Corso di Reti per l'automazione industriale

Elaborato Omnet++

Andrea Calabretta 1000008923, Alessandro Mauro 1000009156

Relazione finale – Elaborato_G

1. Introduzione

L'obiettivo è quello di simulare una rete wireless ad hoc IEEE 802.11 in cui dei nodi mobili scambiano periodicamente dei messaggi.

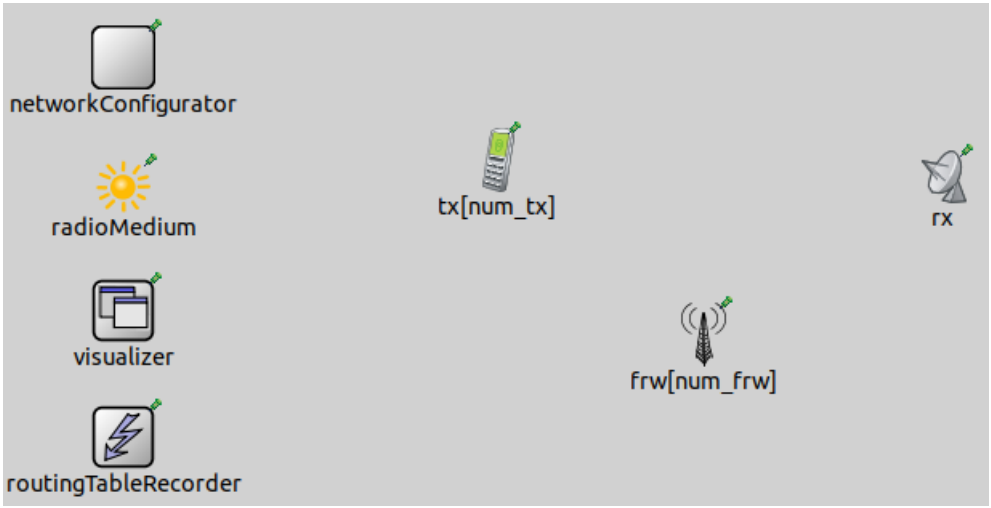


Figura 1 - Network.ned

In tutte le simulazioni sono presenti tre tipologie di nodi: **tx** è un array di nodi che inviano messaggi, **frw** è un array di nodi che si occupano di inoltrare i messaggi ricevuti mentre **rx** è un nodo che riceve i messaggi. Il submodule scelto per tutte e tre le tipologie è **adHocHost**.

Inoltre, sono stati utilizzati i seguenti submodules:

- **networkConfigurator** – InetworkConfigurator: si occupa di creare l'interfaccia di rete.
- **radioMedium** – IEEE802.11ScalarRadioMedium: simula il mezzo radio trasmissivo tramite cui avviene la comunicazione; simula inoltre attenuazioni ed eventuali rumori di canale.
- **routingTableRecorder** – RoutingTableRecorder: registra i cambiamenti nelle tabelle di routing.

È stato valutato l'end to end delay del nodo ricevente in diverse configurazioni di rete; in una prima fase è stata implementata una topologia di rete piuttosto semplice che presentava solo un nodo emettitore e un ricevitore. Successivamente, si è scelto di complicare lo scenario aumentando il numero di nodi e attivando la mobilità dei trasmettitori.

2. Scenario

Sono state implementate tre configurazioni di rete:

- Config. **Completa** -> 1 Nodo Ricevitore, 8 Forwarding Node, 11 End Node: ***Figura 2***
- Config. **InLineForwarder** -> 1 Nodo Ricevitore, 3 Forwarding Node, 1 End Node: ***Figura 3***
- Config. **Static Nodes** -> 1 Nodo Ricevitore, 2 Forwarding Node, 17 End Node: ***Figura 4***

