i vari partecipanti al quesito potranno avere.

3.1 I Giochi Strategici

Abbiamo appena introdotto uno scenario che ci ha permesso di creare nella nostra mente una raffigurazione di ciò che, da adesso in poi, andremo ad identificare come **gioco**. Cerchiamo però ora di delineare al meglio questo disegno fornendo una definizione non ambigua del termine appena utilizzato.

Possiamo quindi affermare che un **gioco** è: *"Una circostanza reale, modellabile attraverso le intenzioni dei giocatori in maniera tale da riuscire a descrivere tutti gli stati possibili che possono accadere."* Non andranno, pertanto, a demarcare una situazione di **gioco** tutti quegli eventi che non ammettono alcuna strategia (l'affidarsi solo al caso) oppure che vanificano la scelta di una di queste per inutilità (il caso del monopolio).

Per essere ancora più precisi, le **parti basilari** che vanno a comporre i **giochi** sono quattro, rispettivamente:

- **I *Giocatori*** - un insieme finito di persone che partecipano attivamente ad un problema effettuando delle scelte:
  $i \in N$ con $N = \{1, ..., n\}$.
- **Le *Regole*** - l'insieme di tutte le direttive che devono essere seguite per partecipare:
  - Chi effettua la decisione;
  - Quando si effettua la decisione;
  - Cosa cambia dopo aver effettuato una certa mossa;
  - Eccetera...
- **Le *Strategie*** - i vari insiemi (uno per ogni giocatore) contenenti tutte le possibili decisioni da parte di un soggetto attivo nel sistema:
  $S_i = \{s_1^i, ..., s_n^i\}$ considerando s_3^2 come indicazione della scelta numero 3 per il giocatore numero 2.
- **I *Payoff*** - funzioni di utilità che vengono applicate alle scelte dei giocatori

per verificare cosa essi traggono dalle decisioni:

$u : S \to \mathbb{R}$ considerando $u_i(s_i, s_{-i})$ come l'utilità del giocatore i che effettua una determinata scelta s_i mentre gli altri giocatori prendono decisioni s_{-i}.

Detto questo è possibile effettuare una prima distinzione tra le tipologie di gioco esistenti. Stiamo parlando della suddivisione che si può avere basandosi sulla possibilità di accordo tra le parti; distinguiamo infatti, secondo questo aspetto, i **giochi strategici** (o **giochi non cooperativi**) ed i **giochi cooperativi**. È necessario sottolineare che, con possibilità di accordo, intendiamo il perseguimento di un fine comune tramite la stipula di un alleanza che porterà a comportarsi come coalizione; in altre parole la funzione di *payoff* dipenderà da un profilo strategico appartenente ad un set che descrive l'associazione delle parti che decidono di muoversi assieme.

In questa sezione, come suggerisce il titolo, andremo a focalizzarci su problemi modellati in maniera tale da non ammettere alleanze tra le parti; successivamente, comunque, descriveremo come sarà possibile la formazione di questi gruppi e quindi il calcolo del profitto per gli appartenenti ad esso. Risulterà naturale, a quel punto, modellare un problema già affrontato, al quale però verrà aggiunta la possibilità di un accordo tra le componenti, appunto un **gioco cooperativo** (o anche detto **gioco coalizionale**).

3.1.1 La Rappresentazione

Nessun quesito, in particolare se di tipo matematico, può essere risolto in modo corretto e non ambiguo se non viene formulato in maniera rigorosa ed accurata. Più precisamente ogni problema deve essere **formalizzato**, letteralmente deve essere data ad esso una **forma** che permetta di descriverlo nei suoi

più minuziosi particolari. Nel nostro caso quindi, si ha bisogno di sistema che esponga il **gioco** sotto il punto di vista delle sue **parti basilari**; gli aspetti da tracciare perciò saranno: le **regole** che si dovranno seguire, le scelte possibili in base alle **strategie** attuabili e le "rendite" che queste faranno conseguire ai vari giocatori.

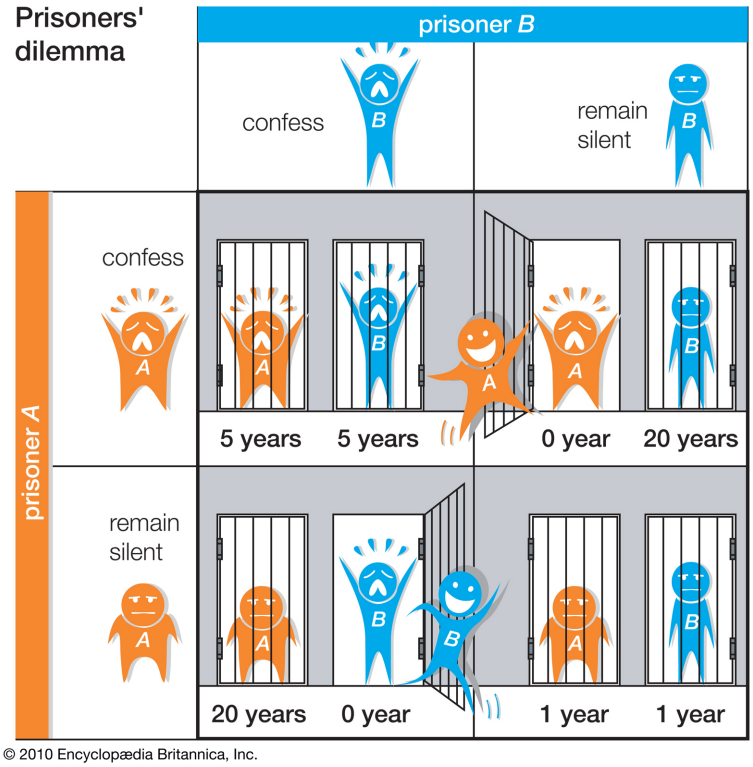


Figura 3.1.1: Dilemma del Prigioniero

Per capire meglio cosa significa formalizzare un quesito è necessario introdurne almeno uno e la scelta non può fare altro che ricadere su uno dei più celebri problemi proposti, nel corso degli anni, per illustrare questa branca della matematica; stiamo ovviamente parlando dell'esempio che va sotto il nome di *Dilemma del Prigioniero* (PD), esistente ormai da più di 60 anni, grazie alle menti di *Merrill Flood* e *Melvin Dresher* (ricercatori della *Research And Deve-*

lopment Corporation), ma perfezionato solo in fase successiva dal matematico canadese *Albert William Tucker [24].*

Questo problema, riportato in figura 3.1.1, consiste nella seguente e popolare vicenda: "Due criminali compiono un determinato reato. I servizi dell'ordine, investigando, riescono a risalire a loro ma non hanno la sicurezza della colpevolezza di entrambi. Decidono quindi di arrestare preventivamente i due individui, per poi interrogarli separatamente senza dar loro la possibilità di accordarsi". Proprio questa forzatura finale costituisce il **gioco**; qui infatti ad ognuno dei due uomini, che si da per scontato si dichiarino innocenti, viene data la possibilità di tradire il complice (senza sapere la scelta dell'altro) delineandogli il seguente scenario (allineato alla figura 3.1.1):

- Se tutti e due tradiscono allora entrambi vengono condannati per cinque anni;
- Se i due si coprono a vicenda verranno condannati ad un solo anno;
- Se le scelte dei malviventi sono in opposizione, chi tradisce è libero e chi rimane in silenzio sconterà venti anni di pena.

Come è semplice notare gli anni di galera non sono altro che i **payoff** dei nostri giocatori, in questo caso ovviamente sono da considerare con il segno negativo dinanzi (è meglio essere condannati ad un anno piuttosto che a venti, appunto $-1 > -20$). Detto questo, considerata anche l'intrinseca avversione al rischio dell'uomo, è possibile concludere che la *Solution Concept* di questa scena consisterà nella convergenza di ambedue le parti verso la scelta di tradire il compagno. Si ipotizza questo perché, non conoscendo la scelta dell'avversario, la vera domanda che viene fatta ad ogni partecipante si traduce in: "Preferisci rischiare di ottenere cinque anni di pena, con la possibilità di essere libero, oppure rischiare venti anni con la possibilità di averne uno?"; la risposta a questa domanda appare scontata (questo perché lo è), di certo però non andrà a configurare la

migliore situazione, cioè quella denotata dalla strategia che vede ambo i membri coprirsi a vicenda.

Ma come rappresentare questo fiume di parole in linguaggio schematizzato? Esistono due tipologie di forme che possono essere utilizzate per simboleggiare **giochi strategici**, rispettivamente troviamo: la **forma normale** e la **forma estesa**.

3.1.1.1 La Forma Normale

Come può suggerirci il nome, questa tipologia di descrizione degli scenari è sicuramente la più utilizzata. Inconsciamente ricorriamo ad essa in ogni giorno della nostra esistenza se siamo chiamati ad effettuare scelte in condizioni non monopolistiche. La **forma normale** consiste nell'avere una situazione dettagliata sulle conseguenze dettate dalla nostra preferenza; in altre parole, in qualsiasi momento ci è possibile verificare la mutazione del nostro **payoff** al variare delle strategie altrui.

Matematicamente in un gioco ad n giocatori verranno definiti:

- n insiemi di scelte $\theta_1, \theta_2, \theta_3, ..., \theta_n$;
- n funzioni che attribuiscono un **payoff** $f_1, f_2, f_3, ..., f_n$. Ogni funzione è definita in maniera tale da prendere in input tutte le possibili combinazioni di scelte e fornire in output un valore numerico, quindi:

  $f_i : \prod_{0<l<n} \theta_l \to \mathbb{R}$ con $\prod$ prodotto cartesiano.

Questo modo di operare permetterà quindi di conoscere il beneficio (o il danno) tratto da ogni giocatore in tutte le possibili situazioni che andranno a verificarsi.

Per capire meglio cosa si intende, possiamo esporre il *Dilemma del Prigioniero* appunto attraverso la **forma normale**:

- $N = \{A, B\} \quad \Rightarrow \quad \mid N \mid = 2 \quad \Rightarrow \quad n = 2$;
- Scelte:

$\theta_1 = \{T, NT\}$ con T tradisce e NT non tradisce;

$\theta_2 = \{T, NT\}$ con T tradisce e NT non tradisce.

- Funzioni:

$f_A : (T,T) \to -5 \bigvee (T,NT) \to 0 \bigvee (NT,T) \to -20 \bigvee (NT,NT) \to -1$;

$f_B : (T,T) \to -5 \bigvee (T,NT) \to -20 \bigvee (NT,T) \to 0 \bigvee (NT,NT) \to -1$.

Come notiamo possiamo tenere sotto controllo l'evolvere del gioco in maniera molto semplice, ad esempio: "Cosa può succedere se il giocatore A tradisce? Rischia 5 anni oppure è libero" si traduce in $f_A : (T, *) \to -5 \bigvee 0$. Da questo punto è quindi molto semplice riportare i risultati in modo tale che siano leggibili al meglio (tabella 3.1).

		B	
		T	NT
A	T	(-5, -5)	(0, -20)
	NT	(-20, 0)	(-1, -1)

Tabella 3.1: Dilemma del Prigioniero in Forma Normale

Per rendere ancora più semplice la fruizione dei dati si può scegliere di scrivere con due colori i vari **payoff**, in questo caso sono stati usati il blu ed il rosso per indicare rispettivamente il giocatore A ed il giocatore B.

3.1.1.2 La Forma Estesa

Questa seconda tipologia di rappresentazione risulta essere meno pratica della prima, ne consegue infatti un minore utilizzo; tuttavia non è solo un modo diverso per descrivere un gioco, può risultare molto utile far ricorso ad essa perché è in grado di fornire elevata potenza espressiva in alcune occasioni, un esempio possono essere i giochi a decisioni non simultanee. In questi ultimi i giocatori "muovono" in maniera alternata, ovviamente la conoscenza da parte del secondo giocatore della scelta effettuata dal primo andrà a condizionare la deci-

sione che il secondo selezionerà. In questo caso, appunto, è possibile etichettare le mosse migliori utilizzando la **forma estesa**; essa consiste nella costruzione di un albero in grado di tenere traccia di tutti i possibili sviluppi (dall'inizio alla fine del gioco). Immaginiamo una partita di scacchi, formalizzarla secondo la **estesa** significherebbe creare un gigantesco albero; nel caso riuscissimo nell'impresa potremmo calcolare l'andamento della percentuale di vittoria ad ogni passo e scegliere quindi la miglior mossa.

Ricorriamo ancora al *Dilemma del Prigioniero* per rendere l'idea della **forma estesa** (figura 3.1.2).

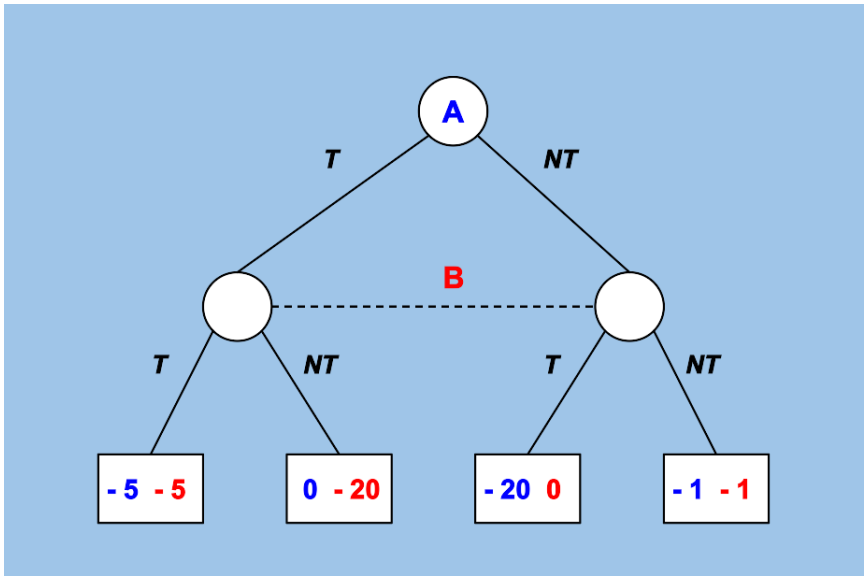


Figura 3.1.2: Dilemma del Prigioniero in Forma Estesa

Ammettendo che in questo caso B muova dopo di A, è chiaro come il primo giocatore citato, muovendo per secondo, faccia ricadere la sua scelta sul tradire visto che in ambo i casi massimizzerà il suo beneficio (considerando che non abbia interesse ad aiutare l'altro).

3.1.2 I Giochi a Somma Costante

Abbiamo parlato di funzioni di payoff rappresentandole come delle regole che permettono il calcolo del provento, attribuibile ad un partecipante, partendo dallo scenario (strategie scelte) in cui si è ricaduti. Non è difficile immaginare, arrivati a questo punto, la possibilità di creare giochi che distribuiscano i profitti dei giocatori in maniera particolare. Questo è il caso dei **giochi a somma costante**, un caso particolare di problema che porta, sempre e comunque, ad avere una circostanza nella quale addizionando i payoff di tutti gli individui si ottiene sempre il medesimo valore (Tabella 3.2).

		Giocatore Due	
		Scelta Uno	Scelta Due
Giocatore Uno	Scelta Uno	(-6, +2)	(-8, +4)
	Scelta Due	(+1, -5)	(+7, -11)

Tabella 3.2: Un esempio di Gioco a Somma Costante

È possibile riuscire a formulare giochi ancora più particolari rimanendo nella sfera della somma costante dei payoff; stiamo parlando del caso in cui, oltre ad avere la proprietà appena citata, il risultato considerato sia sempre pari a zero (**giochi a somma zero**). Possiamo quindi figurare nella nostra mente questi casi come circostanze che terminano assegnando ad un partecipante ciò che ad un altro viene tolto (caso di due soli giocatori); in oltre, assegnando un valore di payoff pari a zero in caso di pareggio, si può riuscire a modellare qualsiasi competizione a due come una partita di calcio, il *pari o dispari* o la *morra cinese* (tabella 3.3) che tra l'altro risulta essere anche un **gioco simmetrico**[1].

[1]Gioco Simmetrico: È un particolare tipo di gioco che attribuisce ad i partecipanti ad esso payoff medesimi in situazioni consimili.

		Giocatore Due		
		Sasso	Carta	Forbici
Giocatore Uno	Sasso	(0, 0)	(-1, +1)	(+1, -1)
	Carta	(+1, -1)	(0, 0)	(-1, +1)
	Forbici	(-1, +1)	(+1, -1)	(0, 0)

Tabella 3.3: Gioco della Morra Cinese in Forma Normale

Per quale motivo però focalizzarci su una proprietà così banale? Risposta semplice: sfruttare questa caratteristica in ambito di rappresentazione. È infatti possibile, attraverso la forma normale, riuscire a costruire un'immagine del gioco contratta, rispetto alle altre, ma che non perda di espressività; la riportiamo, per la *morra cinese* in tabella 3.4.

0	-1	1
1	0	-1
-1	1	0

Tabella 3.4: Rappresentazione della Morra Cinese tramite Payoff Costante

La tabella è di semplice lettura. Vengono riportati tutti i payoff del *Giocatore Uno*, ovviamente considerando che le sue scelte si muovono per riga (si individua la colonna considerando la mossa del *Giocatore Due*). Da questo consegue che cambiando il segno di tutti i valori in tabella 3.4 si avranno praticamente le risultanti delle funzioni per il secondo giocatore. In pratica, la circostanza denotata dall'incrocio della seconda riga e della prima colonna, non fa altro che rappresentare:

- La scelta $\sigma_2 \in \mathbf{S_1}$ da parte di *Giocatore Uno* corrispondente a *carta*;
- La scelta $\sigma_1 \in \mathbf{S_2}$ da parte di *Giocatore Due* corrispondente a *sasso*;
- Il payoff $f_1(\sigma_2 \in \mathbf{S_1})$ del *Giocatore Uno* corrispondente a 1 (vittoria);
- Il payoff $f_2(\sigma_1 \in \mathbf{S_2})$ del *Giocatore Due* corrispondente a -1 (sconfitta).

Si nota facilmente quindi che si riesce ad indicare, in via agevole, una bimatrice[2]

[2]Bimatrice: Di fatto una matrice, riporta però in ogni sua cella due valori al posto di uno. Viene utilizzata, appunto, quando le risultanti non sono singole (cella per cella).

sfruttando una classica matrice singola.

3.1.3 L'Informazione

Concetto principe dei *game* è sicuramente l'informazione, essa può essere definita quanto segue: "*L'informazione è la conoscenza, appartenente agli individui, che permette ad essi di essere di avere la consapevolezza sulla posizione attuale e, decidendo una mossa, anche sulla successiva*". La frase appena scritta traccia esattamente l'immagine di quella che va sotto il nome di **Informazione Perfetta** (**Completa**), mentre ciò che è definibile come **Imperfetta** (o **Incompleta**) lo si ritrova in occasioni nelle quali vengono a mancare dati che non rendono distinguibili, ad alcuni *players*, gli stati (l'informazione non è pubblica).

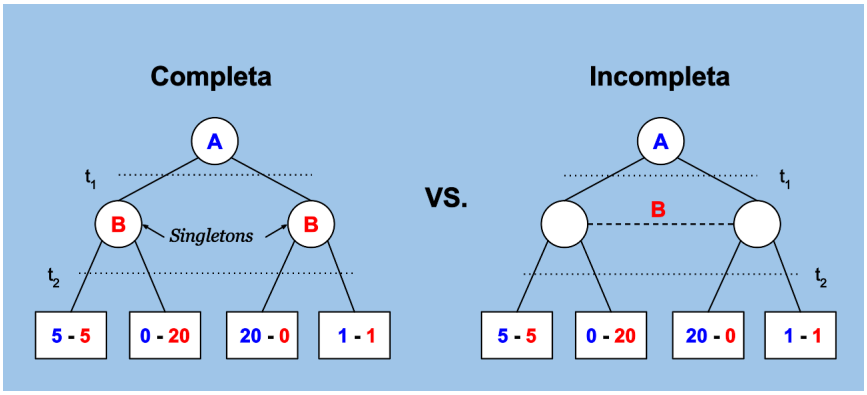


Figura 3.1.3: Informazione Completa contro Informazione Incompleta

Nella figura 3.1.3 possiamo ammirare la differenza tra i due tipi di informazione. In questo caso stiamo prendendo come modello un **Gioco Multistage**[3]. Più precisamente, nell'ipotesi in analisi, esistono due momenti (t_1 e t_2), nel primo muoverà A e nel secondo B. Avremo:

[3]Gioco Multistage: Una particolare tipologia di gioco nella quale questo è diviso in momenti ed in ognuno di essi c'è la mossa di almeno un player.

