

# Università degli studi di Catania

Corso di Laurea Magistrale Informatica

P2P & Wireless networks A.A 2014-2015

Antonio Fischetti W82000021

## *Implementazione in ns2 del DTN (Delay Tolerant Network) nel protocollo DSR*

### Introduzione

Lo scopo del progetto è stato quello di implementare in ns2 del **DTN (Delay Tolerant Network)** modificando il protocollo DSR (Dynamic Source Routing), protocollo di routing utilizzato nelle Mobile ad Hoc Networks.

Il **Delay Tolerant Network** rappresenta un'architettura di rete di telecomunicazione utilizzata in reti caratterizzate da ritardi nella trasmissione. Questo utilizza un meccanismo di store message e forward message. Ovvero se durante la trasmissione di uno o più pacchetti da un nodo sorgente ad un nodo destinazione, qualcosa nella comunicazione fallisce, il nodo trasmittente memorizza il pacchetto per poi spedirlo successivamente.

Questo è l'obiettivo del progetto in questione, implementato attraverso una modifica al DSR. Nello specifico durante lo scambio di pacchetti tra i nodi di una rete wireless simulata attraverso ns2, alcuni di questi non riescono ad arrivare a destinazione. Per questo motivo, utilizzando una struttura dati (una coda) memorizzo i pacchetti per poi rispedirli successivamente.

## Implementazione

Per lo sviluppo del progetto è stato necessario modificare la classe DSRAgent (che si trova nella cartella dsr che è contenuta all'interno della cartella ns2-allinone2.35 , il cui download è disponibile in rete).

All'interno di questa classe ho per prima cosa implementato una struttura dati nella quale memorizzare i pacchetti che vengono persi. Ho optato per l'implementazione di una coda, in quanto questa è una struttura dati di tipo FIFO (First input First output), per cui i pacchetti vengono estratti e quindi successivamente re-inviati in ordine di arrivo nella coda.

**deque<Packet\*> pacchetti\_da\_reinviare;**

La coda implementata prende il nome "pacchetti\_da\_reinviare" e contiene puntatori ad oggetti di tipo Packet. Da notare che, durante la fase di studio del codice del DSR sviluppato in linguaggio c++, mi sono accorto che all'interno della classe DSRAgent sono stati implementati due oggetti di tipo pacchetto. Uno (che è quello che salvo all'interno della struttura dati che è di tipo **Packet\***, ed un altro che è di tipo **SRPacket**, il quale anch'esso, durante la fase di ritrasmissione, utilizzerò per incapsulare il pacchetto ed utilizzare il metodo **sendOutPacketWithRoute()** che prende come parametro un pacchetto di tipo **SRPacket**).

Una volta creata la struttura dati, ho individuato quale fosse il metodo della classe DSRAgent che si occupa della gestione dei pacchetti persi, ovvero i pacchetti che non arrivano a destinazione, ed ho trovato il metodo **xmitFailed()**.

```
//This is the callback function when a MAC transmission failes. Based on this chance, route-error message generated
void
DSRAgent::xmitFailed(Packet *pkt, const char* reason)
/* mark our route cache reflect the failure of the link between
srh[cur_addr] and srh[next_addr], and then create a route err
message to send to the originator of the pkt (srh[0])
p.pkt freed or handed off */
{
    ...
}
```

Com'è possibile notare dal codice e dal commento contenuto in esso **xmitFailed()** “**is a callback function when a Mac transmission failes**” è un metodo di callback che viene richiamato quando la trasmissione a livello MAC fallisce, ed un messaggio di route-error viene generato. Poiché l'obiettivo del progetto è quello di memorizzare i pacchetti dati che non sono stati inviati per un qualsiasi problema ho pensato di effettuare diversi controlli sui pacchetti che transitano da questo metodo di callback per poi memorizzarne quelli che non siano di errore.