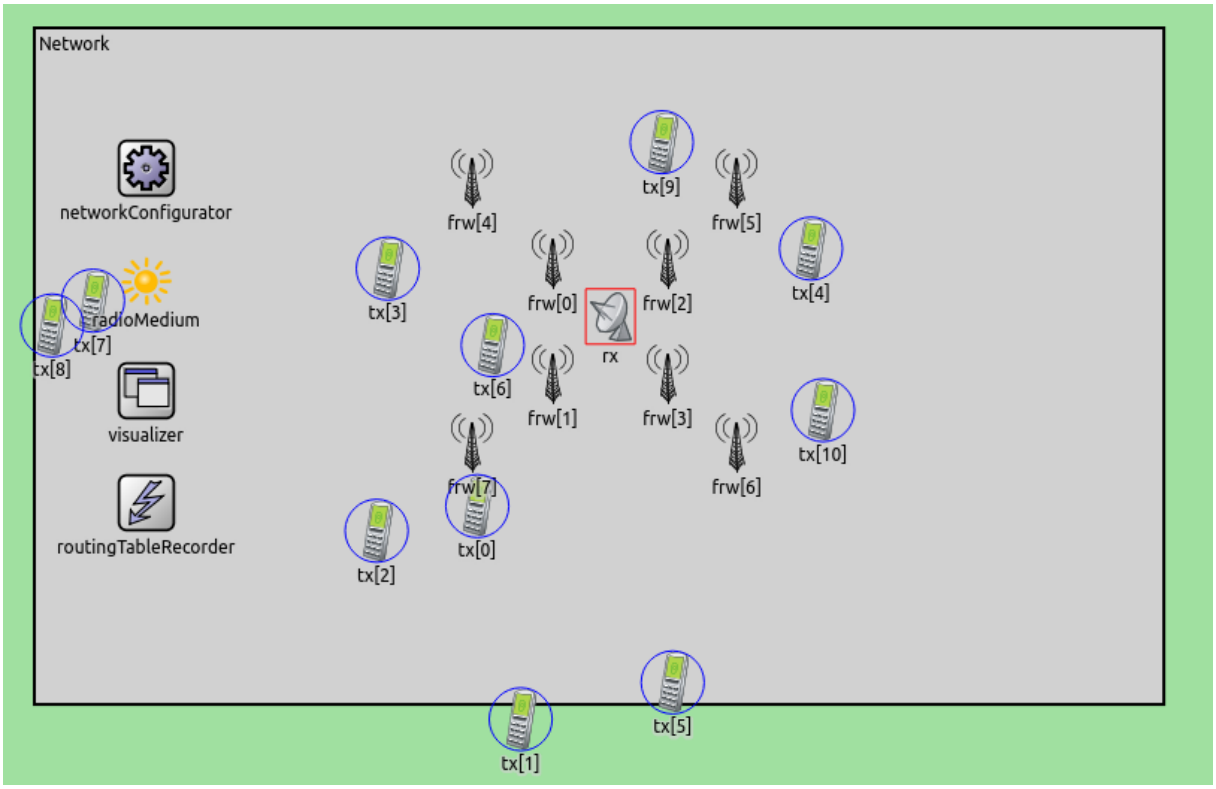


Figura 2 - Configurazione Completa

Nella prima configurazione di rete (Completa), la posizione degli End Node ***tx[*]*** è generata in maniera casuale utilizzando la tipologia di mobilità *"MassMobility"*; si tratta tuttavia di una generazione pseudocasuale quindi, mandando più volte in esecuzione la simulazione, ciascun nodo occuperà la stessa posizione della simulazione precedente.

Il Ricevitore ***rx*** e i Forwarding Node ***frw[*]*** sono statici e la tipologia di mobilità utilizzata è *"StationaryMobility"*.