- Informazione Completa: Se B conosce la mossa di A:

  Si dice che l'**Insieme Informativo** di B è un **Singleton**[14];

- Informazione Incompleta: Se B non conosce la mossa di A:

  L'**Insieme Informativo** di B è un insieme di nodi.

Condizioni sulle quali è necessario avere conoscenza per essere provvisti solo di **Singletons** (e quindi di informazione completa) sono:

- I *Giocatori*;
- Le strategie degli *Stage* passati:

$$\sigma_k^{t_i} \in S_{-i} \quad \text{for all } t_i \in T \text{ Stages Set}$$

- I *Payoff* delle conclusioni possibili partendo dalla *Stage* attuale:

$$f_i(\sigma_j, \sigma_k)\ \forall \sigma_j \in S_i \bigwedge \forall \sigma_k \in S_{-i} \bigwedge \forall i \in N$$

3.1.4 Le Strategie

In questa sottosezione andremo ad illustrare più nello specifico cosa è l'insieme dei profili strategici. Questo, come detto nella definizione di **gioco**, è una delle componenti base di esso e può essere descritto come segue (considerando n il numero dei giocatori ed m^i il numero di opzioni per il giocatore i):

$$S = \{S_1, S_2, S_3, ..., S_n\} \qquad S_i = \{\sigma_1, \sigma_2, ..., \sigma_{m^i}\}$$

$$\text{con:} \qquad \mid S \mid = \mid N \mid = n \quad \bigwedge \quad \mid S_i \mid = m^i$$

Ogni singolo quindi ha diritto, ad esempio in un gioco simultaneo ed ad una mossa, di selezionare un generico σ_j per esprimere la sua posizione; evidenziamo quindi la preferenza del giocatore i scrivendo $\sigma_j \in S_i$ o, più semplicemente, σ_i^* (indicando solo il partecipante), mentre invece le posizioni di tutti i rimanenti

con σ^*_{-i}. Per manifestare quindi i possibili esiti di un gioco per un partecipante i che effettua una scelta ben precisa si potrà scrivere:

$$f_i(\sigma^*_i, \sigma_{-i})\ \forall \sigma_{-i} \in S_{-i} \qquad \text{si noti che esistono} \mid S_{-i} \mid \text{esiti possibili}$$

$$S_{-i} = \{S_1 \bigcup S_2 \bigcup ... \bigcup S_{i-1} \bigcup S_{i+1} \bigcup S_{i+2} \bigcup ... \bigcup S_n\}$$

$$S_{-i} = \bigcup_{S_k \epsilon S \setminus S_i} \text{con } 0 \leq k \leq n$$

Potendo quindi calcolare tutti valori attesi per qualsiasi personaggio non risulterà difficile individuare la preferenza della *choice* σ^*_i su un'altra σ^{**}_i, si avrà ciò se:

$$f_i(\sigma^*_i, \sigma_{-i}) \geq f_i(\sigma^{**}_i, \sigma_{-i})\ \forall \sigma_{-i} \in S_{-i}$$

Quello appena steso non è altro che il concetto di **Dominanza Strategica Debole**. Si dice appunto che una strategia a domina debolmente una strategia b quando il partecipante chiamato ad esprimersi ottiene payoff migliori, o quantomeno uguali, scegliendo a anziché b (qualunque sia la scelta degli avversari).

Parliamo di **Dominanza Strategica Stretta** se la *choice* garantisce, sempre e comunque, un risultato strettamente migliore (non uguale), cioè:

$$f_i(\sigma^*_i, \sigma_{-i}) > f_i(\sigma^{**}_i, \sigma_{-i})\ \forall \sigma_{-i} \in S_{-i}$$

Per rendere l'idea di quello che abbiamo appena discusso riportiamo l'esempio in tabella 3.5.

		Bob		
		X	Y	Z
Alice	A	(3, 3)	(3, 1)	(7, 1)
	B	(3, 2)	(2, 5)	(0, 8)
	C	(2, 5)	(0, 1)	(5, 0)

Tabella 3.5: Dominanza Debole e Stretta di una Scelta

È lampante come in questo caso, per il giocatore uno, la scelta A domini **debolmente** B e **strettamente** C. Ciò porterà Alice a muoversi come indica la prima riga, che è appunto la sua *Solution Concept* (previsione dei comportamenti).

Fino ad ora abbiamo dato per scontato che ogni partecipante dia una risposta netta, è utile però considerare circostanze diverse; ammettiamo, ad esempio, di voler predire situazioni sulle quali abbiamo delle indicazioni che riguardano almeno uno degli aspetti da considerare (ha più senso prendere un ombrello in una giornata uggiosa piuttosto che in una soleggiata). Per poter ragionare in questi termini però, è necessario sfruttare il concetto di probabilità o, meglio, di **distribuzione di probabilità**[4]. In questo caso il procedimento consiste nell'associare ad ogni profilo strategico S_i una funzione che mappi tutte le scelte di un giocatore con valori da zero ad uno (rappresentanti le probabilità sui movimenti):

$$p : \sigma_j \to [0, 1] \quad \text{for all } \sigma_j \text{ in } S_i$$

$$\text{In linea con la proprietà di base :} \qquad \sum_{\sigma_j \in S_i} p(\sigma_j) = 1 \quad \forall S_i$$

Tramite questa seconda tipologia di visione saremo in grado di formulare (e risolvere) quesiti più complessi. Si parla qui di **Strategie Miste** che si con-

[4]Distribuzione di Probabilità: È una funzione, su una variabile, che assegna ad ogni stato (valore) di essa la probabilità che questo si verifichi.

trappongono alle prime (**Strategie Pure**). Naturalmente il calcolo del valore atteso per un determinato giocatore dovrà essere modificato tenendo conto della distribuzione probabilistica:

$$payoff_i = \sum_{\sigma_j \in S_i} p(\sigma_j) f_i(\sigma_j, \sigma_{-i}) \quad \text{for all } \sigma_{-i} \text{ in } S_{-i}$$

Analogamente anche la dominanza subisce variazioni, attestandosi per una decisione σ_i^* su S_i:

- Fortemente:

$$p(\sigma_i^*) f_i(\sigma_i^*, \sigma_{-i}) > p(\sigma_i^{**}) f_i(\sigma_i^{**}, \sigma_{-i}) \; \forall \sigma_{-i} \in S_{-i}$$

- Debolmente:

$$p(\sigma_i^*) f_i(\sigma_i^*, \sigma_{-i}) \geq p(\sigma_i^{**}) f_i(\sigma_i^{**}, \sigma_{-i}) \; \forall \sigma_{-i} \in S_{-i}$$

Con questo abbiamo tutti gli strumenti per comprendere come le persone, poste davanti a questioni da risolvere, ragionino in modo da ottenere il risultato migliore. Ma se, ammessi tutti i piani dei vari avversari, non esistesse la strategia più conveniente? Qualcuno potrebbe ritenere l'inesistenza di una spiegazione, ma ricordando che nella teoria dei giochi si cerca il probabile avvenire e non la soluzione, questa non sarebbe una buona proposizione. Nella sottosezione successiva andremo ad esaminare proprio questo aspetto.

3.1.5 I Punti di Equilibrio

Abbiamo appena enunciato che ci sono casi particolari ove è praticamente impossibile applicare il concetto di **dominanza**. Come abbiamo visto, questo principio, è dato dall'inesistenza di una mossa che, giocata indipendentemente dalle scelte altrui, porti ad una risoluzione massimizzata in qualsiasi finale. Per trovarla possiamo partire da un concetto base che ritroviamo proprio nella dominanza: *un giocatore non sarà mai spinto dalla sua ragione a scostarsi dalla*

posizione ottenuta; quindi la domanda a cui rispondere diventa: è possibile, in un gioco che non presenta dominanze (forti o deboli), riuscire a trovare una proposta di soluzione? Ovviamente si! È possibile, ma naturalmente non sempre.

In questi casi la *Solution Concept* consiste nel ricercare uno o più insiemi di strategie (una per ogni giocatore) che, se selezionati, portano a condizioni che incarnano un equilibrio; si parla allora di **Equilibrio di Nash**, dall'omonimo matematico e vincitore del nobel per l'economia nel 1994 *John Forbes Nash Jr.*[22]. Formalmente ci ritroviamo dinanzi a situazioni di questo tipo quando si verifica tale proposizione: *Viene selezionato un profilo strategico tale che nessuno dei giocatori ha incentivo a deviare con una modifica di tipo unilaterale alla propria tattica.* Matematicamente:

$$u_i(\sigma_i^*, \sigma_{-i}^*) \geq u_i(\sigma_i, \sigma_{-i}^*) \; \forall \sigma_i \in S_i \bigwedge \forall i \in N$$

Questo status però non riesce a garantire nient'altro che una condizione dalla quale risulta essere sconveniente il movimento, ciò significa che esiste l'eventualità nella quale si postuli una soluzione, tramite la scelta di un S^* (insieme dei profili strategici), che risulta essere peggiore per tutti i concorrenti rispetto ad un'altra.

		Fidanzato	
		Balletto	Ristorante
Fidanzata	Balletto	(2, 1)	(0, 0)
	Ristorante	(0, 0)	(1, 2)

Tabella 3.6: Solution Concept in Battaglia dei Sessi

Un aiuto, per comprendere al meglio la nozione di equilibrio, può essere rinvenuto nel celebre gioco *La Battaglia dei Sessi* (tabella 3.6). In questo dilemma popolare abbiamo una coppia di giovani fidanzati che devono decidere su cosa fare per la loro prossima uscita. Le condizioni poste sono le seguenti:

- Il fidanzato preferisce gustare un cena (massimizza il suo guadagno);
- La fidanzata preferisce contemplare un'esibizione di ballo (massimizza il suo guadagno);
- Entrambi traggono giovamento nell'andare in un luogo medesimo (uno dei due non massimizza il guadagno);
- Se non riescono ad accordarsi non escono (non guadagnano nulla).

Il punto di questo *game* è naturalmente la coordinazione, la soluzione sta nel dichiarare in quale luogo i due innamorati dovrebbero andare, senza però essersi parlati prima (**Stage Singolo** o **Multistage** ad **Informazione Imperfetta**); purtroppo per loro, la risposta consiste semplicemente nel recarsi nello stesso luogo, quindi non riusciranno mai ad avere la certezza di incontrarsi. A noi invece, questa sentenza, permette di studiare a fondo il caso; le strategie $(\sigma_{fidanzata} = balletto, \sigma_{fidanzato} = balletto)$ e $(\sigma_{fidanzata} = ristorante, \sigma_{fidanzato} = ristorante)$ costituiscono gli **Equilibri di Nash** (che riportiamo evidenziati in color ciano nella tabella del gioco), questo accade perché, nei due profili strategici, l'uomo (nel primo profilo) e la donna (nel secondo) annullerebbero il loro *payoff* portandolo a zero deviando dalla posizione. Appunto si verifica la condizione di non incentivo a deviare.

		B	
		T	NT
A	T	(-5, -5)	(0, -20)
	NT	(-20, 0)	(-1, -1)

Tabella 3.7: Solution Concept in Dilemma del Prigioniero

Scenario differente invece è quello che troviamo nell'ormai classico *Dilemma del Prigioniero* (tabella 3.7), qui infatti andiamo ad incappare proprio nel problema che prima accennavamo. L'equilibrio, a causa della formazione dei *payoff*, denota una situazione piuttosto inefficiente, è lampante come, se i due potessero accordarsi e godessero l'un l'altro di un buon grado di fiducia, sceglierebbero

di non tradirsi a vicenda per minimizzare la pena. Di certo la selezione di $(\sigma_A = T, \sigma_B = T)$ come *Candidate Solution* (*Solution Concept*) non è un errore ma esclusivamente un limite, limite che deriva comunque dalla sua intrinseca definizione di equilibrio. Immaginando infatti che A e B vogliano collaborare per minimizzare la pena, si evince come, trovandosi in $(\sigma_A = NT, \sigma_B = NT)$, abbiano entrambi istinto a deviare per diventare uomini liberi! La casella verde, in ogni caso, non passa inosservata, essa è la risoluzione **Pareto Efficiente** del sistema, una contingenza definibile come segue: "*Una strategia viene detta Pareto Efficiente se e solo se non esiste nessuna via per aumentare il payoff di una componente senza diminuire quello altrui*", in termini matematici:

$$\neg\exists\sigma_{i'} \in S_i \mid u_i(\sigma_{i'}, \sigma^*_{-i}) \geq u_i(\sigma^*_i, \sigma^*_{-i}) \bigwedge u_j(\sigma_j, \sigma^*_{-i}) \geq u_j(\sigma_j, \sigma^{*'}_{-i})$$

$$\forall j \in N \bigwedge \forall \sigma_j \in S_j$$

È comunque possibile che si formino competizioni, nelle quali esiste una scelta nei vari S_j, che porti ad un equilibrio in cui, oltre al bene del singolo, viene massimizzato anche quello del collettivo. In questi casi, se la condizione di stabilità è unica, non ci saranno problemi di alcuna sorta per gli individui; essi infatti, con certezza assoluta, si dirigeranno verso quella direzione.

3.1.5.1 Problemi noti

Riepilogando ciò che è stato appena scritto elenchiamo brevemente le limitazioni delle quali l'**Equilibrio di Nash** può soffrire:

- Non Unicità;
- Inefficienza;
- Inesistenza.

Le prime due ci sono chiare dagli esempi appena citati mentre la terza non è stata trattata in questa sottosezione. Niente di trascendentale comunque; come si può concepire, data la parola che lo descrive, si tratta semplicemente di un *game*, con un profilo strategico globale, in base al quale non è mai possibile ricadere in una opzione per la quale nessuno non ha mai incentivo a deviare. Utilizzando uno dei giochi descritti in questa tesi possiamo affermare che ad incarnare questa limitazione è sicuramente la *Morra Cinese*.

Detto questo è possibile concludere che l'**Equilibrio di Nash** è sicuramente una condizione che permette di studiare l'evoluzione di un sistema, ma la sua risultante non tiene conto di variabili che esistono nella realtà, come la propensione al rischio o il trascorrere del tempo (è un esempio l'impossibilità di procrastinare una scelta in materia economica).

3.2 I Giochi Cooperativi

Conclusa la parte nella quale abbiamo trattato i giochi strategici, è ora il momento di esporre quella inerente ai **giochi cooperativi**.

Nella vita reale, quando è possibile, le persone tendono a stringere **accordi vincolanti** con l'auspicio di conseguire un valore di beneficio maggiore rispetto a quello che spetterebbe loro senza poter contare su alcuna intesa. È necessario

quindi, come primo passo, definire il concetto di *accordo vincolante*; con questa affermazione non si intende far convergere le scelte dei vari giocatori in modo da ricadere nel profilo strategico **Pareto Efficiente**, ma si va ad indicare la volontà delle persone di unirsi in **coalizioni**, le quali andranno a formare i valori di input per nuove funzioni di *payoff*.

Proprio a causa della formazione di questi gruppi di aggregazione, esistono due tipologie di **giochi cooperativi** o anche detti **coalizionali** (in relazione alla divisione del compenso dell'unione fra i singoli):

- **Giochi NTU** "*Non Transferable Utility*" - In questi giochi la ripartizione degli utili tra gli individui è assegnata a priori;
- **Giochi TU** "*Transferable Utility*" - Siamo dinanzi a *Game* nei quali è possibile l'esistenza di **pagamenti laterali**, cioè modi di assegnare la vincita di coalizione, tra i componenti della stessa, in una maniera ben definita. È necessario sottolineare che, per essere in un **Gioco TU**, devono essere presenti le seguenti caratteristiche:
  - Equivalenza delle funzioni di *payoff* per ogni giocatore;
  - Esistenza di un metodo comune per eseguire i trasferimenti.

Forniamo un esempio di gioco cooperativo per capire cosa stiamo trattando:

"*Un uomo, padre di tre figli, ha bisogno di imbiancare casa. Siccome per fare questo lavoro si deve essere obbligatoriamente in due, l'individuo decide quindi di chiedere aiuto ad uno dei suoi rampolli. Questi decidono di mettere ai voti l'estrazione dello sfortunato, avendo però l'imposizione dal genitore di non poter fare voto circolare (arrivare al pareggio); due dovranno quindi coalizzarsi. Considerando, in maniera valutabile matematicamente, il riscontro del gioco diciamo che:*

- *Chi lavora si stanca (ogni ora di due unità) - $Payoff = -2$;*
- *Chi non lavora può riposarsi su un letto (ogni ora riposa un unità) -*

$Payoff = 1.$

Come formalizziamo i payoff delle coalizioni avendo chiesto il padre aiuto per due ore ed esistendo un solo letto?"

La risposta al quesito è estremamente semplice, la riportiamo chiamando i figli A, B, e C e dichiarando la coalizione che si possono formare:

- $(B, C) \Longrightarrow \quad payoff_A = -4; \quad payoff_{B,C} = 1;$
- $(A, C) \Longrightarrow \quad payoff_B = -4; \quad payoff_{A,C} = 1;$
- $(A, B) \Longrightarrow \quad payoff_C = -4; \quad payoff_{A,B} = 1.$

Ma quale è la *Solution Concept* di questo *Game*? Potremmo essere tentati di dichiarare queste tre opzioni come punti di equilibrio e quindi elencarli come possibili soluzioni, non è sempre vero! Consideriamo che nello stesso gioco due dei figli siano gemelli e il terzo sia il maggiore, nel caso in cui quest'ultimo, agendo di prepotenza, occupi il letto per tutte e due le ore (se non dovesse imbiancare) allora non ci sarebbero tre equilibri; infatti con B e C gemelli:

- $(B, C) \Longrightarrow \quad payoff_A = -4; \quad payoff_B = 1; \quad payoff_C = 1;$
- $(A, C) \Longrightarrow \quad payoff_A = 2; \quad payoff_B = -4; \quad payoff_C = 0;$
- $(A, B) \Longrightarrow \quad payoff_A = 2; \quad payoff_B = 0; \quad payoff_C = -4.$

La soluzione del gioco sta nella prima alternativa; questo perché né C né B giocherebbero mai la strategia che tende a minimizzare il loro beneficio; come possiamo afferrare questo è un caso di gioco coalizionale con pagamenti laterali.

3.2.1 La Funzione Caratteristica

Nel caso appena trattato è stato semplice, leggendo la consegna, calcolare l'utilità singola e di coalizione che i *players* avrebbero potuto avere. Non tutte le volte però ci ritroviamo davanti a situazioni così elementari da valutare, proprio per questo necessitiamo di un mezzo che sia in grado di aiutarci nelle valutazioni;

per i giochi coalizionali questo strumento è detto **Funzione Caratteristica**, è definibile in tale maniera:

"*La **Funzione Caratteristica** è una regola che computa l'utilità assegnabile ad ogni coalizione possibile, fa ciò considerando che i giocatori non facenti parte della stessa non andranno mai a collaborare con quest'ultima.*"

Considerando:

- N - Insieme dei giocatori;
- C - Una coalizione formata;
- S_C - Tutte le strategie della coalizione formata;
- $N \setminus C$ - I rimanenti giocatori (in coalizione);
- $S_{N \setminus C}$ - Le strategie dei giocatori rimasti (in coalizione).

Calcolo $v(C)$ **funzione caratteristica** in due modi possibili:

$$v'(C) = \max_{\sigma_{S_C} \in \sum S_C} \left[\min_{\sigma_{S_{N \setminus C}} \in \sum S_{N \setminus C}} \left[\sum_{i \in C} u_i \left(\sigma_{S_C}, \sigma_{S_{N \setminus C}} \right) \right] \right]$$

$$v''(C) = \min_{\sigma_{S_{N \setminus C}} \in \sum S_{N \setminus C}} \left[\max_{\sigma_{S_C} \in \sum S_C} \left[\sum_{i \in C} u_i \left(\sigma_{S_C}, \sigma_{S_{N \setminus C}} \right) \right] \right]$$

I due risultati ottenuti indicheranno la strategia (o le strategie) che la coalizione andrà a selezionare.