I controlli che vengono effettuati, prima della memorizzazione del pacchetto all'interno della coda sono:

- 1) Verifica se il pacchetto è di route error oppure è un pacchetto dati
- 2) Verifica se il pacchetto è destinato al nodo in questione (quindi un controllo del **current\_address()** e del **next\_address()** due variabili interne dell'oggetto pacchetto.
- 3) Controllo se l'uid del pacchetto non sia 0.

```
if (tell_id == net_id || tell_id == MAC_id)
{
    //printf("no need to send the route error if it's for us (1) \n");
}
else
{
    if(cmh->uid() != 0 )
    {
        //Controllo che l'address corrente ed il successivo non siano gli stessi
        if(srh->get_next_addr() != srh->cur_addr())
        {
            //printf("UID_Packet_Failed=%i - Route=%lu \n",cmh->uid(),p.route.dump());
            //Effettuo una push del pacchetto (Copiandolo)
            pacchetti_da_reinviare.push_back(pkt->copy());
            return;
        }
    }
}
}
```

Dopo aver effettuato questi vari controlli, il pacchetto viene memorizzato all'interno della struttura dati (coda) effettuando una **push\_back()**. E' importante evidenziare che i controlli da me effettuati sui pacchetti che transitano in questo metodo di callback non sono gli unici. Nel codice infatti sono presenti altri controlli.

Dopo la fase di memorizzazione del pacchetto, ho implementato la parte di re-invio. Questa rispetta il protocollo **Delay Tolerant Network** ovvero, il rinvio dei pacchetti avviene dopo un tempo fissato. Ho sviluppato quindi un metodo che viene richiamato dopo un tot di secondi. Per lo sviluppo di questo metodo mi sono basato su un metodo preesistente, molto simile, il quale si occupa di svuotare il buffer per l'invio dei pacchetti. Anch'esso viene richiamato con cadenza temporale.

```
void
```

```
RecallMethod::expire(Event *)
```

```
{
```

```
    //Controllo che la Struttura dati coda non sia vuota
```

```
    while (pacchetti_da_reinviare.size() > 0)
```

```
    {
```

```
        //Creo un SRPacket effettuando un'estrazione in testa
```

```
        hdr_sr *srh = hdr_sr::access(pacchetti_da_reinviare.front());
```

```
        hdr_ip *iph = hdr_ip::access(pacchetti_da_reinviare.front());
```

```
        hdr_cmn *cmh = hdr_cmn::access(pacchetti_da_reinviare.front());
```

```
        SRPacket p(pacchetti_da_reinviare.front(), srh);
```

```
        printf("Size_Temp_Coda=%i - UID=%i \n", pacchetti_da_reinviare.size(), cmh->uid());
```

```
        // Richiamo il metodo sendOutPacketWithRoute (Poichè il pacchetto ha già la route) pass
```

```
        a_->sendOutPacketWithRoute(p, true);
```

```
        //Effettuo una Pop dalla Struttura dati Coda
```

```
        pacchetti_da_reinviare.pop_front();
```

```
    }
```

```
// Richiamo il metodo dopo un totale di secondi decisi
```

```
    resched(CHIAMA_RECALL);
```

```
}
```

Il metodo effettua un controllo della struttura dati, verificando che questa non sia vuota (iterazione del ciclo While) e procedendo successivamente con l'incapsulamento del pacchetto di tipo **Packet\*** all'interno dell'oggetto **SRPacket** (che contiene tra le tante informazioni del pacchetto, anche la route).

```
SRPacket p;
```

```
p.src = net_id;
```

```
p.pkt = allocpkt();
```

```
hdr_sr *srh = hdr_sr::access(p.pkt);
```

```
hdr_ip *iph = hdr_ip::access(p.pkt);
```

```
hdr_cmn *cmnh = hdr_cmn::access(p.pkt);
```

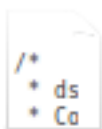
Viene richiamato il metodo **sendOutPacketWithRoute()** che si occupa di rinviare il pacchetto al nodo successivo. Infine viene effettuata un' estrazione nella struttura dati per eliminare il pacchetto che è stato inviato e viene richiamato il metodo **RecallMethod** dopo un tot di secondi stabilito dalla costante CHIAMA\_RECALL che si trova all'interno del file di intestazione DSRAgent.h

```
#define CHIAMA_RECALL 150 // Costante
```

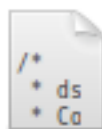
Per effettuare una simulazione è necessario inoltre lo sviluppo della parte in tcl. Questa si occupa della creazione dell'ambiente di lavoro.

In questa fase vengono creati i nodi, viene settata la griglia nel quale i nodi si muoveranno (casualmente nel mio caso), vengono impostate tutte le opzioni che servono per ricreare l'ambiente wireless, viene impostato il tipo di traffico (Cbr), la dimensione massima dei pacchetti (512 bytes), il rate di trasmissione e la durata della simulazione. Infine vengono impostate tutte le variabili che servono alla fine dell'esecuzione per visualizzare la simulazione nell'ambiente nam.