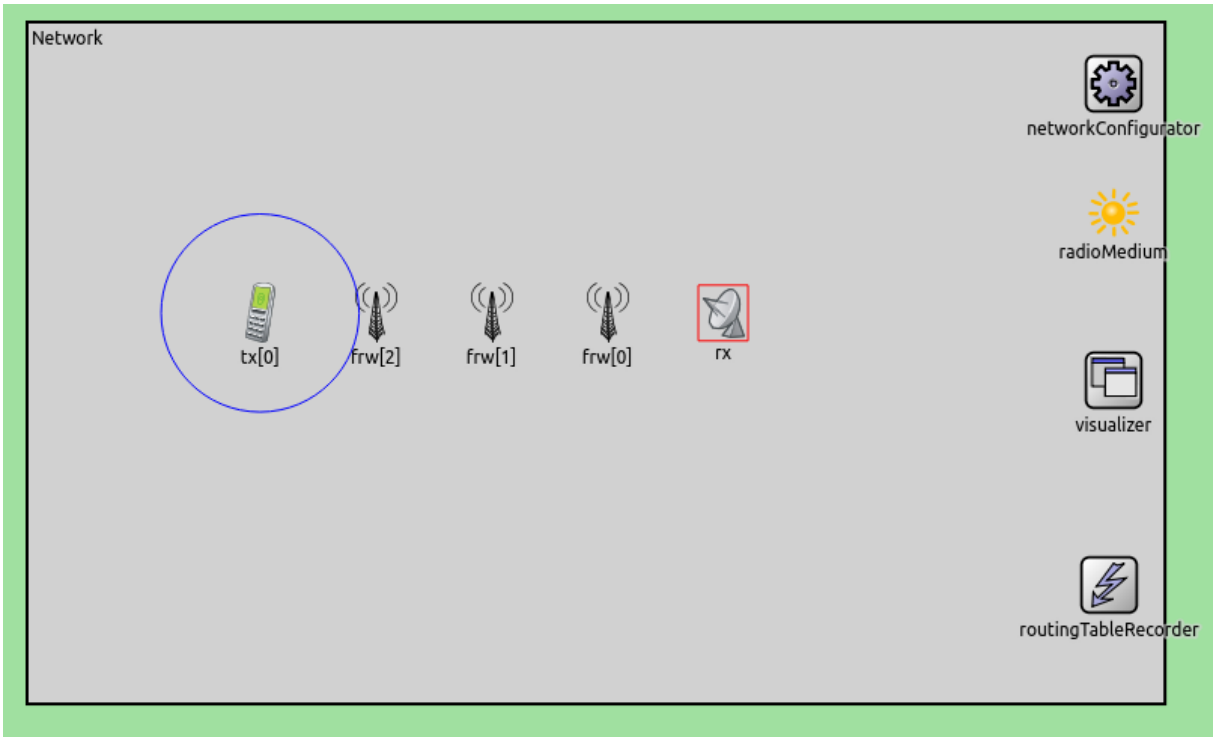


Figura 3 - InLineForwarder

Nella seconda configurazione (*InLineForwarder*) la tipologia di mobilità scelta per il Ricevitore ***rx*** e l'End Node ***tx*** è *"LinearMobility"*: i nodi infatti si muovono in maniera lineare lungo una retta parallela all'asse X allontanandosi dai Forwarding Node fino a raggiungere i confini dell'area del Network per poi riavvicinarsi.

I Forwarding Node sono invece statici (*"StationaryMobility"*).