Utilizziamo un gioco "*Transferable Utility*" (tabella 3.8 etichette α, β e γ) a tre giocatori nel quale imponiamo profili di giocata:

$$\sigma_{S_{C_1}} = (\mathrm{A}, \mathrm{R}); \quad \sigma_{S_{C_2}} = (\mathrm{A}, \mathrm{S}); \quad \sigma_{S_{C_3}} = (\mathrm{B}, \mathrm{R}); \quad \sigma_{S_{C_4}} = (\mathrm{B}, \mathrm{S})$$

$$\sigma_{S_{N \setminus C_1}} = X; \quad \sigma_{S_{N \setminus C_2}} = Y; \quad \sigma_{S_{N \setminus C_3}} = Z;$$

α

3 = X			
		2	
		R	S
1	A	1, 4, -2	1, 0, 3
	B	2, -2, 0	0, 3, 4

β

3 = Y			
		2	
		R	S
1	A	2, 5, -3	3, 1, -2
	B	-2, -2, 3	1, 2, 3

γ

3 = Z			
		2	
		R	S
1	A	-3, 0, 4	-2, 3, 1
	B	0, 1, 5	3, -1, -1

δ

		N\C		
		σ_1	σ_2	σ_3
C	σ_1	5, -2	7, -3	-3, 4
	σ_2	1, 3	4, -2	1, 1
	σ_3	0, 0	-4, 3	1, 5
	σ_4	3, 4	3, 3	2, -1

Tabella 3.8: Gioco Coalizionale con Pagamenti Laterali a tre giocatori

Dati i profili siamo in grado di calcolare l'etichetta δ della tabella, quindi le circostanze possibili del gioco con le coalizioni note. Da questo punto possiamo applicare le formule della funzione caratteristica:

$$v'(C) = \max_{\sigma_{S_C} \in \sum S_C} [(-3,4)(1,3)(1,1)(-4,3)(2,-1)] = (2,-1)$$
$$payoff_C = 2 \Downarrow \wedge \quad S_C = \sigma_4$$

$$v''(C) = \min_{\sigma_{S_{N\setminus C}} \in \sum S_{N\setminus C}} [(5,-2)(7,-3)(2,-1)] = (2,-1)$$
$$payoff_C = 2 \Downarrow \wedge \quad S_C = \sigma_4$$

Possiamo quindi concludere che la coalizione formata dai *players* 1 e 2 giocherà sicuramente $\sigma_{S_{C_4}} = (\mathrm{B}, \mathrm{S})$, sperando che 3 faccia un passo falso (rischi) concedendo un aumento dell'utile ricavabile.

Si noti che, in questo caso, $v'(C)$ e $v''(C)$ coincidono; non è sempre così! L'unica sicurezza è che il *payoff* in $v''(C)$ è almeno tanto buono quanto quello in $v'(C)$.

Nel caso di due o più strategie in output dalla funzione caratteristica, abbiamo l'indicazione che quella coalizione non ha un unica preferenza da mostrare.

3.2.2 Differenza tra Giochi NTU e Giochi TU

Limitarsi ad una definizione composta solo da parole non può ritenersi soddisfacente per chiarire la differenza che intercorre tra un **Gioco *Non Transferable Utility*** ed uno nel quale invece i pagamenti laterali vengono ammessi. Per questo motivo formalizziamo in maniera non ambigua la definizione delle due tipologie:

NTU: Un gioco che non ammette pagamenti laterali non è altro che un'accoppiata $G = (N, V)$ dove N rappresenta l'insieme-giocatori e V è una funzione che indica i *payoff* per ogni coalizione, rispettando:

- $V(C) \subset \mathbb{R}^C$;
- $V(C) \neq \emptyset$.

TU: Un gioco nel quale si ha la possibilità di trasferire, tra gli alleati ed in una maniera qualsiasi, l'utile conseguito è una coppia $G = (N, v)$ dove N, come sopra, è la *Grand Coalition*[5] mentre v rappresenta la funzione caratteristica vista in precedenza (nel *game* essa deve rispettare la proprietà $v(\emptyset) = 0$, cioè nulla deve andare disperso).

La differenza basilare tra i due giochi è data quindi dalla **modalità di calcolo del *payoff*** di coalizione e la sua successiva ripartizione tra le parti; questo perché, come riportato precedentemente, nei giochi **NTU** la distribuzione dell'utile viene definita a priori, rendendo impossibile l'applicazione della **funzione caratteristica** e lasciando libera strada esclusivamente a principi incentrati sulle preferenze, mentre invece nei giochi a pagamenti laterali lo strumento citato è applicabile, ma deve essere supportato da un'ulteriore operazione tesa a dispensare i proventi.

Come cambia realmente un *game* se rimuoviamo il vincolo che fissa in maniera preventiva la suddivisione dell'utile tra gli individui coalizzati? Possiamo

[5]Grand Coalition: È la massima coalizione che posso formare in un gioco cooperativo; essa equivale sempre all'insieme totale dei giocatori (N).

rispondere a questa domanda rifacendoci ancora una volta al *Dilemma del Prigioniero* in tabella 3.1. Innanzitutto, per comodità, possiamo applicare una trasformazione al gioco, cambiamo il segno a tutti i *payoff* tramutando il tutto in un *cost game*[6]; fatto questo è possibile calcolare $V(C)$ considerando tutte le 2^N coalizioni possibili ed i loro valori:

- $V(\emptyset) \Longrightarrow costo = 0$;
- $V(A) \Longrightarrow costo \geq 1$;

  Se tradisce potrebbe essere libero e se non tradisce è costretto almeno ad un anno di carcere ($costo \geq 0 \bigwedge costo \geq 1$).
- $V(B) \Longrightarrow costo \geq 1$;
- $V(\{A, B\}) \Longrightarrow \{(5, 5), (0, 20), (20, 0), (1, 1)\}$.

Apprezziamo meglio i risultati del *game* con la figura 3.2.1; questa riporta con etichetta α la soluzione in strategie pure (i quattro punti colorati in ciano) e con etichetta β la stessa proposta ma con la possibilità di strategie miste (qualunque punto della superficie del poligono). Va considerato che sui singoli il costo è maggiore, o uguale, all'unità, perciò i partecipanti tenderanno a tradire.

[6]Cost Game: È un gioco sul quale vengono indicati dei costi piuttosto che dei ricavi, si va quindi alla ricerca del risultato minore.

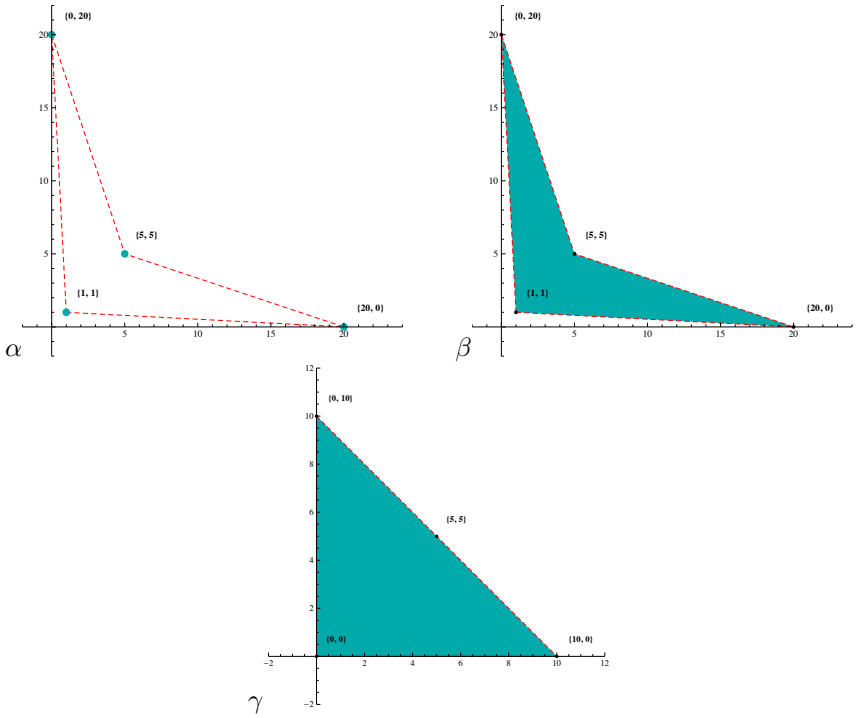


Figura 3.2.1: Dilemma del Prigioniero in Gioco Coalizionale

L'etichetta γ invece indica un gioco nel quale sono ammesse ripartizioni diverse di payoff tra i singoli delle coalizioni. Va quindi calcolata la funzione caratteristica:

- $v'(\emptyset) = v''(\emptyset) = 0$;
- $v'(A) = max\,[min\,[\{(5,*),(0,*)\},\{(20,*),(1,*)\}]] = 1$;
  $v''(A) = min\,[max\,[\{(5,*),(20,*)\},\{(0,*),(1,*)\}]] = 1$;
- $v'(B) = max\,[min\,[\{(*,5),(*,0)\},\{(*,20),(*,1)\}]] = 1$;
  $v''(B) = min\,[max\,[\{(*,5),(*,20)\},\{(*,0),(*,1)\}]] = 1$;
- $v'(\{A,B\}) = max\,[min\,[5+5]] = 10$;
  $v''(\{A,B\}) = min\,[max\,[5+5]] = 10$.

È evidente ora che la soluzione del *Dilemma del Prigioniero* **Coalizionale TU** è una qualsiasi combinazione di ordinata/ascissa appartenente alla superficie

triangolare segnalata in figura 3.2.1 etichetta γ. Il punto che andremo a scegliere infatti sarà una qualsiasi combinazione (x, y) possibile tale che il *payoff* sia non superiore al valore della funzione caratteristica applicata alla *grand coalition*.

Scegliere il punto che indica la spartizione dei guadagni è il fine dei **Giochi TU**, per perseguirlo esistono molteplici vie; queste spaziano dalla elementare divisione in parti uguali, a funzioni più complesse. Nel caso di queste ultime potremo valutare le circostanze in due modi:

- Per **Insiemi**: proporre uno o più vettori di suddivisione del *payoff*, costruiti in maniera tale da essere congruenti alle assegnazioni di valore richieste dal gioco;
- Per **Indici**: suddivisione in base all'apporto che un *player* riesce a fornire.

3.3 L'Integrale di Choquet

Il nostro scopo, fino ad ora ed in questo capitolo, è stato quello di presentare e tentare di risolvere problemi a più partecipanti; il metodo, o che dir si voglia l'idea centrale, che è stato proposto consiste nel commisurare tra loro i possibili valori risultanti, dati da un'espressione di preferenza del singolo combinata alla strategia degli avversari. Questi dati però, come visto, non sono forniti da zero ma devono essere calcolati tramite apposite funzioni più o meno complesse. L'ostacolo però, che in alcuni casi si incontra, non è dato dall'intricata regola da applicare, ma piuttosto dal riuscire a formulare tale funzione in maniera da rispondere alle necessità che essa deve rispecchiare. Quand'è che questo ostacolo si propone però?

Un evento nel quale può presentarsi questo scoglio può essere ritrovato in situazioni in cui dei giocatori sono chiamati a prendere decisioni basate su criteri multipli. In questi casi infatti, la complicanza, che non sussiste nelle selezioni derivanti da parametri singoli, è data dal dover dare un significato unico

all'insieme di valutazioni, che devono essere fatte criterio per criterio. La condizione necessaria diventa quindi riuscire a produrre, o comunque utilizzare, uno strumento in grado di rendere possibile l'**aggregazione** di questi ultimi.

Per nostra fortuna, anche se non da molto, la tematica esposta è largamente studiata in campo di ricerca; ad occuparsi di questo argomento infatti è il ramo della matematica che va sotto il nome di **Ricerca Operativa**, una particolare area, conosciuta anche come **Teoria delle Decisioni**, nella quale si formulano problemi in maniera scientifica e non ambigua in modo da riuscire a scovare la soluzione migliore in base alla condizione delle cose. Ciò che ci interessa principalmente di questa scienza è la disciplina della **Multi Criteria Decision Making** che, come consiglia nel nome, approfondisce proprio le problematiche a cui siamo interessati.

Qual è però il motivo di tanta complessità nella proposta di funzioni per operazioni di integrazione? Non è possibile utilizzare tecniche triviali?

Purtroppo no, non è sempre possibile utilizzare modelli comuni e la ragione risiede nella conformazione delle varie misure che devono essere considerate. Esistono due grandi distinzioni in merito di criteri in questo campo, questi ultimi infatti, rispetto al loro insieme, possono avere proprietà di estensione:

- **Additiva**;
- **Non additiva**.

Il primo caso è quello che non crea alcun tipo di problema, qui infatti, i parametri, non vanno a condizionarsi tra loro, rendendo possibile la fusione di essi, in un valore unico, estremamente semplice (è un esempio la media ponderata); le circostanze che ricadono nel secondo caso sono però diverse, ci troviamo immersi in queste quando può risultare contraddittorio, al fine in merito, utilizzare misure che creano una sorta di compensazione incoerente. Si pensi ad un'azienda che valuta il suo impatto pesando vendite e soddisfazione dei clienti,

dando troppa importanza al secondo termine potrebbe bilanciare il suo indice, rendendolo positivo, anche con un pessimo fatturato, decretando l'inconsistenza dell'indicatore.

Una delle metodologie per risolvere queste situazioni è l'**Integrale di Choquet**, introdotto dal matematico francese *Gustave Choquet*; cerchiamo di capire meglio di cosa si tratta. Questo strumento ha l'intento di misurare l'utilità attesa di un evento incerto, dettato dai *Decision Makers* posti davanti a criteri non additivi, sfruttando le capacità del set di questi ultimi.

Per determinare l'**Integrale di Choquet** è quindi necessario introdurre la definizione di **Capacità di un Set**:

Consideriamo $\mathcal{L}$ un algebra su Ω, allora la **Capacità** μ è una funzione monotona che mappa tutti i subset definiti in $\mathbb{R}^+$:

$$\mu : \mathcal{L} \to \mathbb{R}^+$$

Che soddisfa le seguenti caratteristiche:

- $\mu(\emptyset) = 0 \bigvee \mu(\Omega) = 1$;
- $\mu(A) \leq \mu(B) \qquad$ per ogni $A \subset B$ con $A, B \in \mathcal{L}$.

Inoltre, non essendo una misura additiva, può dare risultati di questo genere:

- $\mu(A \bigcup B) \leq \mu(A) + \mu(B) - \mu(A \bigcap B)$ **Misura Sub−Additiva**;
- $\mu(A \bigcup B) \geq \mu(A) + \mu(B) - \mu(A \bigcap B)$ **Misura Super−Additiva**.

Solo ora possiamo dare la definizione del metodo, considerando $X \in L^\infty(\mathcal{L})$, ergo ammesso μ sia f-misurabile avendo:

$$f : \Omega \to \mathbb{R} \qquad \forall x \in \mathbb{R} : \{\omega \mid f(\omega) \geq x\} \in \mathcal{L}$$

l'**Integrale di Choquet** è descritto in questi termini[8, 9]:

$$Ch \int_{\Omega} X \, d\mu = \mu(X) = \int_{-\infty}^{0} (\mu([X \geq x]) - 1 \, dx + \int_{0}^{+\infty} \mu([X \geq x]) \, dx$$

In semplice maniera, dati:

- Un insieme di tutti i "pesi" possibili dell'algebra $\mathcal{L}$

  $\mathcal{O} = \{\mu_1, \mu_2, ..., \mu_{2^{|\Omega|}}\}$

- Un sottoinsieme di $\mathcal{O}$ contenente i $\mid \Omega \mid$ pesi di interesse

  $\{l_1, l_2, ..., l_{|\Omega|}\} \subset \Omega \quad \mid \Omega \mid \geqslant 2$

- I valori di tutte le alternative

  $\{\omega_1, \omega_2, ..., \omega_{|\Omega|}\} \in \Omega$

Viene calcolato quanto segue:

$$Ch([\{\omega_1, \omega_2, ..., \omega_{|\Omega|}\}, \mathcal{O}]) = \sum_{i=1}^{|\Omega|} (\omega_i - \omega_{i-1}) * \mu(l_i) \quad \text{con} \, \omega_0 = 0$$

Può essere utile adesso esporre un piccolo esempio per togliere ogni dubbio su questo strumento di calcolo. Riportiamo in tabella 3.9 i risultati ottenuti per media, media pesata (pesi 45%, 40% e 15%) ed **Integrale di Choquet** di tre alternative (i colori di sfondo delle celle indicano il rank ottenuto secondo la misura in colonna, rispettivamente verde, giallo e rosso indicano primo, secondo e terzo posto).

	CPU	GPU	Ram	Avg.	W. Avg.	Choquet
Sistema Uno	95	80	90	88,3	88,25	87
Sistema Due	100	90	70	86,6	91,5	92
Sistema Tre	80	95	95	90	88,25	92,5

Tabella 3.9: Comparazione di Strumenti Multi-criterio

La situazione definita vede tre possibili configurazioni di un sistema hardware che si basano su votazioni verso *cpu*, *gpu* e *ram*. Per quel che riguarda le capacità è stato definito tale scenario, ovviamente $\mu_{cpu-gpu-ram} = 1$ e $\mu_{\emptyset} = 0$:

$$\mu_{cpu} = 0,4; \quad \mu_{gpu} = 0,3; \quad \mu_{ram} = 0,1;$$

$$\mu_{cpu-gpu} = 0,8; \quad \mu_{cpu-ram} = 0,5; \quad \mu_{gpu-ram} = 0,6.$$

Nel nostro caso i valori di interesse, avendo utilizzato l'ordine seguito in tabella, sono:

$$\mathcal{O}_{configurazioni} = \{\mu_{cpu-gpu-ram} = 1; \quad \mu_{gpu-ram} = 0,6; \quad \mu_{ram} = 0,1\}$$

Procediamo al calcolo per confermare i risultati riportati (solo per il sistema uno):

$$Ch_{Sistema\,Uno}([\{95, 80, 90\}, \mathcal{O}_{configurazioni}]) =$$

$$= [(95 - 0) * 1] + [(80 - 95) * 0,6] + [(90 - 80) * 0,1] = 87$$

In questo esempio si capisce come sia stato d'aiuto l'utilizzo di questo integrale. Esso infatti ci permette di capire quale è la reale combinazione di fattori migliore; ciò è appunto dato dal fatto che i criteri utilizzati si condizionano tra loro nel campo della scelta.

Capitolo 4

Il Prototipo di Protocollo Proposto

Ci siamo dilungati molto, nel secondo capitolo, nel descrivere in minuziosa maniera le *Mobile Ad-hoc networks* ed il loro funzionamento; questa esposizione, ora, ci rende in grado di valutare codesti sistemi in base alle scelte implementative che sono state compiute al momento di creazione della rete. Le selezioni che vengono fatte, sulle quali possiamo essere chiamati ad essere giudici, sono ovviamente un numero finito, anzi potremmo dire che le opzioni a nostra disposizione non sono affatto parecchie confrontate con la vastità di comportamenti ipotizzati, a priori, per questi organismi. Avendo quindi, da una parte, una quantità esigua di regole e dall'altra un bisogno di tecniche che riescano a far fronte all'ampia gamma di criticità che possono insorgere, la somma di queste due condizioni non può altro che risultare nella volontà di creazione di metodologie differenti atte a migliorare, sotto il punto di vista prestazionale, quelle esistenti. In altre parole, ciò che si è tenta di fare è la messa in piedi di un nuovo **protocollo di comunicazione** per interconnettere e far comunicare i

vari device delle reti di cui stiamo parlando.

L'intento del capitolo è proprio questo, cioè proporre un nuovo modello di protocollo per *Mobile Ad-hoc network*; la parte interessante consisterà nel fare ciò che è stato appena detto, attingendo dalle nozioni di modellazione per problemi a più partecipanti viste in questo scritto.

4.1 La Motivazione della scelta

Teoria dei Giochi e Multi-criteria Decision-making sono due importanti discipline che hanno conosciuto, e stanno tuttora conoscendo, un roseo periodo date le elevatissime potenzialità a loro disposizione per risolvere alcuni problemi che, con le convenzionali tecniche, non sarebbero possibili da trattare. Ci riferiamo quindi a tutta quella massa di quesiti ai quali facciamo fatica a rispondere, ma ahimè a cui non possiamo sottrarci se vogliamo giungere ad un punto di arrivo. Partendo da questa frase e guardando alla linea temporale, ritroviamo applicazioni di queste due materie di studio che vanno a spaziare dalle aste alle decisioni di entrata in mercato, dalla produzione di determinati beni all'analisi di situazioni di rischio, fino ad arrivare ad essere messe in pratica addirittura per lo studio di reti (siano esse sociali o di dispositivi [4, 10]). Proprio l'ultimo punto è quello che più attrae il nostro interesse, le potenzialità esposte dai due insegnamenti, per più di una volta, hanno dimostrato quanto essi siano in grado di ottimizzare o, addirittura, di rimuovere ostacoli bloccanti allo sviluppo; per questo motivo si ritiene che dall'applicazione di queste tecniche nel nostro ambito di interesse, riguardante appunto le reti, non si possa altro che trarre giovamento in termini di prestazione, a patto che si riesca a sviluppare un modello congruo a rappresentare la situazione in adeguata maniera.