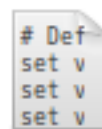
Il file in questione è **wireless.tcl**



dsragent.cc



dsragent.h



wireless.tcl

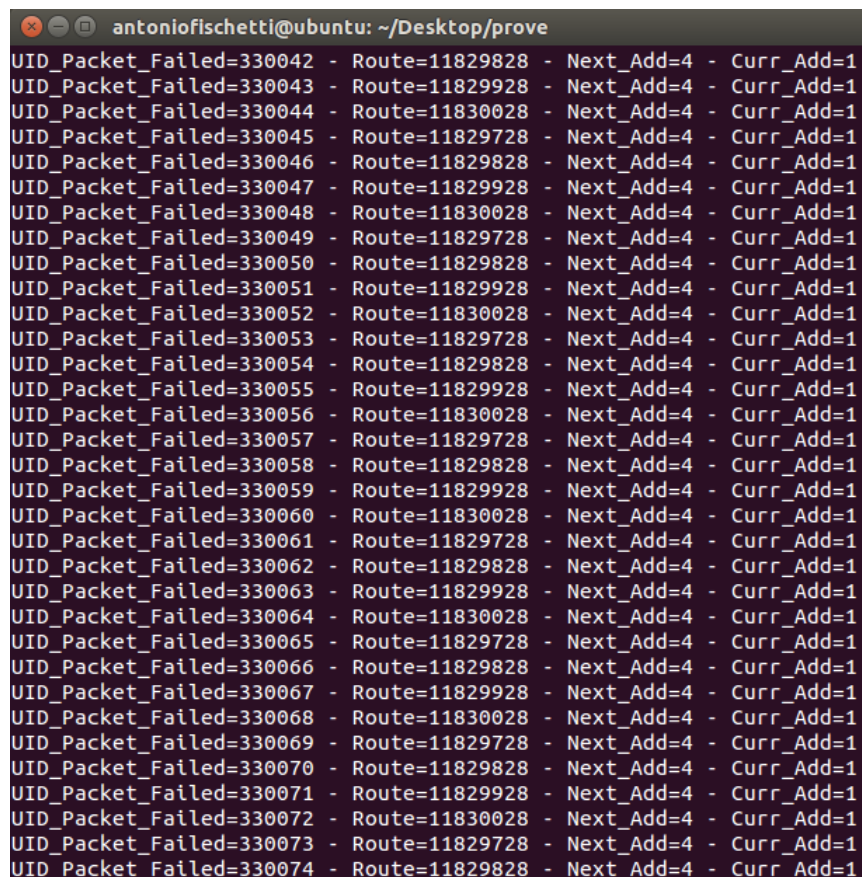
# Simulazione

La simulazione è stata effettuata tenendo conto dei parametri descritti precedenza nella parte relativa al file wireless.tcl

Lo scopo della simulazione , oltre che necessaria per capire l'efficienza e la correttezza dello sviluppo del codice, ha lo scopo di capire quale sia il numero di pacchetti che vengono salvati nella struttura dati e poi re-inviati al variare del tempo. Per capire quali siano i pacchetti che vengono salvati all'interno della coda e successivamente re-inviati, utilizzo dei printf() che stampano nella console, il numero di elementi contenuti nella coda ogni qual volta viene richiamato il metodo di re-invio dei pacchetti, stampando di questi sia l'identificativo UID, che la route. Quest'ultimo mi è stato utile in fase di sviluppo nel controllo dei pacchetti.

Per capire quindi l'andamento della simulazione ed estrapolarne le conclusioni ho effettuato diversi test, modificando il numero di secondi nel quale viene richiamato il metodo del re-invio dei pacchetti, ed esaminando la correlazione che vi è tra il numero totali di pacchetti che si trovano nella coda e che vengono re-inviati e quelli che dopo il re-invio sono stati correttamente recapitati.