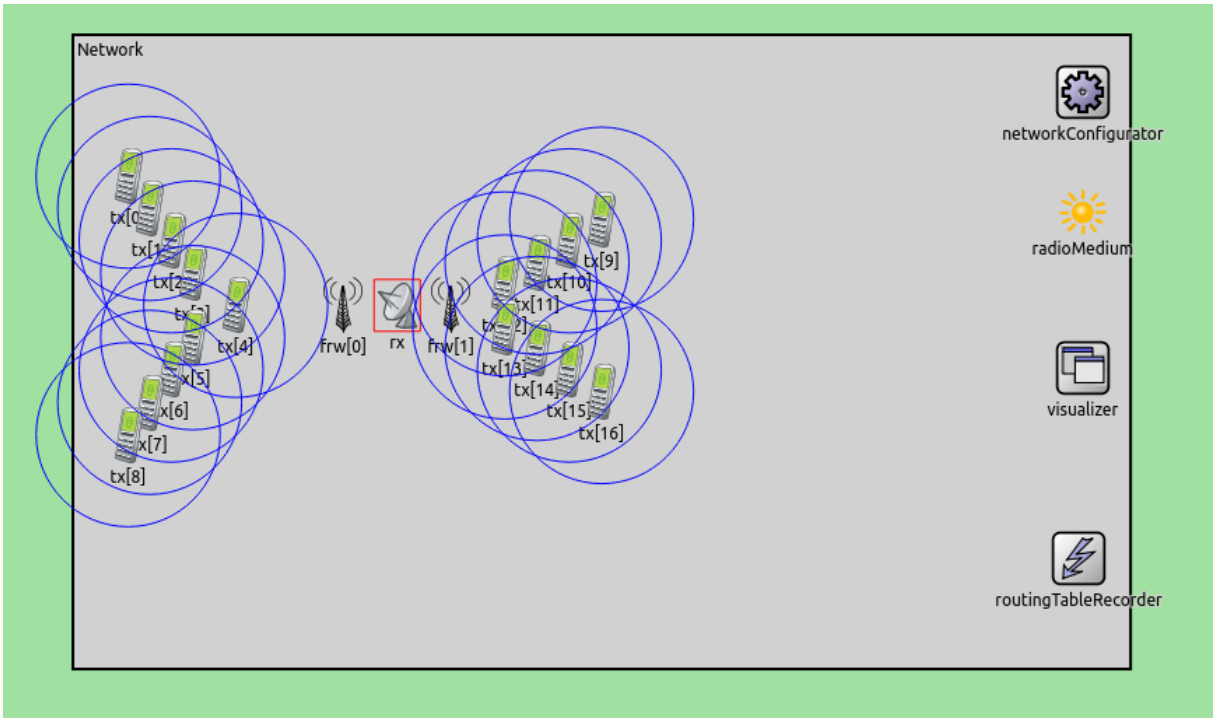


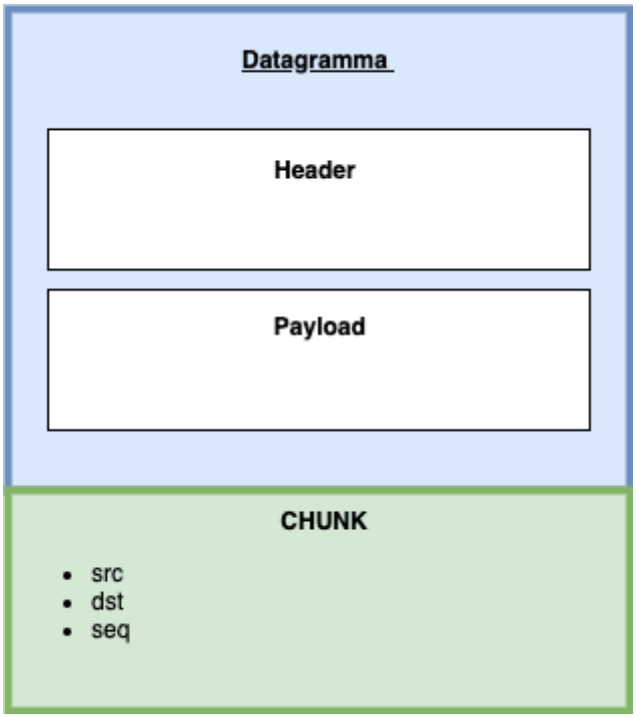
Figura 4 - Static Nodes

Infine, la terza configurazione (*StaticNodes*) è caratterizzata dal fatto che tutte e tre le tipologie di nodo sono statiche *"StationaryMobility"*.

3. Scelte implementative

Al fine di soddisfare le specifiche, si è ritenuto opportuno aggiungere al datagramma un "chunk" contenente i campi richiesti:

- **src**: indirizzo sorgente
- **dst**: indirizzo destinazione
- **seq**: numero di sequenza



La simulazione prevede che questo pacchetto venga generato da un mittente e arrivi al nodo ricevitore tramite un percorso predefinito attraverso i forwarding node.

É possibile tracciare questo percorso facendo uso del parametro **nextHop** che specifica quale nodo, tra tutti quelli che ricevono il pacchetto in broadcast, rappresenta l'hop successivo.
Infatti, tutti i nodi che ricevono un pacchetto broadcast lo scarteranno, eccezion fatta per il nextHop che invece lo inoltrerà in broadcast a sua volta.
Il procedimento appena descritto, eseguito per ogni pacchetto generato, si ripeterà per ogni nodo coinvolto, fino a quando il pacchetto arriverà a destinazione.