4.2 Descrizione Introduttiva

Presentare un nuovo protocollo, come detto in precedenza, deve essere un'operazione in grado di donare un'innovazione tangibile almeno in un settore di applicazione di esso. In questa tesi andremo a presentare un modello diverso per la comunicazione che chiameremo **GMCRP** (**G**ame Theoretical **M**ulti-criteria Decision-making **C**lustering **R**outing **P**rotocol), appunto un protocollo pensato per essere utilizzato in *Mobile Ad-hoc Networks*. Come ci suggerisce il nome, questo set di regole è progettato in maniera tale da clusterizzare la rete, ovvero suddividerla in tante piccole realtà, in modo tale che lo scambio di pacchetti possa essere organizzato nella migliore maniera possibile; utilizzando una tipologia clusterizzata infatti, si ha la sicurezza di una più snella comunicazione quando si è obbligati a comunicare in broadcast per il floading dei dati, inoltre si riescono ad ottimizzare le procedure di *discovery* dei possibili *path* sui quali far correre i bit indirizzati ad una precisa destinazione conosciuta. Per ottenere queste due principali migliorie, derivanti dalla clusterizzazione, è comunque necessario avere un device che vada a fare le veci di un router nella "sub-rete" venutasi a creare grazie al raggruppamento di nodi; un dispositivo di questo genere tuttavia è impossibile da avere in una *MANet* (la sua presenza la trasformerebbe in una *infrastructure* o, nella migliore delle ipotesi, in un ibrido) indi per cui, la soluzione che pare preferibile, semplificante anche la costruzione dei cluster, è quella che ricade nell'elezione di un portavoce dell'aggregazione stessa. Quest'ultimo nodo sarà quindi utilizzato come simil-router per quanto riguarda la comunicazione *intra-cluster*, dovrà avere quindi piena conoscenza della posizione (da intendersi a livello di percorso) di tutti i dispositivi membri del cluster di cui egli è il rappresentante.

4.3 La Terminologia

In ogni definizione di protocollo non può mancare una sezione dedicata alla terminologia che viene utilizzata all'interno dello stesso. Nel **GMCRP** ritroviamo:

- *Nodo*

  Con questa parola si intende un qualsiasi dispositivo presente in rete;

- *Id*

  L'identificatore (*id*) non è altro che un numero intero associato ad ogni dispositivo della *Mobile Ad-hoc network* per poterlo riconoscere in maniera univoca nella rete, per questo motivo deve essere unico, ovvero non deve esistere la possibilità che due nodi condividano lo stesso *id*. Questa condizione viene garantita dalla rete stessa, ogni dispositivo infatti, prima di diventare parte attiva, è tenuto a richiedere ad essa l'assegnazione di un numero identificativo; a questo punto, la rete, tramite il floading periodico dell'informazione topologica, ha la capacità di indicare un *id* che non è stato precedentemente concesso. Si noti che, se un nodo fuoriesce dalla rete, si preferisce non riutilizzare il suo *id* per un nuovo dispositivo, questo perché in alcune tabelle di *routing* potrebbero essere rimaste informazioni al suo riguardo;

- *Choquet Score*

  Nel protocollo che promuoviamo è di vitale importanza che ad ogni nodo sia associato un punteggio, ciò permetterà di creare un *rank* tra i vari dispositivi. Spiegheremo in seguito la metodologia di calcolo e la motivazione di utilizzo dello stesso;

- *Cluster*

  Un cluster non è altro che un raggruppamento di nodi ottenuto tramite l'utilizzo di un determinato algoritmo, chiamato algoritmo di clusteriz-

zazione (nel caso delle *MANet* queste procedure si basano su informazioni geografiche piuttosto che, come accade solitamente, su *funzioni di similarità*[1]);

- *Host Cluster*

  Indichiamo con il termine *Host Cluster* di un determinato nodo n l'intero gruppo di appartenenza del nodo in causa, costruito attraverso la procedura di clusterizzazione;

- *Nodo Indefinito*

  È un qualsiasi nodo della rete che sta tentando l'accesso ad essa; può ricevere ed inviare pacchetti, ma la mancanza di un *id* (non ancora assegnato) lo limita alla richiesta d'accesso;

- *Nodo Non Associato*

  Questa è la condizione di un nodo entrato ormai a far parte della rete, ma non ancora appartenente a nessun cluster. Ovviamente, prima del processo di clusterizzazione, tutti i nodi della rete risultano come non associati;

- *Clusterhead*

  Il *Clustered* non è altro che il nodo portavoce del cluster a cui egli appartiene, esso viene scelto, per questo compito, secondo particolari direttive. Questo device deve essere in grado di poter entrare in contatto con ogni membro del cluster di cui esso è la "testa", così da svolgere il lavoro per il quale è stato prescelto;

- *Cluster Member*

  Un nodo n è considerato *Cluster Member* di un cluster c se e solo se n fa parte di c e non è il *clusterhead* di quest'ultimo. Naturalmente un device può essere membro di più gruppi;

[1] **Funzione di Similarità**: È una funzione che assegna a due dati in input un punteggio che indica la similarità tra gli stessi.

- *Gateway*

  Con la parola *Gateway* ci riferiamo ad un nodo che sia *cluster member* di almeno due *cluster*. L'appartenenza molteplice permette, a questo attore, di svolgere la funzione di *discovery* per ogni gruppo in cui egli è presente. Essere *gateway* in un numero n di *cluster* è condizione sufficiente ma non necessaria per affermare che il nodo in oggetto può riuscire a raggiungere n o più *clusterhead*;

- *l-Gateway*

  Un *l-Gateway* è un *Gateway* tra (almeno) due cluster che però non appartiene ad entrambi. Questa circostanza si ottiene quando c'è un collegamento fisico tra due nodi membri di cluster differenti;

- *Vicino ad n Hop(s)*

  Solitamente, con il termine *vicino*, si identifica, in relazione ad un determinato nodo, un dispositivo che raggiunge il suddetto tramite la propagazione di onde elettromagnetiche. Per conformazione del modello però, indicheremo come vicini, di un particolare device, anche gli hardware che che riescono a portare a termine la funzione di raggiungimento utilizzando un determinato numero di salti che indicheremo con n;

- *Link*

  Un *link* non è altro che la consapevolezza di raggiungibilità da parte di due o più nodi. Essendo in campo wireless, quindi avendo connessioni senza fili, siamo soggetti al problema della direzionalità dei link, cioè alla loro doppia natura. I vari dispositivi presenti in rete non sono tutti equipaggiati con lo stesso hardware, per la precisione non tutti hanno la stessa antenna; per questo motivo possono crearsi due tipologie di link (come mostrato in figura 4.3.1): bi-direzionali e uni-direzionali. Rispettivamente, queste due tipologie, descrivono una situazione nella quale ambo i nodi

possono ricevere ed inviare pacchetti tra loro e una circostanza dove la comunicazione è praticabile verso una sola via (purtroppo in questo caso il mittente non si avvede di aver raggiunto il destinatario, nell'immagine quindi, riferendoci al collegamento uni-direzionale, il nodo rosso non si accorge di aver raggiunto il blu perché quest'ultimo non possiede abbastanza *range* per far pervenire un *ack*);

- *Hello Message*

  È il fondamento di qualsiasi protocollo che gestisca le *Mobile Ad-hoc Networks*, altri non è che un pacchetto che viene inviato in maniera periodica ed in *broadcast*; risulta di vitale importanza, essendo l'unica metodologia che permette di informare i dispositivi presenti in rete sulla topologia che la stessa ha assunto.

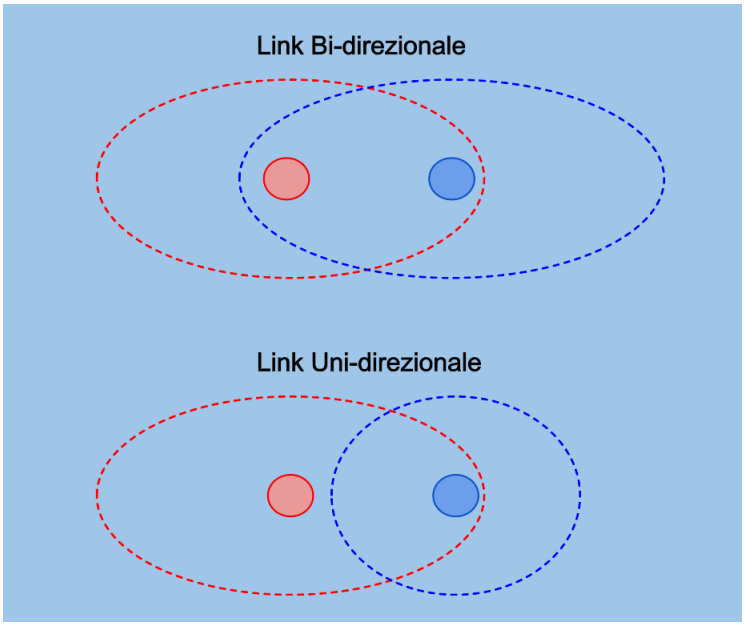


Figura 4.3.1: Direzionalità dei link wireless

4.4 Le Strutture Dati

Ogni sistema, per essere ben formato, ha bisogno di strutture dati logiche dove porre le informazioni, parziali ed intermedie, che rendono possibile le funzionalità dello stesso. Il protocollo **GMCRP** ovviamente non fa eccezione, per far sì che le operazioni descritte in esso siano possibili, questo descrive la *Tabella di vicinanza Nodi* e la *Tabella di vicinanza Cluster*.

4.4.1 Tabella di Vicinanza Nodi

Questa particolare struttura serve ad immagazzinare le informazioni sui link presenti nella rete. Essa è ovviamente mantenuta da tutti i nodi ed ogni occorrenza contiene indicazioni sia su vicini ad un *hop* che a due *hops*. I dati mantenuti in questa struttura sono:

ID_ VICINO [n bit] Questo dato non è altro che l'*id* assegnato ad nodo che riesce a raggiungere il device, mantenente la suddetta tabella, con un suo pacchetto;

CHOQUET_ SCORE [32 bit] Un numero reale che indica lo *Score* di *Choquet* del mittente;

IP_ VICINO [32 bit / 128 bit] Utilizzando il protocollo TCP/IP sappiamo che ogni device in rete ha un indirizzo ip assegnato, risulta utile mantenerlo in memoria in questa tabella per le future operazioni;

TIPO_ VICINO [1 bit] Considerando che in questo protocollo, per costruzione, vogliamo mantenere informazioni su vicini fino a due *hops*, abbiamo bisogno di indicare la lontananza del nodo di riferimento. Con lo *zero* indichiamo un vicino stretto (un *hop*) mentre con lo *uno* andremo ad indicare un nodo a due *hops* (deve essere comunque verificata la condizione di raggiungibilità);

TIPO_NODO [1 bit] Questo flag è utilizzato per indicare se il nodo vicino è un *Clusterhead*. Lo è se il bit è impostato a *True* (valore 1);

TIPO_LINK [2 bit] Il campo in oggetto viene utilizzato per indicare la tipologia di link che c'è tra il vicino e chi mantiene il dato; per comprendere il significato del valore rappresentato è necessario consultare anche il campo *TIPO_VICINO*. I possibili scenari sono riportati in figura 4.4.1 (si noti che i risultati sono solo sei perché, ricordiamo, i link bi-direzionali individuabili sono solo quelli in entrata).

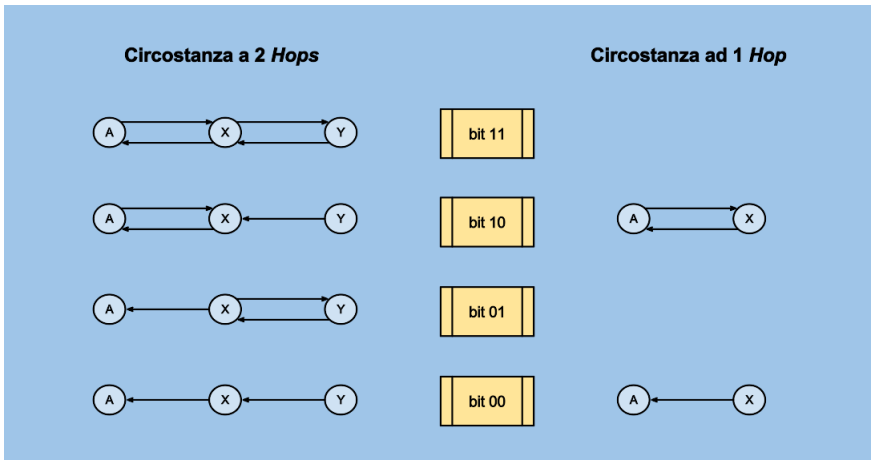


Figura 4.4.1: Rappresentazione dei link protocollari in bit

4.4.2 Tabella di Vicinanza Cluster

La *Tabella di Vicinanza Nodi*, come appena visto, è una struttura utilizzata per descrivere tutti i collegamenti presenti in rete; perciò non ci da informazioni in merito alla clusterizzazione di essa. Introduciamo quindi la *Tabella di Vicinanza Cluster*, questo insieme di dati ci permetterà, attraverso la sua divulgazione, di scovare tutti gli *l-Gateway* presenti in rete. La tavola della quale stiamo parlando è molto simile alla precedente ma si differenzia per costruzione; qui infatti, a far parte di essa, non sono più tutti i nodi vicini ma solo alcuni,

quelli appunto che forniscono informazioni su cluster adiacenti senza però essere Gateway. Forniamo un esempio in figura 4.4.2.

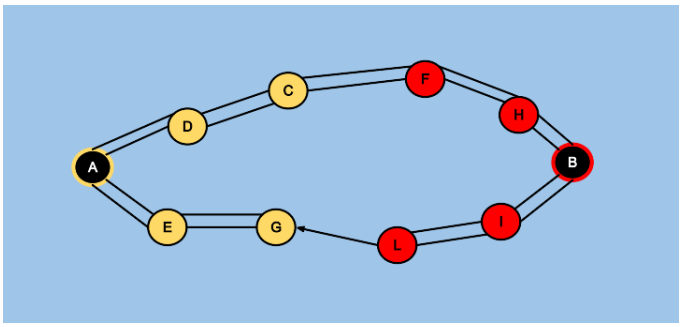


Figura 4.4.2: Cluster in collegamento tramite L-Gateway

Nell'immagine precedente è chiaro come i nodi *C*, *F* e *G* ricevano informazioni, rispettivamente da *F*, *C* ed *L*, riguardo *clusterhead* di unioni attigue. Questi nodi quindi, pur non potendosi definire *Gateway*, sono in grado di essere sfruttati per la comunicazione; vengono detti *l-Gateway* (*l* come limitati appunto) e mantengono una tabella dove indicano:

ID_ VICINO [*n* bit] Analogo al bit della *Tabella di vicinanza Nodi*;

IP_ VICINO [32 bit / 128 bit] Analogo al bit della *Tabella di vicinanza Nodi*;

ID_ HEAD_ VICINO [*n* bit] È il dato indicante l'*id* del *clusterhead* che mantiene il nodo vicino;

IP_ HEAD_ VICINO [32 bit / 128 bit] È l'IP del *clusterhead* che mantiene il nodo vicino;

TIPO_ LINK [2 bit] Ovviamente solo per circostanza ad un *hop* (riferimento in bit in figura 4.4.1).

Ma in che caso viene creata una *entry* per questa tabella?

> Un nodo m è una entry per la *Tabella di Vicinanza Cluster* di un nodo n **se e solo se**:

- m riesce a raggiungere n in un *hop*;
- m non appartiene al cluster di n (e viceversa).

Ammettere *discovery* di *path* simili a questo fluidifica di gran lunga le operazioni di circolazione dei pacchetti, basarsi solo su *Gateway* completi (membri dei cluster raggiungibili) non permetterebbe al network di esprimersi al meglio, costringendo i dati a percorsi non ottimizzati.

4.5 I Pacchetti Base

Qualsiasi sistema informatico di comunicazione necessita che in esso circolino pacchetti-dato (a 32 bit) in grado di informare le varie componenti circa le operazioni vitali di tale sistema. Come visto nel secondo capitolo, queste ultime, almeno per quello che riguarda le *Mobile Ad-hoc Networks*, sono riassumibili nella *route discovery* e nella *route maintenance*; nel caso del *Game Theoretical Multi-criteria Decision-making Clustering Routing Protocol* però viene inserita un'ulteriore routine essenziale. Questa è la *topology fomation and discovery*, dividere infatti la rete in cluster è un'operazione primaria nel **GMCRP**, come lo è la conoscenza della divisione stessa per i nodi interessati ad una zona ben precisa dell'universo-rete.

4.5.1 Pacchetti di Topologia

In questa categoria ritroviamo i pacchetti che informano i nodi sulla conformazione geografica della rete. Qui esiste una sola tipologia di pacchetto ed a farne le veci è lo *Hello Message*.

4.5.1.1 Hello Message

Come detto in precedenza **ogni** *MANet* si avvale di questa tipologia di messaggi, la differenza in questo caso però consiste nell'utilizzo che si fa di questi. Qui, i pacchetti inviati tra nodo e nodo, non informano solo ed esclusivamente sui vicini immediati ma possono essere sfruttati sia per ottenere delucidazioni su device a due *hops* che sulla "forma" assunta dalla rete. Ciò è ottenuto grazie alla trasmissione delle due strutture dati tabellari mantenute dai nodi, il messaggio viene formato per:

Tabella di vicinanza Nodi (riconosciuta dal primo bit inviato, il bit M, settato a *zero*):

STATO_MITTENTE (**SM**) [2 bit] Indica lo stato del nodo che invia lo *Hello Message*, devo utilizzare 2 bit perché i valori possibili sono tre: *Nodo Non Associato* (bit 00), *Clusterhead* (bit 01) e *Cluster Member* (bit 10);

NUMERO_VICINI [13 bit] Questa variabile indica il numero di vicini (uno e due *hops*) "scoperti" dal nodo mittente;

CHOQUET_SCORE [32 bit] Medesimo lo si ha nella *Tabella di vicinanza Nodi*;

TIPO_VICINO (**V**) [1 bit] Analogo al bit della *Tabella di vicinanza Nodi*;

TIPO_LINK (**TL**) [2 bit] Significato analogo al bit della *Tabella di vicinanza Nodi*;

STATO_VICINO **(SV)** [2 bit] Indica lo stato del nodo vicino, come per il mittente si utilizzano 2 bit essendo in presenza di tre possibilità;

IP_VICINO [32 bit / 64 bit] Stesso dato mantenuto nella *Tabella di vicinanza Nodi*;

ID_VICINO [n bit] L'*id* assegnato al vicino in questione (Non essenziale).

Si noti che non possono essere inviati gli IP dei vicini uno dietro l'altro. Con IPv4 (indirizzi a 32 bit), ad esempio, posso inviare solo cinque nodi alla volta

perché prima di ogni quintetto devo inoltrare le informazioni dei device e cioè i dati **V**, **TL** ed **SV** per un totale di 5 bit; infatti in $\mathbb{N}$ la divisione di 32 per 5 è proprio 5. Per inoltrare un numero maggiore di vicini si utilizza quindi un semplice espediente: ogniqualvolta si raggiunge il limite massimo si rinvia la trafila di 30 bit composta da **V**, **TL** ed **SV**.

Tabella di vicinanza Cluster (riconosciuta dal primo bit inviato, il bit M, settato a *uno*):

NUMERO_ CLUSTER [13 bit] Questa variabile indica il numero di *host cluster* per il nodo in questione;

IP_ HEAD_ VICINO [32 bit / 128 bit] È l'indirizzo IP del *clusterhead* per il device che invia questa informazione;

ID_ HEAD_ VICINO [n bit] L'*id* del *clusterhead* in questione (Non essenziale).

Analogamente al messaggio per la vicinanza dei nodi, i primi 32 *bit* vengono utilizzati per indicare la tipologia di dato che si sta inviando ed il numero di device che si vanno ad elencare.