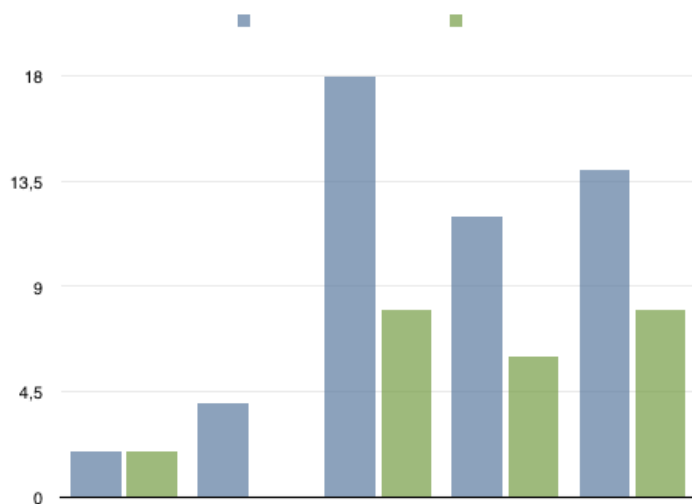
Per comprendere ciò ho preso in considerazione gli UID dei pacchetti.



```
antoniofischetti@ubuntu: ~/Desktop/prove
UID_Packet_Failed=330042 - Route=11829828 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330043 - Route=11829928 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330044 - Route=11830028 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330045 - Route=11829728 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330046 - Route=11829828 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330047 - Route=11829928 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330048 - Route=11830028 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330049 - Route=11829728 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330050 - Route=11829828 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330051 - Route=11829928 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330052 - Route=11830028 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330053 - Route=11829728 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330054 - Route=11829828 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330055 - Route=11829928 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330056 - Route=11830028 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330057 - Route=11829728 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330058 - Route=11829828 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330059 - Route=11829928 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330060 - Route=11830028 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330061 - Route=11829728 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330062 - Route=11829828 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330063 - Route=11829928 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330064 - Route=11830028 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330065 - Route=11829728 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330066 - Route=11829828 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330067 - Route=11829928 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330068 - Route=11830028 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330069 - Route=11829728 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330070 - Route=11829828 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330071 - Route=11829928 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330072 - Route=11830028 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330073 - Route=11829728 - Next_Add=4 - Curr_Add=1
UID_Packet_Failed=330074 - Route=11829828 - Next_Add=4 - Curr_Add=1
```

Esempio 1 : Re-invio dei pacchetti persi ogni 100 sec.

UID=701009	UID=701013	UID=701013	UID=701013	UID=701013	UID=701029
UID=701010	UID=701014	UID=701014	UID=701014	UID=701014	UID=701030
	UID=701017	UID=701017	UID=701021	UID=701029	UID=701055
	UID=701018	UID=701018	UID=701022	UID=701030	UID=701056
		UID=701021	UID=701029	UID=701045	UID=701063
		UID=701022	UID=701030	UID=701046	UID=701064
		UID=701025	UID=701037	UID=701055	UID=701071
		UID=701026	UID=701038	UID=701056	UID=701072
		UID=701029	UID=701045	UID=701059	UID=701075
		UID=701030	UID=701046	UID=701060	UID=701076
		UID=701033	UID=701051	UID=701063	UID=701079
		UID=701034	UID=701052	UID=701064	UID=701080
		UID=701037		UID=701067	UID=701083
		UID=701038		UID=701068	UID=701084
		UID=701041			UID=701087
		UID=701042			UID=701088
		UID=701045			UID=701091
		UID=701046			UID=701092
INVIATI	2	0	8	6	8
TOTALI	2	4	18	12	14





La tabella ed i grafici mostrano cosa accade quando viene impostato il tempo di ritrasmissione a 100sec. Vengono evidenziati in azzurro i pacchetti rinviati che non rimangono nella coda, mentre in nero quelli che rimangono.