Il parametro nextHop viene inizializzato all'interno del file *omnetpp.ini* e gestito all'interno del file *NextHopForwarding.cc*, dove sono state apportate alcune modifiche per rendere possibile l'utilizzo del chunk precedentemente introdotto.

Il chunk in questione viene descritto nel file *backChunk.msg* situato al seguente path:
`inet/src/inet/networklayer/nexthop/backChunk.msg`

4. Statistiche

In queste simulazioni, l'oggetto del nostro interesse è l'**end-to-end delay** ovvero il tempo impiegato da un pacchetto per essere trasmesso attraverso la rete dal nodo sorgente alla destinazione.

Per ciascuna configurazione di rete è stato misurato l'end-to-end delay evidenziando i valori massimo, minimo e medio, valutato in due differenti condizioni di canale: una prevede un fattore di attenuazione α=3 e una varianza σ=7 mentre l'altra, richiesta dal docente, prevede α=4.03 e σ=4.98. Questa seconda condizione si è rivelata la peggiore poiché quasi la totalità dei pacchetti inviati dagli End Node non veniva mai ricevuta a causa delle pessime condizioni del canale radio.

5. Risultati

Configurazione Completa

La configurazione Completa eseguita per 5 secondi con condizioni di canale **α**=3 e **σ**=7, ha generato il seguente grafico relativo all'end-to-end delay:

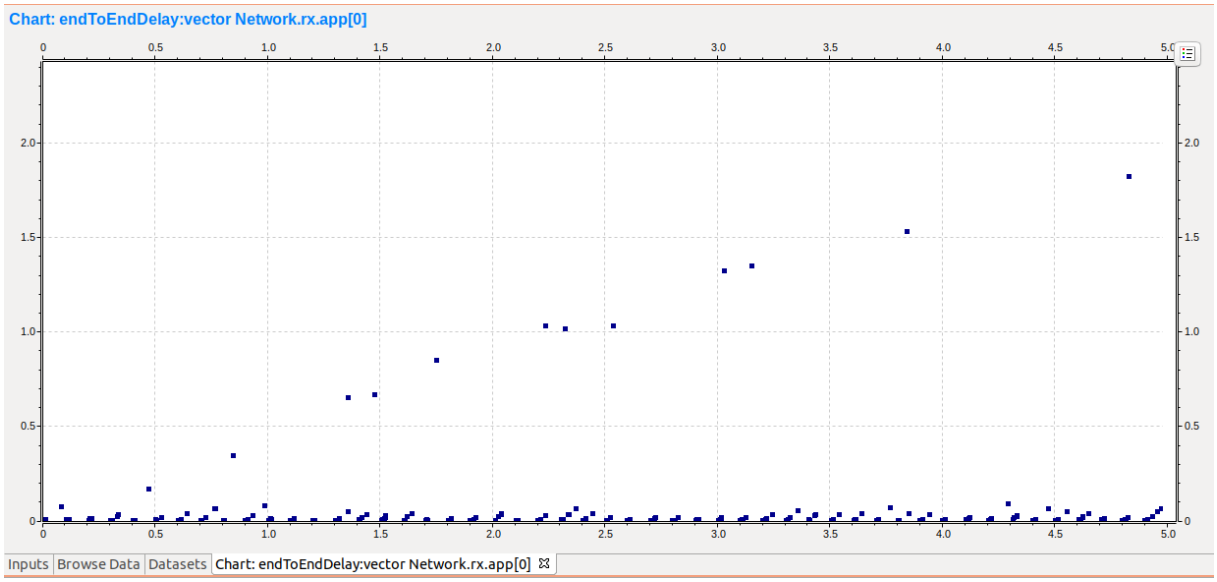


Figura 5 - End-to-end delay Configurazione Completa

Dal grafico si evince che i valori di end-to-end delay sono tendenzialmente bassi ad eccezione di qualche pacchetto che vede il proprio valore di end-to-end delay aumentare con un andamento lineare.
La motivazione potrebbe essere riconducibile al fatto che nella maggior parte dei casi, le code che si creano all'interno dei nodi che inoltrano i pacchetti vengono gestite in maniera efficace, mentre in una minima parte, saturano e quindi in quelle occasioni si registra un aumento sporadico dell'end-to-end delay.