In figura 4.5.1 possiamo esaminare un messaggio *Hello* del protocollo. Nel caso volessimo inserire anche gli *id*, allora li innesteremo appena dopo lo *Choquet Score*, per la vicinanza dei nodi, e oltre l'indicazione del numero di cluster, per la tabella di vicinanza relativa ad essi; ovviamente con una opportuna rottura sull'informazione, avendo a disposizione solo 32 *bit* alla volta.

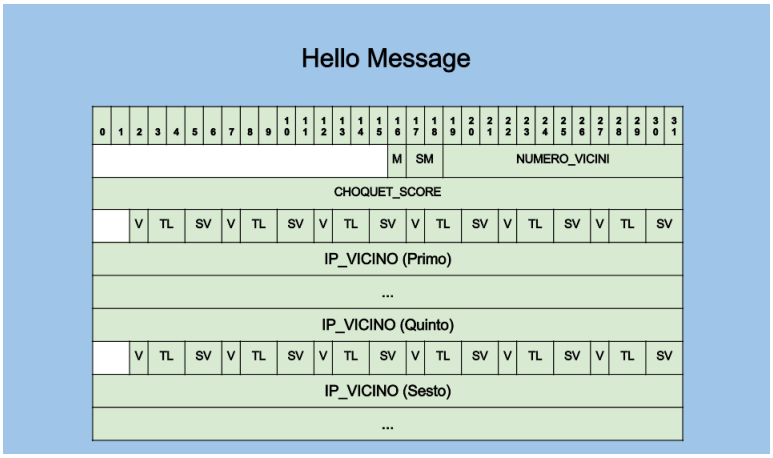


Figura 4.5.1: Parte di un Hello Message con Indirizzi a 32 bit

4.5.2 Pacchetti di Routing

I pacchetti di routing sono particolari tipi di dato che permettono ai device di riuscire ad avere informazioni sui vari cammini presenti in rete. Ritroviamo in questa categoria il *Route Request* (**RREQ**), il *Route Replay* (**RREP**) e il *Route Error* (**RERR**).

Scegliamo di non soffermarci nella descrizione di questi pacchetti per quel che riguarda il **GMCRP**, la decisione viene presa perché in più di un protocollo vengono utilizzate le medesime tipologie di struttura; appare infruttuoso ripetere la loro conformazione in questo elaborato. Ricordiamo comunque, sotto forma di elenco, la loro principale funzionalità:

- Route Request - Nel caso un nodo voglia spedire ad un altro device un pacchetto e non conosca neanche un *path* verso questo, allora invia un **RREQ** (in floading). Il *Request* è un pacchetto che generalmente termina il suo corso quando incontra il destinatario oppure sotto alcune regole protocollari;

- Route Replay - È il pacchetto che torna verso il mittente se il **RREQ** ha incontrato il destinatario. Il **RREP** utilizza le informazioni arrivate ad i nodi "toccati" dalla richiesta antecedente per sfoltire il flooding di dati. Questo pacchetto permette la creazione di un *path* tra chi vuole comunicare;
- Route Error - Nel caso un mittente utilizzi un determinato *path* e l'invio fallisse per l'uscita di un device dal percorso, allora il dispositivo che non riesce più a contattare quest'ultimo spedisce a ritroso, con le politiche di un **RREP**, un pacchetto di errore. Se, provati tutti i *path*, non è possibile effettuare la consegna allora si genera una nuova richiesta.

4.6 Le Operazioni Basilari

Per ottenere gli obiettivi che si prefigge, il nostro protocollo deve descrivere le operazioni base delle quali si fregia. In questa sezione andremo a presentare le principali, in modo da riuscire a capire il reale meccanismo di gestione rete.

4.6.1 La Clusterizzazione

Se fossimo chiamati ad esporre una ed una sola caratteristica di questo protocollo, non ci sarebbero dubbi sulla scelta da effettuare. Ciò che rende più chiara l'idea, in merito a questa tipologia di routing, è senza ombra di dubbio il **processo di clusterizzazione**. Esso, in questo caso, serve a definire una scala gerarchica per uno *status* totalmente privo di note distintive, fornendoci l'abilità di valutare meglio il nostro contesto che, altrimenti, sarebbe mancante di ogni tipo di struttura. Il fine ultimo è quindi quello di avere un ordinamento in grado di semplificare le azioni di gestione del network.

Ogni algoritmo di clustering, per reti mobili *Ad-hoc*, deve descrivere un criterio di associazione di nodi e, nel caso lo preveda, anche una regola di decisione

per eleggere un nodo allo stato più congruo al suo comportamento ed al suo essere. Nel nostro caso, non considerando l'*indefinito*, ogni nodo può assumere le condizioni di *non associato*, *membro* oppure *clusterhead*. Se per quel che riguarda i primi due gradi, siamo nell'ambito di un range di distanza secondo alcune regole, nell'ultimo ci troviamo a dover selezionare il device attraverso metriche più fini con le quali si prova a garantire una configurazione spaziale ottimale. È proprio qui che andiamo ad utilizzare lo *Score* di *Choquet* citato nella parte relativa alla terminologia.

In molte routine di selezione di portavoce si tende a semplificare il processo fino a ridurlo all'osso; il problema però, a volte, consiste proprio nell'aver snellito troppo il sistema di elezione, si configurano infatti circostanze nelle quali è lampante che non sia stato usato alcun tipo di criterio, ne è un esempio l'impiego del *LIC* (Lowest ID Cluster Algorithm) [7] in presenza di dispositivi eterogenei nella stessa rete. Scelte di questo genere sono spesso giustificate dalla volontà di non sottoporre a sforzi (calcoli) i dispositivi delle *Mobile Ad-hoc Networks*, si deve però considerare che questi, come tutti, sono in continuo sviluppo ed ormai sono arrivati, pur mantenendo la caratteristica di estrema mobilità, ad avere una potenzialità non indifferente, tanto da poter far fronte anche a richieste computazionali più "pesanti". Questo ha quindi, soprattutto negli ultimi anni [11, 2], incoraggiato la comunità scientifica nell'intraprendere scelte più acute per ciò che riguarda la designazione dei *clusterhead*, sempre e comunque però considerando il peso del consumo energetico. Per quel che riguarda questa tesi, in codesto ambito, lo strumento per compiere la "scelta astuta" è l'integrale di *Choquet*, per il calcolo appunto dello *Choquet Score*.

Ci siamo occupati del mezzo matematico appena citato nel terzo capitolo e abbiamo asserito che fondamentale, per la sua corretta applicazione, è la dichiarazione puntuale di tutti i valori del *power set* composto dai criteri scelti in

un problema *MCDM*, cioè delle $2^{|criteri|}$ **capacità** μ, in modo tale da riuscire a calcolare la funzione:

$$Ch \int_{\Omega} X \, d\mu = \mu(X)$$

in tale maniera:

$$\int_{-\infty}^{0} (\mu([X \geq x]) - 1 \, dx + \int_{0}^{+\infty} \mu([X \geq x]) \, dx$$

il che equivale a:

$$Ch([\{\omega_1, \omega_2, ..., \omega_{|\Omega|}\}, \mathcal{O}]) = \sum_{i=1}^{|\Omega|} (\omega_i - \omega_{i-1}) * \mu(l_i) \quad \text{con} \, \omega_0 = 0$$

Il punto in questione è allora riuscire a definire l'insieme $\mathcal{O}$, sia come elementi che come valori degli stessi. Per quel che riguarda il *Game Theoretical Multi-criteria Decision-making Routing Protocol*, vengono definiti quattro specifici criteri fissi e si lascia ai sistemisti di turno la scelta per quel che riguarda l'impostazione delle capienze; si ottiene così un modello parametrico che può essere adattato a situazioni differenti in base al "peso" che si decide di imputare ad ogni singolo subset.

I criteri selezionati per il calcolo dello *Score* sono:

Cardinalità-Cluster [C] Non è altro che il numero dei membri del cluster che si formerebbe scegliendo un determinato nodo come *clusterhead*. Considerato che il nostro protocollo prevede che ogni gruppo sia un *four-hops-diameter cluster* (il membro più distante è lontano al massimo due *hops* dal "centro"), questo numero non è altro che la somma della conta dei vicini con quella degli adiacenti di questi. Dare molto peso a questo criterio vuol dire preferire *clusterhead* che creino unioni di nodi numerose (di diametro pari a cinque);

D1H/D2Hs [D] Questo rapporto è ottenuto dividendo il numero di nodi raggiunti in un *hop* con quelli raggiunti a due da un possibile *clusterhead*. Più il numero è alto e più il cluster si sviluppa nell'intorno immediato del portavoce. Impostare una valore elevato per le capienze contenenti questo indicatore significa prediligere clusterizzazioni che non si espandono molto a distanza, risultando in una conformazione diversa dei *gateway*;

Indice-Prestazionale [P] Come detto è possibile, anzi probabile, che il network si componga di dispositivi eterogenei tra loro. Per questo motivo è utile etichettare ogni device con un indice di prestazione (banalmente che tenga conto anche dell'irradiamento dell'antenna) in grado di fornire un sunto delle sue potenzialità. In relazione alle caratteristiche considerate si privilegerà un *clusterhead* più prestazionale a discapito di altri che faticano a soddisfare gli stessi bisogni;

Batteria [B] È del tutto naturale tener conto, in una rete mobile, dell'energia rimanente e dei dispositivi. Con questo indice non facciamo altro che prediligere i device con un carica energetica residua elevata. La scelta del peso deve essere effettuata considerando al meglio la rete che si andrà a sfruttare. Se è bene avere un dispositivo che possa svolgere per molto tempo il compito di *clusterhead*, dobbiamo comunque considerare che, per motivazioni legate all'algoritmo, questo potrebbe decadere dal suo grado ben prima dell'esaurimento della carica.

Detto questo avremo il seguente insieme:

$$\begin{aligned}\mathcal{O} \;=\; \{&\mu_\emptyset, \mu_C, \mu_D, \mu_P, \mu_B, \mu_{CD}, \mu_{CP}, \mu_{CB}, \mu_{DP},\\ &\mu_{DB}, \mu_{BP}, \mu_{CDP}, \mu_{CDB}, \mu_{DBP}, \mu_{CPB}, \mu_{CDPB}\}\end{aligned}$$

E procederemo a scegliere i 14 valori che permetteranno il calcolo dello *Choquet Score* nella seguente maniera (riportiamo in veste grafica, nella figura 4.6.1, il medesimo calcolo e ricordiamo che questo non cambia per nessuna delle permutazioni ammissibili degli addendi):

$$\psi(Ch) = (C - \omega_0) * \mu_{CDPB} + (D - C) * \mu_{DPB} + (P - D) * \mu_{PB} + (B - P) * \mu_B$$

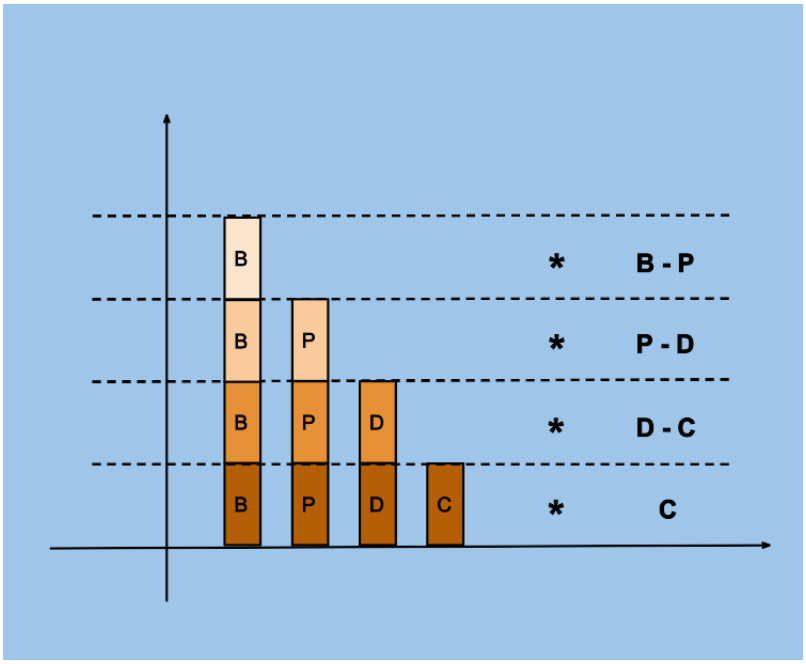


Figura 4.6.1: Rappresentazione grafica dello Choquet Score

Ora che conosciamo l'equazione per il computo del "valore" di un nodo, possiamo passare a descrivere come utilizzarla, delineando un algoritmo, per la creazione dei cluster. Da adesso e fino alla fine della sottosezione, quando parleremo di vicini di un nodo n, intenderemo indicare i dispositivi fino a due *hops* di distanza da questo (cioè qualsiasi valore del bit **V** nella tabella di vicinanza

dei nodi) e legati solo con link esclusivamente bi-direzionali.

Ogni device è visto dalla rete, dopo l'assegnazione di un indirizzo IP, come *Non Associato* cioè invia in broadcast un *hello message* nel quale imposta i 2 bit del suo stato (**SM**) al valore 00; questa non è altro che la situazione di partenza di ogni network che non possiede ancora una reale topologia strutturata. Partiamo da questo caso iniziale per poter descrivere la procedura di formazione dei cluster. Il compito del **GMCRP** è quello di formare coalizioni di diametro pari a 5 dispositivi, l'operazione da realizzare è quindi scovare il miglior candidato portavoce e "costruire" attorno a lui la subnet. Ciò è ottenuto scegliendo tra tutti i nodi *Non Associati* quello che riporta lo *Score* di *Choquet* maggiore, per poi considerare tutti i suoi vicini come membri del gruppo appena creato; fatto questo non resta che reiterare il processo fino a che tutti i device in rete non siano diventati *Clusterhead* o *Member Node*, cioè risultino impostati i due bit **SM** ai valori 01 o 10.

Il primo accorgimento da adottare riguarda la stima del punteggio, siccome essa è ottenuta considerando anche informazioni non residenti nel nodo stesso, c'è bisogno che queste pervengano prima di effettuare delle scelte; perciò, non appena un dispositivo entra nello stato **SM** pari a 00, è obbligato ad aspettare *timer_presentazione* secondi nei quali vengono inviati e ricevuti pacchetti *Hello* (forniti di magic number) tesi ad indicare i relativi vicini ad ogni device. Scaduto questo timer si può procedere nell'intento mostrato prima, ogni dispositivo *Non Associato* setta un ulteriore timer (*timer_nass*) entro il quale aspetta di essere contattato da un portavoce e diventare membro, se ciò non succede ed egli è vicino ad altri nodi diventa *clusterhead*, altrimenti torna allo stato originario. A scongiurare il "pericolo" di *clusterhead* troppo attigui tra loro arriva in aiuto un terzo timer (*timer_clusterhead*), è il tempo in cui due device a distanza minore o uguale a due *hops* possono rimanere portavoce nello stesso momento, alla sua

scadenza uno dovrà cedere la *leadership*.

Riassumendo l'idea centrale in punti:

- Un nodo che non sente dispostitivi vicini con *Score* più alto è *Clusterhead*;
- Un nodo è sicuramente membro dello *host cluster* di cui un suo vicino è *Clusterhead*, questo ha uno *Choquet Score* più alto del suo;
- Un nodo che sente n *clusterhead* è membro di n cluster, cioè è un *Gateway Node*.

Si noti che per non eleggere troppi *Gateway* si utilizza la seguente regola:

> Se un nodo a, già membro di un cluster x, riceve un *Hello* da un componente b dei cluster x e y che lo indica come possibile *Gateway* xy, questi declina l'offerta perché b, che si trova nello stesso *path* è già gateway per i due cluster (si veda la figura 4.6.2 etichetta ϵ, dove S e T rifiutano di appartenere al cluster con portavoce U).

In figura 4.6.2 riproduciamo la formazione dei cluster ottenuta con questo metodo, ovviamente non riportiamo tutta le azioni ma le dividiamo in maniera intuitiva (si considerino i nodi ordinati in modo decrescente per *Score* come riportato nella matrice seguente disposta per colonne-righe dove vengono evidenziati i *Clusterhead*, es. L è il quarto valore più alto).

Y	F	Q	X	A	T
B	C	E	G	K	
S	U	Z	W	I	
L	D	V	M	N	
H	P	R	J	O	

Tabella 4.1: Ordinamento nodi per Score in supporto alla figura 4.6.2

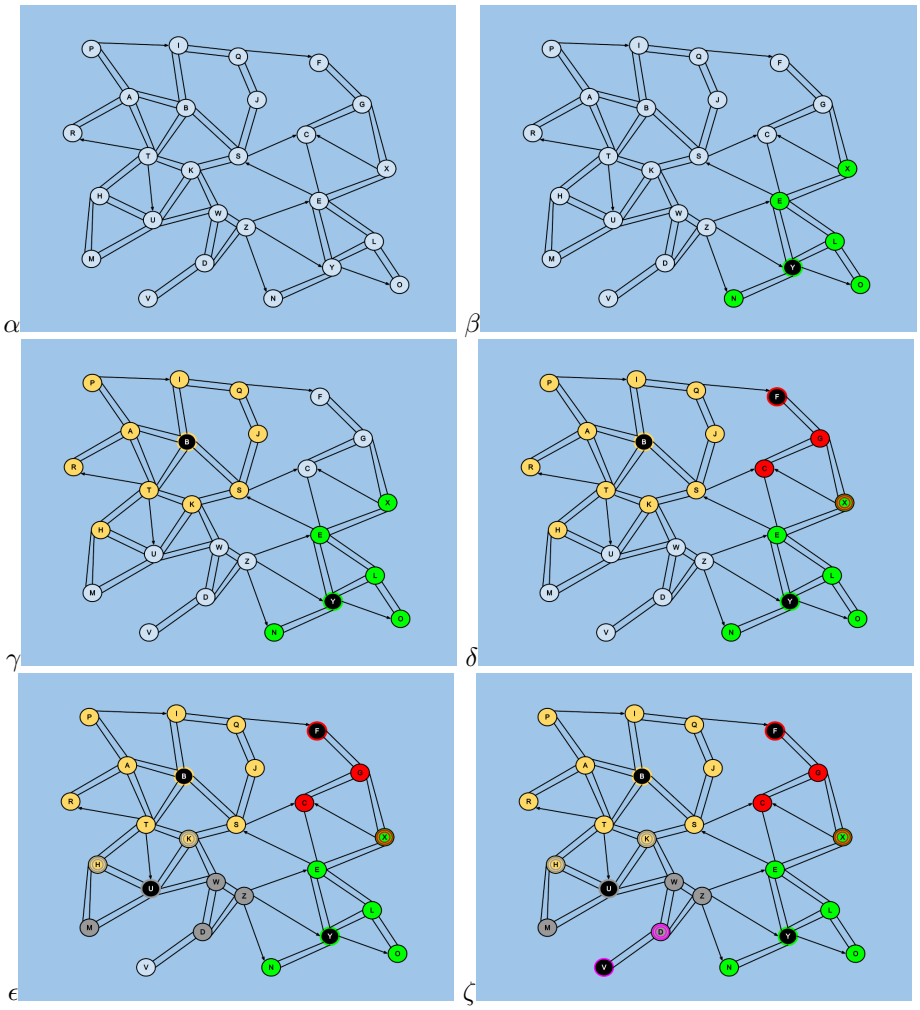


Figura 4.6.2: Clusterizzazione di una MANet secondo lo Choquet Score

Riportiamo in ultimo lo pseudocodice della nostra routine:

```
1.    Node n gets an IP Address;
2.    ETICHETTA-UNO
3.    n goes in Non Associato Mode;
4.    timer_n := 0;
5.    n starts timer_n;
6.    While (timer_n < timer_presentazione):
7.        n sends Hello Message in broadcast setting the Hopcount := 0
8.        n updates his Tabella di Vicinanza Nodi as specified by Hello Messages
9.        If n receives an Hello Message with Hopcount==0
10.       Then:
11.           n sets Hopcount := 1 of the received Hello Message and forwards it
12.   timer_n := 0;
13.   n starts timer_n;
14.   While (timer_n < then timer_nass):
15.       If n receives an Hello Message from a Member Node or Non Associato
16.       Then:
17.           n updates his Tabella di Vicinanza Nodi
18.       Else If n receives an Hello Message from Clusterhead
19.       Then:
20.           n updates his Tabella di Vicinanza Nodi
21.           n goes in Member Node Mode
22.           n stop timer_n
23.           JUMP to ETICHETTA-DUE
24.   If Tabella di Vicinanza Nodi is empty
25.   Then:
26.       JUMP to ETICHETTA-UNO
27.   Else:
28.       n goes in Clusterhead Mode;
29.   ETICHETTA-DUE
30.   While True
31.       If n is in Member Node Mode
32.       Then:
33.           If n receives an Hello Message
34.           Then:
35.               It updates his Tabella di vicinanza Nodi
36.           If n loses his Clusterheads
37.           Then:
38.               If n has the best Choquet Score in his zone
39.               Then:
40.                   n goes in Clusterhead Mode
41.               Else
42.                   n goes in Non Associato Mode
43.                   JUMP to ETICHETTA-UNO
44.       If n is in Clusterhead Mode
45.       Then:
46.           If n receives an Hello Message from a Member Node or Non Associato
47.           Then:
48.               n updates his Tabella di Vicinanza Nodi
49.           Else If n receives an Hello Message from Clusterhead
50.           Then:
51.               timer_n := 0;
52.               n starts timer_n;
53.               While (timer_n < timer_clusterheads):
54.                   Wait()
55.               If n is still in contention && has worse Choquet Score
56.               Then:
57.                   n goes in Member Node Mode;
```

4.6.2 Il Routing di Dati

Raggruppati i nodi della rete in tanti cluster è il momento di capire come questi rendano possibile il *routing* di pacchetti tra device e device; cioè con quale sistemi di base si va a svolgere la *discovery* dei *path* ammissibili, sui quali poi far circolare l'informazione.