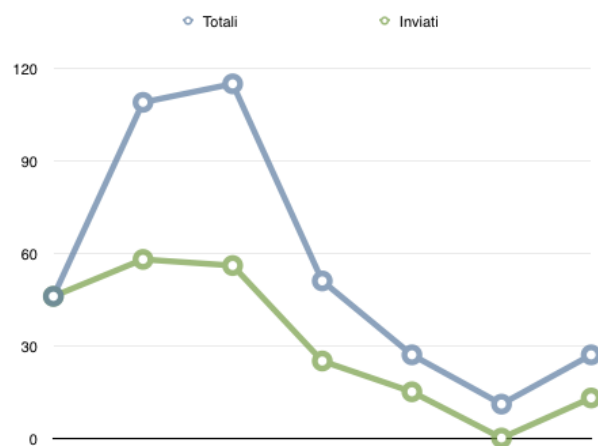
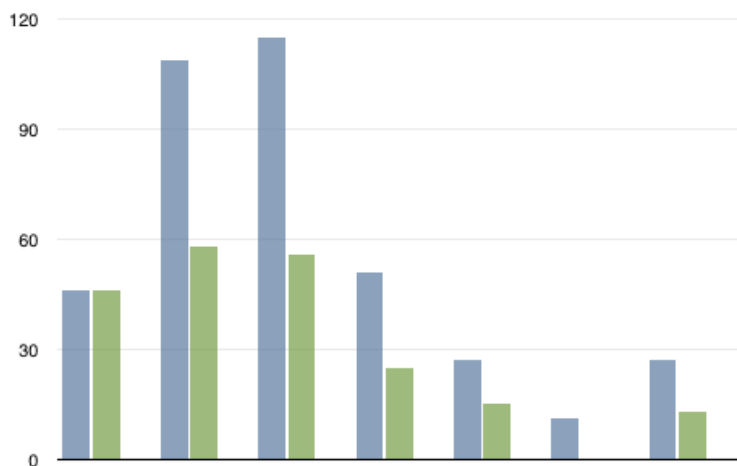
E' possibile notare come alcuni pacchetti vengono rinviati più volte prima di essere recapitati in modo corretto. Attraverso un grafico mostro la correlazione tra il numero totali di pacchetti all'interno della coda ed il numero di pacchetti rinviati che sono stati recapitati.

### Esempio 2 : Re-invio dei pacchetti persi ogni 150 sec.

1	UID=401967	UID=401790	UID=401790	UID=401790	UID=401908	UID=401908	UID=401908	UID=401658	41	UID=402014	UID=401927	UID=401915	UID=401915
2	UID=401968	UID=401908	UID=401908	UID=401908	UID=401658	UID=401658	UID=401704	UID=402082	42	UID=402015	UID=401928	UID=401916	UID=401916
3	UID=401971	UID=401658	UID=401658	UID=401658	UID=402078	UID=402078	UID=401658	UID=401667	43	UID=402016	UID=401931	UID=401919	UID=401919
4	UID=401972	UID=402020	UID=402020	UID=402020	UID=401667	UID=401667	UID=402082	UID=402083	44	UID=402017	UID=402026	UID=402085	UID=402085
5	UID=401975	UID=401840	UID=401840	UID=401840	UID=401674	UID=401674	UID=402078	UID=401678	45	UID=402018	UID=401932	UID=401920	UID=401920
6	UID=401976	UID=401667	UID=402078	UID=402078	UID=401678	UID=401678	UID=401754	UID=401893	46	UID=402019	UID=401935	UID=401923	UID=401923
7	UID=401979	UID=401892	UID=401667	UID=401667	UID=401685	UID=401685	UID=401667	UID=401770	47		UID=401936	UID=402025	UID=402025
8	UID=401980	UID=401673	UID=401892	UID=401892	UID=402021	UID=402021	UID=402083	UID=402080	48		UID=401939	UID=401924	UID=401924
9	UID=401982	UID=401674	UID=401673	UID=401673	UID=402080	UID=402080	UID=401674	UID=402024	49		UID=402027	UID=401927	UID=401927
10	UID=401983	UID=401678	UID=401674	UID=401674	UID=401693	UID=401693	UID=401759	UID=402081	50		UID=401940	UID=401928	UID=401928
11	UID=401984	UID=401681	UID=401678	UID=401678	UID=402081	UID=402081	UID=401678	UID=401923	51		UID=401943	UID=401931	UID=401931
12	UID=401985	UID=401685	UID=401681	UID=401681	UID=401704		UID=401893	UID=402025	52		UID=401944	UID=402086	
13	UID=401986	UID=402021	UID=402079	UID=402079	UID=402082		UID=401685		53		UID=401947	UID=402026	
14	UID=401987	UID=401686	UID=401685	UID=401685	UID=401754		UID=402021		54		UID=401948	UID=401932	
15	UID=401988	UID=401690	UID=402021	UID=402021	UID=402083		UID=401770		55		UID=401951	UID=401935	
16	UID=401989	UID=401693	UID=401686	UID=401686	UID=401759		UID=402080		56		UID=402028	UID=401936	
17	UID=401990	UID=401697	UID=401690	UID=401690	UID=401893		UID=401775		57		UID=401952	UID=401939	
18	UID=401991	UID=401844	UID=402080	UID=402080	UID=401770		UID=401693		58		UID=401955	UID=402027	
19	UID=401992	UID=401704	UID=401693	UID=401693	UID=401775		UID=402024		59		UID=401956	UID=402087	
20	UID=401993	UID=401705	UID=401697	UID=401697	UID=402024		UID=402081		60		UID=401959	UID=401940	
21	UID=401994	UID=402022	UID=401844	UID=401844	UID=401916		UID=401916		61		UID=402029	UID=401943	
22	UID=401995	UID=401754	UID=402081	UID=402081	UID=401919		UID=401919		62		UID=402030	UID=402088	
23	UID=401996	UID=401910	UID=401704	UID=401704	UID=401923		UID=401923		63		UID=402031	UID=402089	
24	UID=401997	UID=401758	UID=401705	UID=401705	UID=402025		UID=402025		64		UID=402032	UID=402090	
25	UID=401998	UID=401759	UID=402022	UID=402022	UID=401928		UID=401928		65		UID=402033	UID=402091	
26	UID=401999	UID=401763	UID=402082	UID=402082	UID=401931		UID=401931		66		UID=402034	UID=402092	
27	UID=402000	UID=402023	UID=401754	UID=401754					67		UID=402035	UID=402093	
28	UID=402001	UID=401893	UID=401910	UID=401910					68		UID=402036	UID=402094	
29	UID=402002	UID=401770	UID=401758	UID=401758					69		UID=402037	UID=402095	
30	UID=402003	UID=401771	UID=402083	UID=402083					70		UID=402038	UID=402096	
31	UID=402004	UID=401775	UID=401759	UID=401759					71		UID=402039	UID=402097	
32	UID=402005	UID=402024	UID=401763	UID=401763					72		UID=402040	UID=402098	
33	UID=402006	UID=401912	UID=402023	UID=402023					73		UID=402041	UID=402099	
34	UID=402007	UID=401915	UID=401893	UID=401893					74		UID=402042	UID=402100	
35	UID=402008	UID=401916	UID=401770	UID=401770					75		UID=402043	UID=402101	
36	UID=402009	UID=401919	UID=402084	UID=402084					76		UID=402044	UID=402102	
37	UID=402010	UID=401920	UID=401771	UID=401771					77		UID=402045	UID=402103	
38	UID=402011	UID=401923	UID=401775	UID=401775					78		UID=402046	UID=402104	
39	UID=402012	UID=402025	UID=402024	UID=402024					79		UID=402047	UID=402105	
40	UID=402013	UID=401924	UID=401912	UID=401912					80		UID=402048	UID=402106	