Come prima cosa dobbiamo descrivere le due tipologie di adiacenza che permettono la comunicazione tra unioni gerarchizzate di nodi, troviamo:

- Collegamento Mono-Direzionale:

  Questo allacciamento si ottiene quando, tra due cluster, l'insieme intersezione dei membri risulta vuoto ma esiste almeno un link mono-direzionale tra un appartenente del primo gruppo ed uno del secondo. In caso di più link, per mantenere la condizione di clusters collegati mono-direzionalmente, è necessario che gli allacciamenti siano tutti dello stesso verso (tutti da cluster x a cluster y);
- Collegamento Bi-Direzionale:

  Apparentemente questo collegamento può sembrare di semplice definizione, al pari del precedente. Così non è però, allacciamenti di questo genere infatti possono essere formati tramite tre diverse vie. La **prima**, come si può intuire, consiste in un link bi-direzionale tra due nodi appartenenti a cluster diversi. La **seconda** è invece costituita dalla presenza di un *Gateway Node* (ricordiamo che un nodo Gateway è membro di due cluster, quindi è raggiunto in link, o doppio link, bi-direzionale da due o più *Clusterhead*). Infine è considerato un collegamento Bi-direzionale di cluster anche una **terza** alternativa; questa si verifica quando, tra due cluster, non esistono *Gateway* ma sono presenti due o più *l-Gateway* con allacciamenti Uni-direzionali tra loro. Almeno uno di questi però deve essere di "verso" opposto a tutti gli altri, esempio in figura 4.6.3).

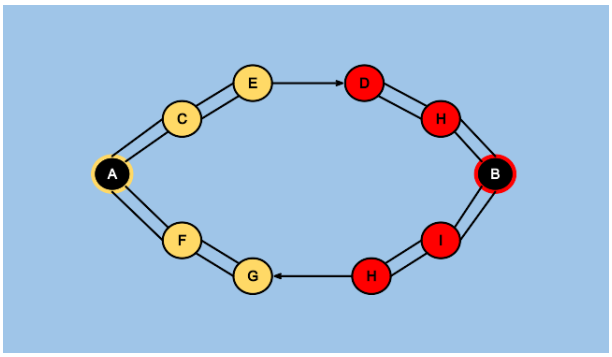


Figura 4.6.3: Collegamento bi-direzionale tramite links mono-direzionali

Detto questo possiamo spendere delle parole per ciò che riguarda l'invio e la ricezione di dati nel **GMCRP**. Utilizzando la tecnica del *source routing*, appare quasi pleonastico descrivere tutta la procedura di funzionamento, dato il fatto che, il metodo in questione, è sfruttato in più di un protocollo di gestione per *MANet's*; ci limitiamo quindi a riportare gli aggiustamenti necessari.

Gerarchizzare una *Mobile Ad-hoc Network* è un procedimento che ha un fine molto specifico nel campo dell'invio di informazioni, questo consiste nel voler ridurre al minimo il *flooding* dei dati senza rinunciare alla forte interconnessione di rete, intrinseca nella tecnologia. Detto questo riportiamo la classica divisione collezionabile in qualsiasi protocollo che si avvalga di *source routing*:

- *Route Discovery*

  Questa è la procedura attraverso la quale, un determinato nodo mittente, cerca di riuscire a scovare un *path* (o una *route*) verso un nodo destinazione. Si ottiene inviando in *flooding* il pacchetto **RREQ** a tutti i nodi della rete e aspettando che giunga al device voluto. Quest'ultimo avrà l'onere di rispondere allo speditore attraverso un **RREP**, con il quale, nel caso di operazione a buon fine, sarà possibile computare il percorso voluto;

- *Routing*

  L'operazione di *routing* è materialmente quella con la quale si inviano i dati del messaggio tra una sorgente ed una destinazione. Questa routine può essere utilizzata solo nel caso in cui sia già avvenuta la computazione dei *path* richiesti, si interrogano le tabelle che mantengono i dati relativi a questi e se esiste una via conosciuta si richiama la procedura in oggetto; se il cammino non è ancora riportato allora si richiama la *discovery*.

  Nella sventurata contingenza in cui un link appartenente ad un tragitto noto venisse a mancare, l'ultimo device raggiungibile dal *routing* avrà l'obbligo di segnalare la circostanza attraverso un **RERR** che andrà ad informare, come avviene con la *Route Reply*, il mittente.

Come accennavamo prima è necessario riportare le regole utilizzate per adattare il *source routing* al nostro protocollo. Innanzitutto è opportuno sottolineare che la tipologia della quale stiamo parlando è di tipo *on demand*, quindi adatta a quelle situazioni nelle quali si preferisce riempire il meno possibile le tabelle di vicinanza dei dispositivi. C'è però una considerazione da fare, i dispositivi a due *hops* di distanza fra loro hanno piena conoscenza della topologia da loro stessi generata; ergo le loro strutture, nella zona di interesse, contengono informazioni pari a quelle di una rete guidata da un approccio *Table Driven*. Come sfruttare ciò quindi? Sappiamo che i dati tenuti in memoria, riguardo i nodi prossimi, arrivano fino al secondo passo di distanza, allo stesso tempo siamo a conoscenza che un *Clusterhead* ha membri, del suo cluster, che non sorpassano il suddetto spazio. Questo principio ci suggerisce che, appena formate le subnet, abbiamo "gratuitamente" la possibilità di servirci di alcuni *path* (quelli dal portavoce ai limiti della sottorete); ciò sancisce l'obbligo di avvalersi delle richieste solo ed esclusivamente se il nodo destinatario è al di fuori del cluster del mittente, delineando una situazione ibrida nella quale sono i *Clusterhead* a decidere il da

farsi. È proprio questa la caratteristica che minimizza il numero circolante di pacchetti-sistema nella rete, favorendo una migliore diffusione dei pacchetti-dato e garantendo quindi un indice globale di prestazione superiore.

Andiamo ora a capire, in maniera schematica, con quali regole il *source routing* per il Game Theoretical Multi-criteria Decision-Making Routing Protocol, agisce in maniera adattata a quest'ultimo (ovviamente la parte interessata è esclusivamente quella relativa alla *discovery* essendo il *routing* una banale interrogazione di tabella):

> m (mittente) vuole inviare un messaggio a d destinatario (N.B. si consideri che un *Clusterhead* conosce sempre un *path* verso un suo *Cluster Member* e viceversa):

1. Se m conosce un *path* verso d:
   (a) Utilizza il *path*;
2. Se m non conosce un *path* verso d:
   (a) Genera un **RREQ** e lo invia verso il suo *Clusterhead* (Se m è *Clusterhead* salta il passaggio)
   (b) Il *Clusterhead* che riceve il **RREQ** lo inoltra a tutti i portavoce dei cluster adiacenti collegati Bi-direzionalmente (sfrutta ovviamente i *Gateway* o gli *l-Gateway* che generano Bi-direzionalità tra gruppi)
   (c) Ogni *Clusterhead* che riceve il **RREQ**:
      i. Se ha già ricevuto lo stesso pacchetto:
         A. Lo scarta
      ii. Altrimenti:
         A. Si reitera tutto il processo dal punto 1 (come se questo *Clusterhead* fosse m ai fini dell'algoritmo).

Tramite queste norme è quindi possibile ottenere *route* come mostrato in figura 4.6.4; si tenga presente che la generazione di **RREP** è affidata comunque al nodo

destinatario, così da ottenere un controllo sui *path* conosciuti dai *Clusterhead*. Basarsi su regole di rete derivanti dalla gerarchia creata è troppo debole come strategia, quindi si passa direttamente al *routing* solo quando m ed s sono nello stesso gruppo.

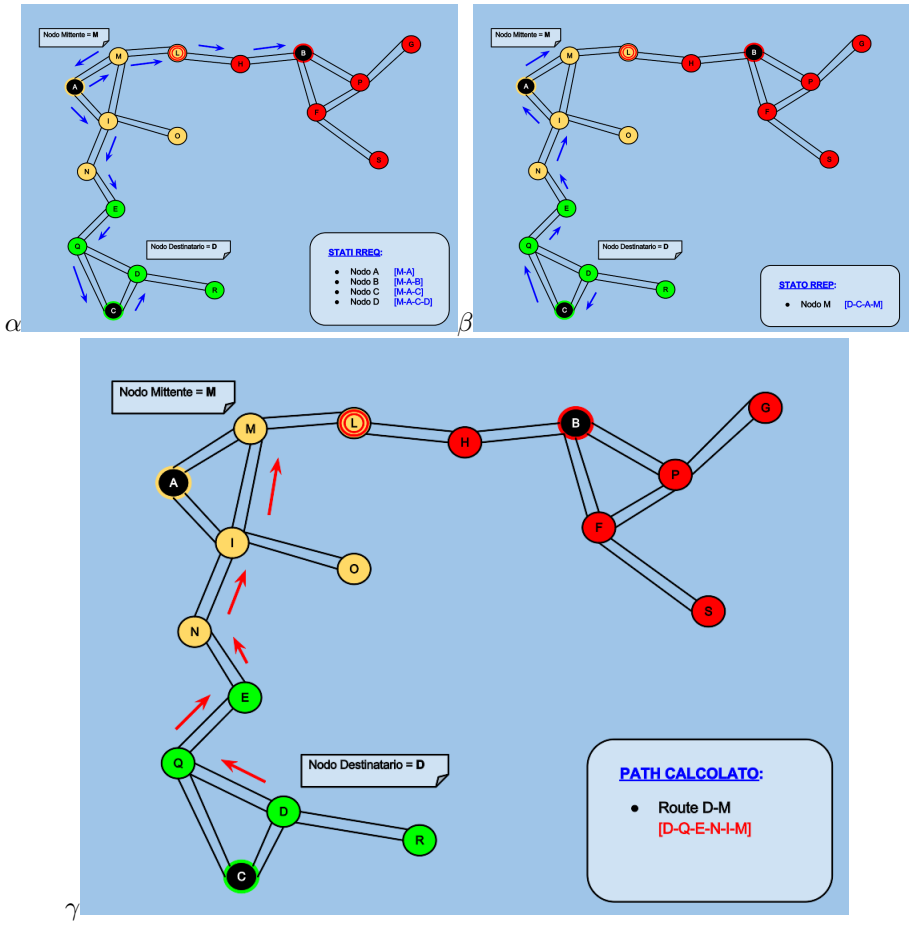


Figura 4.6.4: Procedura di Discovery per un determinato cammino

4.7 Le Operazioni di Ottimizzazione

4.7.1 Stimolo della Partecipazione

Fino ad ora, per caratterizzare una rete *Ad-hoc*, ci siamo focalizzati sulle funzionalità base di essa e forse abbiamo dato per scontato un concetto fondamentale. Stiamo parlando dell'essenza che rappresenta sistemi di questo genere, cioè quelli privi di un'infrastruttura centrale, che non è altro la cooperazione tra i device associati.

Come sottolineato più e più volte, per arrivare a destinazione, un pacchetto ha bisogno di essere preso in consegna e poi inoltrato da una serie di nodi intermedi, ma i dispositivi, per svolgere l'attività appena citata, compiono uno "sforzo", calcolato in termini di computazione ed energia, per il quale non ricevono alcun compenso. Questa condizione, unita alla mancanza di un'autorità centrale, può portare a situazioni nelle quali gli appartenenti alla rete decidono di non collaborare, visto che non ricavano praticamente nulla nel fungere da ponti. Come è elementare immaginare però, un atteggiamento di queso genere, non può altro che portare al collasso del sistema; infatti se consideriamo il **beneficio** attribuibile solo al nodo mittente (perché è egli a voler spedire), che i nodi intermedi vedono la loro operazione di inoltro esclusivamente come **costo** e che solo il destinatario ha una posizione **neutra** (ricevere non richiede costi e non costituisce utilità data l'assenza della richiesta), allora possiamo prevedere che lo scambio avverrebbe solo per device ad un solo *hop* di distanza, decretando quindi il tramonto della comunicazione *multi-hops* e, in maniera derivata, del sistema.

Cerchiamo di descrivere la situazione esposta attraverso un modello di gioco strategico[20] che ci permetta di intuire la *Solution Concept* in termini di condotta dei partecipanti, già esposta nel periodo precedente; avremo:

- $N = \{1, 2, 3, ..., n\}$

Insieme indicante i device partecipanti alla rete appena costituita;

- $S = \{\sigma_1, \sigma_2, \sigma_3, ..., \sigma_n\}$ *con* $\sigma_i \in \{0, 1\}$

  Spazio delle strategie della rete intera, rispettivamente 0 ed 1 equivalgono a non cooperare e cooperare;

- $N_C = \{i \in N \mid \sigma_i = 1\}$

  Insieme indicante tutti i nodi che decidono per la collaborazione;

- $r_i(S) = \sum_{1 \leq j \leq n \bigwedge j \neq i} \sigma_j \in S$

  Ricavo ottenuto dal dispositivo i, dipende dalle scelte di tutti i nodi escluso proprio i;

- $c_i(\sigma_i) = \sigma_i$

  Costo sostenuto dal dispositivo i, dipende esclusivamente dal nodo i;

- $u_i(S) = r_i(S) - c_i(S) = (\mid N_C \mid -1) - \sigma_i$

  Utile ottenuto dal dispositivo i, dipende dalle scelte di tutto S (tutta la rete, nessun nodo escluso).

Dato questo prototipo, risulta triviale il calcolo dell'equilibrio di Nash; ogni nodo i ha infatti due soluzioni tra le quali scegliere (che comportano altrettante alternative in ambito di utilità):

Collaborare

$u_i(S) = \mid N_C \mid -2$

Non-Collaborare

$u_i(S) = \mid N_C \mid -1$

È ovvio come un qualsiasi dispositivo abbia incentivo a deviare verso la strategia $\sigma_{dispositivo} = 0$ e rimanervi, ne consegue che l'equilibrio di Nash è verificabile **se e solo se** si verifica la condizione $\mid N_C \mid = 0$, cioè quando:

$$N_C = \{\emptyset\}$$

In altri termini, raggiungiamo la situazione di stabilità solo quando nessun

nodo è disposto a collaborare. Come dicevamo prima però, questa situazione è inaccettabile perché non permette il *routing multi-hops* e decreta l'eclissi della rete *Ad-hoc*.

Presentato lo scenario sappiamo che la sfida consiste quindi nel riuscire a proporre un metodo che incentivi i dispositivi a comportarsi in modo propizio alla comunità, una tecnica facente sì che i device classifichino il loro equilibrio in un range dal quale derivi il bene comune. I primi due approcci, che la ricerca ha tentato di adottare, si sono basati rispettivamente su meccanismi di **credit exchange** e **reputation**; il primo consiste nell'inserire all'interno del network una *moneta virtuale*, con la quale remunerare l'inoltro di informazione per poi essere spesa in richieste da effettuare (come in [21]), mentre il secondo nella messa in atto di un indice in grado di soppesare il contributo dei nodi in base al traffico passante per essi (un esempio in [18]), questo permette di imporre un *rank* con il quale schedulare le pretese di ciascun nodo.

Ambedue le idee risultano essere valide sulla carta e anche nella pratica, molte simulazioni di sistemi concepiti in questa maniera ottengono ottimi risultati infatti, tuttavia non risultano essere esenti da problematiche. In prima battuta entrambe soffrono della loro stessa definizione formale, appare chiaro infatti che vanno a ricadere nell'insieme delle euristiche e pertanto non forniscono un perfetto procedimento algoritmico in grado di prevedere, senza ambiguità, il futuro di rete. In secondo luogo, dividendo gli approcci, evidenziamo: nel **credit exchange** la tendenza dei nodi ad una sfrenata rincorsa verso l'accumulo di un tesoretto futuro, ciò a volte risulta sia inutile che dispendioso (rispettivamente perché si affastella talmente tanto credito da superare di gran lunga quello speso in richieste e perché si utilizzano risorse energetiche che non vengono ripagate, ovviamente, in aumenti di batteria), nel **reputation** un'inclinazione, dei dispositivi di sistema, all'imbroglio per quel che concerne la stima che gli

altri dovrebbero nutrire verso loro (è il caso del *Sybil Attack* che consiste nella introduzione di identità fasulle, da parte di un nodo, alle quali inoltrare pacchetti finti per innalzare il proprio grado di considerazione in rete). Proprio per quello che abbiamo appena esposto in questo paragrafo, possiamo affermare che queste due tecniche risultano essere limitate per suggerire un comportamento ai partecipanti, serve quindi un modello diverso per impugnare la situazione.

Ricorriamo ancora una volta al *Dilemma del Prigioniero*, la versatilità di questo problema riesce a renderlo utile per più analisi possibili; in questo caso scegliamo una sua variante: il *Dilemma del Prigioniero Ripetuto*. Nel *PD* ci troviamo a dover valutare quale sarà la scelta dei due partecipanti e l'unica indicazione che abbiamo è l'intrinseca avversione al rischio dell'essere umano, andremo quindi a cercare un equilibrio sulla base di un profilo strategico ristretto. Nella variante appena citata invece possiamo eseguire stime differenti, ripetendo il gioco un numero n di volte è ovvia la possibilità, che dal secondo turno in poi, sarà possibile basare la scelta futura di un giocatore su quella precedente del suo oppositore. Una delle possibili tecniche utilizzabili, nonché la più conosciuta, è la *Trigger Strategy*; essa consiste nella piena cooperazione da parte di uno dei due partecipanti fino a che l'altro non innesta il grilletto (*Trigger*), in questo caso, il "tradito", si sente ora libero di punire il traditore attraverso diverse strategie, le due più conosciute sono:

Grim-Trigger Consiste nel cooperare dal primo turno e smettere di farlo appena l'avversario tradisce. Qui non ci sono scusanti per l'oppositore, se questi tradisce, anche una sola volta, non verrà mai perdonato (chi adotta la *Grim-Trigger* lo tradirà per tutta la ripetizione del gioco);

Tit-For-Tat Questa strategia può essere definita come la "*occhio per occhio, dente per dente*". Il giocatore A decide la sua giocata al tempo *zero* e, dal secondo turno in poi, rende "*pan per focaccia*" al giocatore B.

L'utilizzo di una *Trigger Strategy* e l'ampliamento del *Dilemma del Prigioniero* a n partecipanti, rendono quest'ultimo un buon modello di base per la descrizione comportamentale (partecipazione al routing) dei vari nodi [5].

Il punto di arrivo è riuscire a definire una funzione di utilità che porti ad un *payoff* π attraverso il quale trovare un equilibrio che permetta alla *MANet* di "reggersi in piedi". Considerando s un mittente, d un destinatario, r un *path*, $f_1, f_2, f_3, ..., f_l$ i nodi intermedi (scelti per creare il *path* più corto), c il costo unitario di invio di un pacchetto e dividendo il tempo in *slot*, procediamo con ordine definendo:

- $p_i(t) \in [0,1]$

  Il **grado di cooperazione** di un determinato nodo, indicando con 1 il livello massimo;

- $T_s(r)$

  Il **Throughput** che una sorgente s vorrebbe avere sul *path* r;

- $\tau(r,t) = T_s(r) * \prod_{1 \leq k \leq l} p_{f_k}(t)$

  Il **Throughput** reale che ha la sorgente s sul *path* r al tempo t;

- $\hat{\tau}(r,t) = \dfrac{\tau(r,t)}{T_s(r)} = \prod_{1 \leq k \leq l} p_{f_k}(t)$

  Il **Throughput** reale normalizzato;

- $\xi_s(r,t) = u_s\left(\tau(r,t)\right)$

  Il **ricavo** per la sorgente s nello sfruttare il *path* r al tempo t;

- $\eta_{f_j}(r,t) = T_s(r) * c * \hat{\tau}_j(r,t) \quad con\, \hat{\tau}_j(r,t) = \prod_{1 \leq k \leq j} p_{f_k}(t)$

  Il **costo** dell'inoltro di un pacchetto per un nodo intermedio j.

Considerando ancora $S_i(t)$ tutti i *path* dove i è la sorgente e $F_i(t)$ quelli in cui lo stesso risulta intermedio; siamo in grado, a questo punto, di definire il *payoff* di ogni singolo nodo presente in rete come segue:

$$\pi_i(t) = \sum_{q \in S_i(t)} \xi_i(q,t) - \sum_{r \in F_i(t)} \eta_i(r,t)$$

E anche il metodo di calcolo del *grado di cooperazione*:

$$p_i(t) = \sigma_i([\hat{\tau}(r,(t-1))]) \quad con\, r \in S_i(t-1)$$

Sono possibili ora le definizioni delle strategie:

Collaborare Qualsiasi cosa succeda si sceglie $p_i(t) = 1$;

Non-Collaborare Qualsiasi cosa succeda si sceglie $p_i(t) = 0$;

Tit-For-Tat Si ribatte all'avversario con la sua stessa scelta;

Anti-Tit-For-Tat Si ribatte all'avversario con la scelta complementare (ad 1).

Queste andranno ad essere scelte tramite il metamodello definito da *Jean-Pierre Hubaux* e *Mark Felegyhazi*, costituito da una visuale dei nodi a macchine automatiche che hanno il loro input ed output definito dal *grafo delle dipendenze*. Questo non è altro che un grafo orientato nel quale si genera un arco da A verso B se e solo se esiste un *path* dove A è l'origine e B un qualsiasi dispositivo intermedio (figura 4.7.1). Ciò che ci interessa di più in questa struttura sono i cicli, la presenza di questi infatti indica che il *grado di cooperazione* futuro di un determinato nodo dipende anche dalle scelte compiute dallo stesso in passato. Distinguiamo, per un nodo i, due tipologie di cicli: se tutti i nodi presenti nel ciclo di i, tranne egli stesso, giocano una strategia che dipende dalle scelte degli altri allora siamo dinanzi ad un *reactive dependency loop* altrimenti si parla di un *non-reactive dependency loop*.

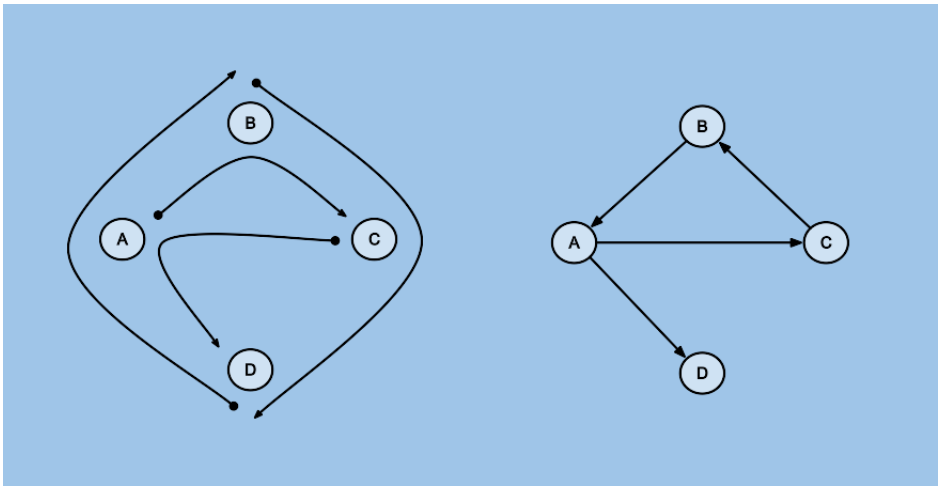


Figura 4.7.1: Paths di una MANet con relativo Grafo di Dipendenza

Dalla figura appena sopra e dal metamodello, indicando con $\Phi(r)$ tutti i nodi intermedi in *path* r, con $scr(r)$ il sorgente dello stesso percorso e con Φ l'insieme di tutti i device che sono almeno una volta degli inoltratori, abbiamo due fondamentali teoremi che portano a altrettanti possibili equilibri di Nash [5].

Teorema 1. *Se $i \in \Phi$ e non ha cicli di dipendenza, o ne ha solo di reattivi, allora la sua migliore risposta è giocare la strategia* ***Non-Collaborare****.*

Teorema 2. *Se $i \in \Phi$ la sua migliore risposta è* ***Collaborare*** *se*:

- $\forall r \in F_i \mid \exists L_{i,src(r)}$ *contenente l'arco* $(i, src(r))$;
- $\forall r \in F_i \mid \dfrac{u_i^{'}(T_i) * T_i * \omega^{|L_{i,src(r)}|-1}}{\mid F_i \mid} > T_{src(r)} * c$ *con* ω *indice di discount del payoff (ad ogni time slot il payoff viene diminuito secondo tale fattore, si utilizza nel calcolo del payoff globale per un nodo)*;
  $\overline{\pi}_i = \sum_{t=0}^{\infty} [\pi_i(t) * \omega^t] \quad con\, 0 < \omega < 1$;
- *Tutti gli altri nodi di* Φ *giocano* ***Tit-For-Tat***;

Nel caso del primo teorema avremo equilibrio se tutti i giocatori scelgono **Non-Collaborare**, nel secondo invece la stabilità arriva con tutti i nodi in Φ che propendono per il **Tit-For-Tat**.

Con questo modello, pur avendo trovato una seconda alternativa di equilibrio, rimane il problema del bilanciamento della rete verso la tattica della non cooperazione se un nodo decide di deviare verso il livello 0 (in upload).

Questa circostanza, unita ai lati negativi che hanno nella loro natura il **credit exchange** ed il **reputation**, ci costringe ad asserire che l'unico metodo per ottenere un equilibrio in grado di non far collassare la rete è l'adozione di un meccanismo di punizione. Questo potrebbe essere descritto come un gioco nel quale, il comportamento dei partecipanti, fa tendere ad una situazione di collasso, ma le scelte che portano i giocatori a questa condizione li puniscono per n turni successivi, così da scoraggiare quella opzione (es. in tabella 4.2). È opportuno che i partecipanti siano informati della presenza delle punizioni prima che attuino una condotta egoistica; altrimenti se venissero puniti, appena scontata la sanzione, ricadrebbero ancora una volta nella stessa contingenza.

		B	
		C	NC
A	C	(-1, -1)	(-1, 0) [Punizione **B** = **-2**]
	NC	(0, -1) [Punizione **A** = **-2**]	(0, 0) [Punizioni **Entrambi** = **-3**]

Tabella 4.2: Dilemma del Prigioniero con Punizioni

Una proposta che appare sensata, per applicare questa tecnica, è l'aggiunta di campi al pacchetto *Hello* che abbiano una doppia funzione:

- Comunicazione della scelta effettuata ai nodi vicini;
- Comunicazione dell'eventuale punizione di un nodo verso tutti i vicini che subiscono un "danno" dal suo comportamento.

4.7.2 Diminuzione dei Gateway

Il criterio con il quale abbiamo clusterizzato il network, per ovvi motivi, non può tener conto in maniera preventiva dei pacchetti circolanti in rete. Questo fa sì che non ci sia alcuna logica, tranne quella algoritmica, per l'elezione di Gateway; semplicemente questi vengono scelti in base alla posizione che occupano nello spazio. Ciò però ci pone davanti ad un problema, sapendo che una delle peculiarità che si vogliono ottenere riguarda il risparmio di batteria e sapendo altresì che l'appartenenza ad n cluster moltiplica per n lo sforzo energetico compiuto dal nodo (nella discovery); allora risulta elementare voler diminuire la cardinalità dell'insieme intersezione tra due gruppi di nodi. Approfittiamo per ricordare che un *Gateway* inoltra le richieste a tutti i suoi *Host Cluster* e che un *l-Gateway* ha **un solo** *Host Cluster*.

Questa operazione però può essere eseguita solo dopo la generazione di una buona quantità di traffico, la quale ci darà una base con cui potremo (o meglio potrà il network) effettuare una determinata scelta (es. rimanere con un solo *Gateway*). Come stimare quale è il *Best Gateway*, per una determinata unione di nodi, è però un problema che ha bisogno di una buon modellazione, non possiamo infatti basarci semplicemente sui dati ottenuti dai *Clusterhead* attraverso l'operazione di *retrieve* di informazione. Ciò che vogliamo ottenere è un sistema che calcoli il valore aggiunto che ogni Gateway riesce a portare all'interno del suo cluster, per poi mantenere solo il migliore, come membro, e rilasciare tutti gli altri (declassarli ad *l-Gateway*). Una probabile ottimizzazione potrebbe essere quella proposta in figura 4.7.2, nella quale si introduce una situazione creata dall'operazione di clusterizzazione (α) per poi presentare la successiva topologia (β) dovuta all'eliminazione di alcune appartenenze multiple di dispositivi ad insiemi.

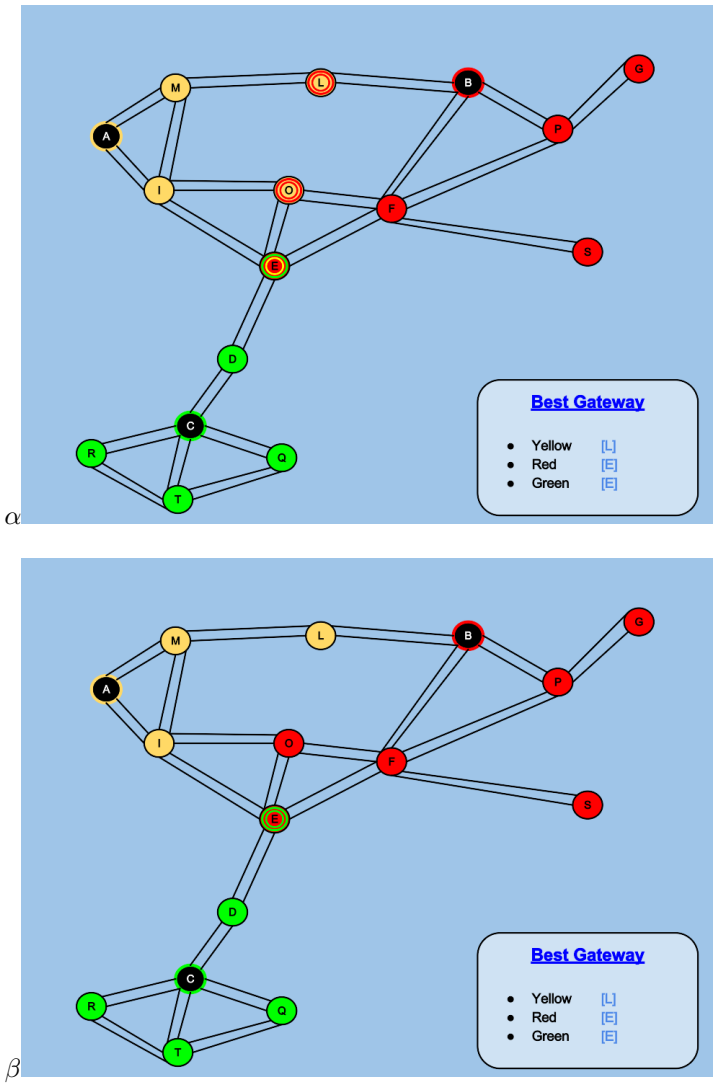


Figura 4.7.2: Eliminazione dei Gateway meno necessari

Nel caso presentato è ovvio il risparmio energetico ottenibile; infatti:

- il nodo L non deve più inoltrare verso B;
- il nodo O non deve più inoltrare verso A;
- il nodo E non deve più inoltrare verso A.

Immaginando di eseguire questa operazione per tutta l'area geografica in cui la

rete si estende, il numero di nodi ed il relativo risparmio energetico pro capite non sarebbe sicuramente esiguo! Ma come ottenere questo sistema?

Abbiamo parlato, nel terzo capitolo di questo elaborato, di giochi cooperativi ed abbiamo accennato agli indici che permettono di risolverli, cioè assegnare ad ogni partecipante il meritato *payoff*; aguzzando l'ingegno potremo dar forma di un *Cooperative Game* alla nostra circostanza, in modo da poter così utilizzare questi indicatori per la selezione del *Best Gateway*. Il "trucco", in questo caso, consiste nel creare tanti giochi quanti sono i cluster presenti ed impostare, come relativi partecipanti di questi, i soli nodi *Gateway* appartenenti all'unione caratterizzante il *Game*. Procediamo così:

- $N = \left\{N^A, N^B, N^C, ..., N^\Omega\right\}$

  Indichiamo così l'insieme di tutti i cluster;
- $N^\omega = \{n_1^\omega, n_2^\omega, n_3^\omega, ..., n_n^\omega\}$

  Così l'insieme di tutti i nodi membri del cluster con *Clusterhead* ω;
- $N_g^\omega \subset N^\omega$

  L'insieme dei *Gateway* per il cluster capitanato da ω;
- $\Gamma = \left\{G^A, G^B, G^C, ..., G^\Omega\right\}$

  L'insieme dei giochi cooperativi nella rete;
- $G^\omega = \left(N_g^\omega, v^\omega\right)$

  Il gioco cooperativo legato al cluster con *Clusterhead* ω per il quale si definisce una specifica funzione caratteristica v^ω.

Vengono proposti $\mid N \mid$ giochi, rispettivamente uno per ogni cluster, con rispettive $\mid N \mid$ funzioni v^ω per enfatizzare la differenza di utilità globale nella partecipazione di un *Gateway* a reti diverse; infatti un generico $n_i^\omega \in N_g^\omega$ potrebbe anche appartenere ad un $N_g^{\omega'}$, in questo caso il suo apporto va valutato rispettivamente con v^ω nel primo caso e con $v^{\omega'}$ nel secondo. Rimane quindi da capire come calcolare la partecipazione di un *Gateway* relativamente alla sua

subnet, una soluzione a questo problema può essere data dallo *Shapley Value*.

Il valore appena citato, dovuto al matematico statunitense *Lloyd Stowell Shapley*, ci permette di analizzare l'utilità marginale aggiunta da un giocatore ad ogni coalizione; nel nostro caso "l'aumento della possibilità di *discovery* di dispositivi e di numero di *path* che un device dona alla subnet a cui partecipa". Questo indice non è altro che un vettore indicante il *payoff* di ogni partecipante, viene così calcolato (considerando $P(\pi, i)$ la coalizione di giocatori che il *player* i segue, nella permutazione scelta):

$$\phi(v) = [\phi_1(v), \phi_2(v), \phi_3(v), ..., \phi_n(v)]$$

$$\phi_i(v) = \frac{1}{\mid N \mid} \sum_{\pi} \left[v\left(P(\pi, i) \bigcup \{i\} \right) - v\left(P(\pi, i) \right) \right]$$

Risulta però più comodo, ai fini della computazione, utilizzare la seguente formula abbreviata:

$$\phi_i(v) = \sum_{S \subseteq N, i \in S} \frac{(\mid S \mid -1)!\,(\mid N \mid - \mid S \mid)!}{\mid N \mid!} [v(S) - v(S \setminus \{i\})]$$

Ai nostri fini calcoleremo quindi lo *Shapley Value* di qualsiasi giocatore *Gateway* per ogni cluster, selezioneremo il migliore come *Best Gateway* e rilasceremo i restanti; ovviamente la procedura di proscioglimento di un nodo va subito comunicata così da non far verificare che uno o più device vengano espulsi da tutti i loro *host cluster*.

Cruciale è la scelta della funzione caratteristica per ogni agglomerato di nodi, purtroppo non avendo simulato il modello non si hanno dati che permettono la sua stesura; dobbiamo limitarci ad esclamare che questa, considerato il nodo da valutare, dipende:

- Dal numero di *path* che fornisce;
- Dalla circolazione a buon fine di **RREQ** e **RREP**;
- Dai **RRER** segnalati;
- Dal traffico che sostiene;
- Da parametri del device stesso.

In ogni caso, come asserito in precedenza, un'ottimizzazione di questo genere costituisce un risparmio energetico non indifferente, aprendo a scenari di riconfigurazione della rete che non sarebbero possibili in altre vie (ricordiamo che uno dei criteri input per l'integrale di *Choquet* è il livello di batteria rimanete).

Capitolo 5

Conclusioni

In questo elaborato ci siamo focalizzati nel descrivere un modello di gestione di *Reti Mobili Ad-hoc* molto diverso dal solito. Abbiamo utilizzato la *Teoria delle Decisioni* per approntare un meccanismo di divisione gerarchico-topologica dei device e, successivamente, descritto due metodi per ottimizzare la nostra risultante.

Le migliorie che crediamo possano derivare dall'utilizzo delle nostre normative sono principalmente due. Avendo clusterizzato la rete nel modo descritto, possiamo asserire, in primo luogo, di aver strutturato una più acuta scelta dei capigruppo (grazie allo *score* presentato); questo si traduce in una migliore formazione dell'agglomerato dei nodi ed in un perfezionamento della gestione delle richieste degli stessi. In secondo luogo siamo convinti che nel nostro caso, rispetto ad altri protocolli *cluster-based*, ci sia una preferibile gestione del risparmio energetico; ciò è attribuibile al ridotto numero di cluster, dato il diametro di zona pari a cinque anziché tre (come solitamente avviene), ed alla riduzione della cardinalità degli insiemi intersezione di queste sub-reti.

Inoltre, per la natura conferita al sistema, siamo in grado di stimare, at-

traverso le nostre competenze, che strutture così ideate possano dare il loro massimo contributo se utilizzate in reti ampie e composte da device non estremamente dinamici. Queste restrizioni non sono da considerarsi un handicap, le tipologie di network che possono configurarsi, tramite il paradigma *MANet*, sono molto eterogenee tra loro e quindi è naturale, nonché ragionevole, plasmare regole che siano calzanti per determinate condizioni e meno per altre.

Concludiamo dicendo che, tipicamente, la realizzazione di un protocollo per *Mobile Ad-hoc Networks* passa attraverso due lunghe ed ardue operazioni:

- Modellazione;
- Simulazione.

Nella tesi abbiamo affrontato il primo punto. Sarebbe ora ottimale, ai fini del lavoro svolto, predisporre dei test che siano in grado di valutare la reale bontà del prototipo descritto, così da riuscire a stimare valori-limite che indichino in quali circostanze sia congeniale l'utilizzo di tale protocollo; tutto ciò attenendosi a quelli che sono considerabili indici di giudizio significativi in una MANet, come ad esempio il *goodput* o la *latenza* nel calcolo di *path*.

Appendice A

Formazione di Cluster in MANet's

Riuscire a formare *cluster* in maniera adeguata, quando si parla di reti mobile *Ad-hoc*, non è un problema banale da affrontare. In queste tipologie di sistemi, come già ampiamente esposto, i nodi sono liberi di muoversi in qualsiasi direzione e soprattutto non sono controllati da nessuna autorità centrale; analizziamo ancora una volta, ora sotto il punto di vista della *clusterizzazione*, queste due caratteristiche delle *MANet's*.

Iniziamo con il chiarire il significato della parola *cluster*: "*A cluster is a group of similar things or people positioned or occurring closely together*" [The Oxford Dictionaries]; il concetto espresso dalla frase è estremamente semplice, le complessità infatti si celano dietro la realizzazione di questa operazione (intesa in via automatizzata). Come rendere infatti, una macchina, in grado di scernere se due attori, in un determinato schema, possono considerarsi simili? Solitamente, per eseguire questa ardua opera, non si tenta di studiare un algoritmo originale, ma piuttosto si concentrano i propri sforzi su particolari funzioni da eseguire

attraverso routine ben prestabilite. Queste funzioni sono dette di *similarità* ed hanno il compito di riuscire ad attribuire, ad ogni coppia di atomi[1] controllati, un punteggio che sia in grado di esprimere quanto la diade sia composta da soggetti simili. Reiterando poi la funzione, con apposite procedure (algoritmi ben definiti), sarà possibile analizzare tutto l'insieme universo, così da riuscire a formare i gruppi secondo i criteri voluti.