80	UID=402048	UID=402106	100	UID=402068	UID=402126					
81	UID=402049	UID=402107	101	UID=402069	UID=402127					
82	UID=402050	UID=402108	102	UID=402070	UID=402128					
83	UID=402051	UID=402109	103	UID=402071	UID=402129					
84	UID=402052	UID=402110	104	UID=402072	UID=402130					
85	UID=402053	UID=402111	105	UID=402073	UID=402131					
86	UID=402054	UID=402112	106	UID=402074	UID=402132					
87	UID=402055	UID=402113	107	UID=402075	UID=402133					
88	UID=402056	UID=402114	108	UID=402076	UID=402134					
89	UID=402057	UID=402115	109	UID=402077	UID=402135					
90	UID=402058	UID=402116	110		UID=402136					
91	UID=402059	UID=402117	111		UID=402137					
92	UID=402060	UID=402118	112		UID=402138					
93	UID=402061	UID=402119	113		UID=402139					
94	UID=402062	UID=402120	114		UID=402140					
95	UID=402063	UID=402121	115		UID=402141					
96	UID=402064	UID=402122								
97	UID=402065	UID=402123								
98	UID=402066	UID=402124	TOT	46	109	115	51	27	11	27
99	UID=402067	UID=402125	INVIATI	46	58	56	25	15	0	13



La tabella ed i grafici mostrano cosa accade quando viene impostato il tempo di ritrasmissione a 150sec. Nella tabella vengono evidenziati in blu i pacchetti rinviati che non rimangono nella coda. Anche in questa situazione è possibile notare come alcuni pacchetti vengono rinviati più volte prima di essere recapitati in modo corretto.

Attraverso il grafico mostro la correlazione tra il numero totali di pacchetti all'interno della coda ed il numero di pacchetti rinviati che sono stati recapitati. Da questo è possibile dedurre che ad ogni ritrasmissione, più del 50% dei pacchetti re-inviati viene recapitato al nodo destinatario.

# Conclusioni