Le *Mobile Ad-hoc Networks* però stravolgono questo concetto basato sulla similarità. Nelle reti trattate in questa tesi, le somiglianze tra nodi, par far sì che essi siano raggruppabili, non possono altro che essere basate sulla posizione geografica degli stessi. Questa appunto risulta essere l'unica metodologia attuabile, visto che non si avrebbe nessuna utilità a raggruppare macchine con caratteristiche prestazionali simili a distanza elevata tra loro; in questa eventualità, pur appartenendo allo stesso gruppo, non avrebbero infatti opportunità di scambiarsi messaggi tra loro, facendo risultare vana la divisione. Inoltre, per tornare al discorso iniziale, si deve tenere conto di due punti: la mancanza di una *base station*, che rende necessario un sistema di coordinamento decentralizzato, e la grande libertà di mobilità dei nodi, che costringe a dover tener conto di una buona procedura di riconfigurazione. Tutto questo fa sì che, in ambito *Manet's*, si debba riflettere, piuttosto che sulle similarità dei device nel network, sulle procedure e su come attuarle, nonché su quali criteri puntare [3].

[1] **Atomo (Cluster)**: Con la parola atomo, nei contesti di clusterizzazione, si intende proprio un cluster. È necessario sottolineare che un elemento singolo sta costituendo un cluster a sé stante, per cui confrontare un soggetto con un gruppo di n elementi ben omogenei significa comunque considerare due atomi.

Si riescono a contare fino a sei tipologie di metodo [1]:

- Identifier-based clustering;
- Connectivity-based clustering;
- Mobility-aware clustering;
- Low cost of maintenance clustering;
- Power-aware clustering;
- Combined-weight based clustering.

Brevemente, per ognuna di queste, andiamo a descrivere il funzionamento base.

Identifier-based clustering

In questa metodologia di clustering, ad ogni elemento della rete, viene assegnato un identificatore utilizzato per individuare univocamente ogni device. Ogni nodo della rete è obbligato, in maniera periodica, ad inviare in broadcast il proprio *id*; questo fa si che ogni nodo conosca il proprio codice e quello di tutti i suoi vicini. A questo punto, con regole molto semplici (data a scarsa informazione circolante) basate sull'*id*, si procede alla elezione di un *clusterhead* e, subito dopo, si selezionano come membri di quel gruppo tutti i nodi raggiungibili dalle onde radio del membro centrale. Un esempio di *Identifier-based clustering* è il *Lower Id Cluster Algorithm* che seleziona il *clusterhead*, in un gruppo di nodi, scegliendo quello con l'identificativo più basso.

Connectivity-based clustering

Nel Connectivity-based si cerca di valutare la capacità dei singoli nodi nel riuscire ad essere un punto nevralgico del sistema, questo in base alla conformazione topologica che la rete ha assunto. Si passa da algoritmi che eleggono a *clusterhead* i nodi che risultano essere quelli con il massimo numero di vicini (anche a più *hops*), a routine più laboriose che considerano anche eventuali circostanze di congestione; fanno ciò determinando, anche dinamicamente, un numero

massimo e un numero minimo di membri in cluster, questo rispettivamente per evitare i colli di bottiglia e gli sprechi di energia. Ovviamente possono esistere configurazioni di rete che non rendono possibile alcuna ottimizzazione (un nodo isolato può esclusivamente appartenere ad un cluster a cardinalità 1, dove egli stesso è *clusterhead*).

Mobility-aware clustering

In algoritmi di questo tipo, come ci suggerisce il loro nome, è di centrale importanza la conoscenza, all'interno della rete, di informazioni riguardanti la mobilità dei device. Grazie all'analisi di questi frequenti rapporti è possibile deputare un nodo a capo del suo futuro cluster di appartenenza. Questa operazione deve essere comunque eseguita considerando dati non banali, vista la grande libertà di mobilità; possibili criteri da utilizzare per realizzare un *mobility-aware* sono: la velocità di movimento di un nodo (quindi scelta del *clusterhead* come nodo più lento) o le distanze medie misurate nel tempo (consentono di determinare il diametro, in metri ed in *hops*, da considerare attorno ad un nodo portavoce).

Low cost of maintenance clustering

Solitamente, gli algoritmi di clusterizzazione, agiscono in maniera tale da impostare una formazione iniziale richiamando ricorsivamente le istruzioni che li compongono, questo fino a quando nessun nodo ha la possibilità di essere marcato come *clusterhead* (praticamente sino a che nessun device rimane nella condizione *initial*, cioè non ha subito perturbazioni di stato); in maniera periodica poi, queste procedure, tendono a ripeterete l'operazione ricorsiva per far sì che si riesca a mantenere la comunicazione. Negli algoritmi a mantenimento *low cost*, si scinde in maniera netta la procedura di formazione da quella di mantenimento; in pratica si cercano *escamotages*, come la mancata ricezione di pacchetti entro un determinato tempo t, per modificare gli stati dei device invece che

rilanciare intere procedure di formazione che verrebbero sfruttate solo in parte, sottoponendo la rete ad un sovraccarico inutile.

Power-aware clustering

Come risaputo, uno dei problemi che attanaglia più chi utilizza le reti mobili *Ad-hoc* è sicuramente la batteria dei dispositivi. Un dispositivo scarico equivale ad un dispositivo inesistente in rete, come ovvio pensare sono stati pensati algoritmi in grado di minimizzare questo impiccio. Nelle situazioni in cui si decide di fare ricorso a questi metodi, le risorse di corrente elettrica dei vari dispositivi devono essere costantemente controllate e comparate tra loro; fatto questo si procede ad attuare politiche di risparmio di energia, le quali possono variare da una assegnazione temporizzata di *clusterhead* ad un gruppo, alla limitazione di membri solo se in posizione ottimale rispetto al capo-gruppo (distanza e sovrapposizione di onde elettromagnetiche) o addirittura alla non considerazione dell'informazione che viaggia su archi (ovviamente virtuali perché wireless) che si estendono tra due vertici che sono entrambi device semplici del cluster.

Combined-weight based clustering

Come era possibile immaginare, l'ultima procedura esistente per partizionare una MANet non è altro che una fusione di tutte le precedenti. Ogni tecnica che abbiamo visionato ci permette di ottimizzare la clusterizzazione di rete secondo un ben determinato criterio; in questo procedimento si sceglie di combinare, in maniera pesata, le misure viste in precedenza (connettività, mobilità ed energia) cercando di creare raggruppamenti di nodi che riescono ad adattarsi alle situazioni più disparate (ciò viene permesso dalla scelta, appunto, del peso che sarà attribuito alle varie caratteristiche valutabili). In alcuni casi questi algoritmi hanno anche la peculiarità di essere adattivi, essi possono, sotto ordine o in maniera periodica, valutare le condizioni delle rete e decidere se un determinato

parametro fornito in input può essere modificato per il bene della comunicazione nel network.

Come notiamo, il numero di algoritmi di formazione cluster, considerando tutte le loro varianti, è potenzialmente infinito; la vastità di queste procedure è giustificata comunque dalla moltitudine di scenari in cui la tecnologia delle reti mobili *Ad-hoc* può essere applicata. Esistono infatti campi dove si prediligono modelli adatti a reti dinamiche, altri dove si favoriscono i range di comunicazione corti, altri ancora dove l'unica preoccupazione è data esclusivamente dalla potenza dei device ed anche situazioni dove si sceglie di non adottare alcun criterio legato a topologia o caratteristiche di nodi. La scelta va quindi effettuata a priori, cercando di immaginare quale potrà essere quella che "calzerà" di più alla rete in utilizzo.

Bibliografia

[1] Ratish Agarwal, Dr Motwani, et al. Survey of clustering algorithms for manet. *International Journal of Computational Science and Engineering*, 1(2):98–104, 2009.

[2] Arifa Azeez and K. G. Preetha. A novel approach for dynamic leader election and key based security in manet clusters for the secured data distribution. In *Proceedings of the Second International Conference on Computational Science, Engineering and Information Technology*, CCSEIT '12, pages 52–56, New York, NY, USA, 2012. ACM.

[3] Mohammad U Bokhari, Hatem SA Hamatta, and Shams Tabrez Siddigui. A review of clustering algorithms as applied in manets. *International Journal*, 2(11), 2012.

[4] Dimitris E Charilas and Athanasios D Panagopoulos. A survey on game theory applications in wireless networks. *Computer Networks*, 54(18):3421–3430, 2010.

[5] Mark Felegyhazi, J-P Hubaux, and Levente Buttyan. Nash equilibria of packet forwarding strategies in wireless ad hoc networks. *Mobile Computing, IEEE Transactions on*, 5(5):463–476, 2006.

[6] Nishu Garg and RP Mahapatra. Manet security issues. *International Journal of Computer Science and Network Security*, 9(8):241, 2009.

[7] Mario Gerla and Jack Tzu-Chieh Tsai. Multicluster, mobile, multimedia radio network. *Wireless networks*, 1(3):255–265, 1995.

[8] Itzhak Gilboa and David Schmeidler. Updating ambiguous beliefs. In *Proceedings of the 4th conference on Theoretical aspects of reasoning about knowledge*, pages 143–162. Morgan Kaufmann Publishers Inc., 1992.

[9] Itzhak Gilboa and David Schmeidler. Additive representations of non-additive measures and the choquet integral. *Annals of Operations Research*, 52(1):43–65, 1994.

[10] Zhu Han, Dusit Niyato, Walid Saad, Tamer Basar, and Are Hjorungnes. *Game theory in wireless and communication networks*. Cambridge University Press, 2012.

[11] A Hussein, Sufian Yousef, Samir Al-Khayatt, and Omar S Arabeyyat. An efficient weighted distributed clustering algorithm for mobile ad hoc networks. In *Computer Engineering and Systems (ICCES), 2010 International Conference on*, pages 221–228. IEEE, 2010.

[12] Robert E Kahn, Steven A Gronemeyer, Jerry Burchfiel, and Ronald C Kunzelman. Advances in packet ratio technology. *Proceedings of the IEEE*, 66(11):1468–1496, 1978.

[13] Young-Bae Ko and Nitin H Vaidya. Location-aided routing (lar) in mobile ad hoc networks. *Wireless Networks*, 6(4):307–321, 2000.

[14] D Kreps. *Microeconomia per manager/a cura di Gilli, MR*. Egea, 2005.

[15] J v Neumann. Zur theorie der gesellschaftsspiele. *Mathematische Annalen*, 100(1):295–320, 1928.

[16] P. Victer Paul. Mobile networks: Ip routing and manet routing algorithms, 2011.

[17] P. Victer Paul. Routing in mobile ad-hoc networks, 2011.

[18] M Tamer Refaei, Vivek Srivastava, Luiz DaSilva, and Mohamed Eltoweissy. A reputation-based mechanism for isolating selfish nodes in ad hoc networks. In *Mobile and Ubiquitous Systems: Networking and Services, 2005. MobiQuitous 2005. The Second Annual International Conference on*, pages 3–11. IEEE, 2005.

[19] Amit Kumar Saha and David B Johnson. Modeling mobility for vehicular ad-hoc networks. In *Proceedings of the 1st ACM international workshop on Vehicular ad hoc networks*, pages 91–92. ACM, 2004.

[20] Amit Shrivastava, Aravinth Raj Shanmogavel, Avinash Mistry, Nitin Chander, Prashanth Patlolla, and Vivek Yadlapalli. Overview of routing protocols in manet's and enhancements in reactive protocols. 2005.

[21] Sheng Zhong, Jiang Chen, and Yang Richard Yang. Sprite: A simple, cheat-proof, credit-based system for mobile ad-hoc networks. In *INFOCOM 2003. Twenty-Second Annual Joint Conference of the IEEE Computer and Communications. IEEE Societies*, volume 3, pages 1987–1997. IEEE, 2003.

Sitografia

[22] The MacTutor History of Mathematics archive. Nash biography. http://www-history.mcs.st-and.ac.uk/Biographies/Nash.html. [URL controllato il 16-gennaio-2014].

[23] Wikipedia. Manet — wikipedia, l'enciclopedia libera. http://it.wikipedia.org/wiki/MANET. [URL controllato il 23-ottobre-2013].

[24] Wikipedia. Prisoner's dilemma — wikipedia, the free encyclopedia. http://en.wikipedia.org/wiki/Prisoner's_dilemma. [URL controllato il 7-gennaio-2014].

[25] Wikipedia. Router — wikipedia, l'enciclopedia libera. http://it.wikipedia.org/wiki/Router. [URL controllato il 26-ottobre-2013].

A mente libera...

Finalmente, ma a malincuore, giunge a termine un percorso. Finisce oggi un pezzo della mia vita ed io mi ritrovo a tirare le somme di questi ultimi anni.

Ricordo benissimo l'inizio di questa avventura, nel primo viaggio intrapreso verso "la Rossa" il cuore batteva forte; talmente forte da dare quella sensazione mista a vertigine e mancanza d'ossigeno, che ti accorgi immediatamente essere paura. Fortunatamente non fu difficile contrastare questo duro sentimento. L'aiuto venne dai primi compagni di sorte, ragazzi ai quali voglio manifestare tutta la mia gratitudine per aver contribuito, a volte anche in ruvida maniera, a forgiare un fermo lato del mio carattere. Grazie **Alessandro**, Grazie **Luca**, Grazie **Marco**. *Per avermi accompagnato nel mio arduo percorso.*

Il primo capitolo di questa storia volgeva a termine con il conseguimento della bramata laurea triennale e, data la ricchezza di questo, mai avrei potuto immaginare che il successivo si sarebbe composto delle meraviglie nelle quali esso si è infine articolato. La seconda parte dell'avventura iniziava ufficialmente con una scelta, mi sovvengono ancora adesso i pensieri che arrovellavano la mia mente in quell'estate: «Terminare gli studi con il solo triennio o tentare la magistrale?» ed ancora «In caso proseguissi, continuare con Informatica o passare a Scienze di Internet?». La decisione arrivò; ed ora, guardandomi indietro, posso asserire, con assoluta certezza, che non avrei potuto scegliere strada migliore da

percorrere.

Passando al "lato oscuro" ho avuto l'onore di conoscere persone che mai avrei pensato esistessero. Ho trovato, nei ragazzi della mia **classe** (così mi piace chiamarla), un'inconsueta solidarietà ed un ancor più raro senso di fratellanza, caratteristiche che fai fatica a ritrovare in ambiente universitario. Desidero quindi ringraziare loro dal primo all'ultimo. *Per avermi motivato ed aiutato in mille maniere e circostanze.*

I corsi di laurea, come ben noto agli studenti, non si compongono solo ed esclusivamente di ragazzi affamati di sapere; accanto a questi ultimi si incastra chi, quello stesso sapere, lo elargisce. Di queste persone, molte hanno impresso in me qualcosa che va oltre le nozioni accademiche; ma una in particolare ha avuto un'accezione da educatore. Grazie a TE **Jan**. *Per avermi fatto capire quanto sia importante relazionarsi con qualcuno senza pretendere nulla in base al ruolo che si ricopre, cercando di guadagnarsi rispetto ed amicizia mandando avanti il proprio essere piuttosto che il proprio status.*

Il tempo passava mentre io, ogni giorno, sedevo in quelle aule ed in quei laboratori; passava inesorabile per me e per gli altri... ...mi ritrovai a dover cambiare casa. Sì, lo so che era già successo in passato, ma la squadra era sempre rimasta la stessa! Stavolta mi ritrovavo a dover cercare campo e compagni per continuare la partita. Il caso volle, che proprio in quel periodo, stessi stringendo un forte legame con due ragazzi della mia classe; ero sempre nella loro casa per un motivo o per l'altro. Un bel giorno i due, assieme alle rispettive metà, mi proposero di raggiungerli al Ranzani 13; non dimenticherò mai quanto mi vennero incontro per permettere che questo accadesse. Quei quattro li considero il mio **poker** fortunato. *Li ringrazio per avermi fatto sentire desiderato sotto il profilo dell'amicizia. A mio avviso, questa sensazione, è una fonte dalla quale attingere forza per far fronte alle sfide che, giornalmente, la vita propone.*

Ranzani 13 ovviamente non è solo un condominio, Ranzani 13 non si compone solo del poker citato poche parole fa, Ranzani 13 è un gruppo di persone più vasto, Ranzani 13 è un'unione di anime in un'unica essenza, Ranzani 13 è collaborazione, Ranzani 13 è il sogno di ogni studente universitario. Ma soprattutto Ranzani 13, assieme a tutte le persone che lo compongono, è **Casa** (e la "C" maiuscola non è un errore). *Ringrazio questa entità per avermi accolto, con le braccia talmente aperte tanto da avermi fatto sentire profumo di famiglia*; *siete fantastici dal primo all'ultimo.*

Solitamente, in pagine come queste, oltre a ringraziare ti ritrovi sempre a ricordare un momento particolare che, per qualche motivo, si è impresso in maniera indelebile nella tua mente. Di tante uscite, esami e discorsi, io ricordo soprattutto una serata. Strano! Ho come l'impressione di aver già scritto questa frase! Comunque, tornando a noi, la serata che stavo per menzionare è quella in cui, con Carlo e Francesca, andai a sorseggiare una birra al Pratello. Perché è questo l'avvenimento che mi sovviene per primo vi chiedete? Semplice! Proprio in quella circostanza, **Francesca** mi chiese se volevo essere suo ospite nella splendida Sardegna l'estate successiva. Come non accettare? *Grazie veramente Fra; non mi hai regalato soltanto la possibilità di divertirmi e vedere posti sensazionali, ma mi hai donato anche l'opportunità di conoscere persone meravigliose.*

Allacciarsi all'ultima proposizione è qualcosa che non posso tralasciare di fare. In quell'isola ho conosciuto **Debora**, una ragazza che definire speciale (anzi spesciale) è riduttivo. Di lei mi colpì immediatamente, oltre che la fine bellezza, la squisita personalità; ricordo nitidamente i suoi occhi brillare mentre parlava della sua passione principe, l'effetto che fece sulle mie emozioni fu debordante. *Desidero ringraziarti per aver deciso di tentare questa relazione, che purtroppo non sappiamo quanto potrà durare. E ancora ti ringrazio per ogni sorriso che*

sei in grado di donarmi; non t'immagini quanto ognuno di essi riesca a riempire il mio cuore di felicità e speranza. 🌵🌵🌵🌵

Tutto quello che ho scritto, la mia favola, non sarebbe mai potuta iniziare né tantomeno giungere a lieto fine senza la spinta dei miei **genitori**. Essi mi hanno sorretto in ogni modo, assecondando qualsiasi mia scelta; hanno riposto in me tutta la loro fiducia che, nei miei confronti, non ho mai sentito vacillare. Olga e Roberto sono due persone di cui andare fiero e non lo sto dicendo perché sono loro figlio, lo dico perché lo credo sul serio. *Grazie Mamma e grazie Papà per quanto mi siete stati vicini e quanto avete creduto in me, penso che non avrei mai potuto chiedere di meglio.*

Cosa c'è più importante della vita? La vita è quella cosa che regala le emozioni più belle. Ella ti dona gli amori, gli amici, i profumi, gli affetti, le gioie e mille altre cose stupende. È vero, a volte purtroppo ti sbatte in faccia sensazioni terribili; ti trovi a dover far fronte a circostanze che ti sembrano insuperabili e soprattutto ingiuste. Proprio in quel momento però, devi rimboccarti le maniche e tirare in fuori il petto. Devi camminare a testa alta, sforzarti di riuscire con tutto te stesso... ...e quando arriva il momento in cui senti che potrai solo essere schiacciato dal problema in questione, è lì che devi dare il meglio di te. È lì che devi far ricorso a tutto quello che hai dentro, usare fino all'ultima goccia di forza con estrema tenacia, pensando solo a lottare e non a vincere. Perché anche se la vittoria non dovesse arrivare, lottando, hai dimostrato quello che vali.

Grazie **Massimo**

"La differenza tra Informatica e Scienze di Internet è pari a quaranta"

Alessandro Di Teodoro