Nella figura successiva sono riportati i valori di end-to-end delay tra i quali quelli su cui ci focalizziamo: **massimo, minimo e medio**.

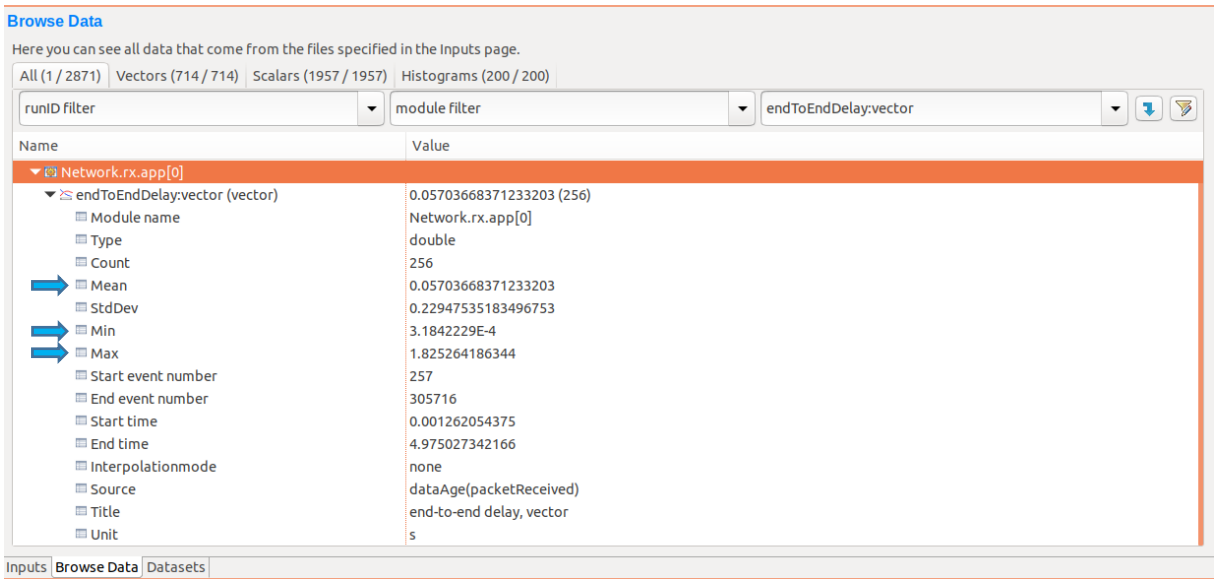


Figura 6 - Statistiche

Cambiando le condizioni di canale, ovvero impostando un fattore di attenuazione **α** = 4.03 e una varianza **σ** = 4.98, il numero di pacchetti ricevuti da ***rx*** è molto inferiore al precedente e inoltre è necessario che i nodi siano più vicini per compensare le sfavorevoli condizioni del canale.

Poiché gli End Node sono mobili e fino a 1.6 secondi nessuno di questi è sufficientemente vicino ad un Forwarding Node, non si registra ricezione di pacchetti.
Successivamente, quando uno degli End Node entra in un range di copertura favorevole alla comunicazione con uno dei Forwarding Node, vengono ricevuti i primi pacchetti e l'end-to-end delay è tendenzialmente basso poiché sia il mittente che il ricevitore si trovano addossati ad un Forwarding Node.

Dal grafico seguente si nota che l'end-to-end delay dei primi pacchetti inviati è maggiore rispetto ai valori successivi e ciò potrebbe essere riconducibile al fatto che in quegli istanti il mittente è appena entrato all'interno del range di copertura del Forwarding Node e in queste condizioni di canale molto sfavorevoli il risultato che ne consegue è quello misurato.

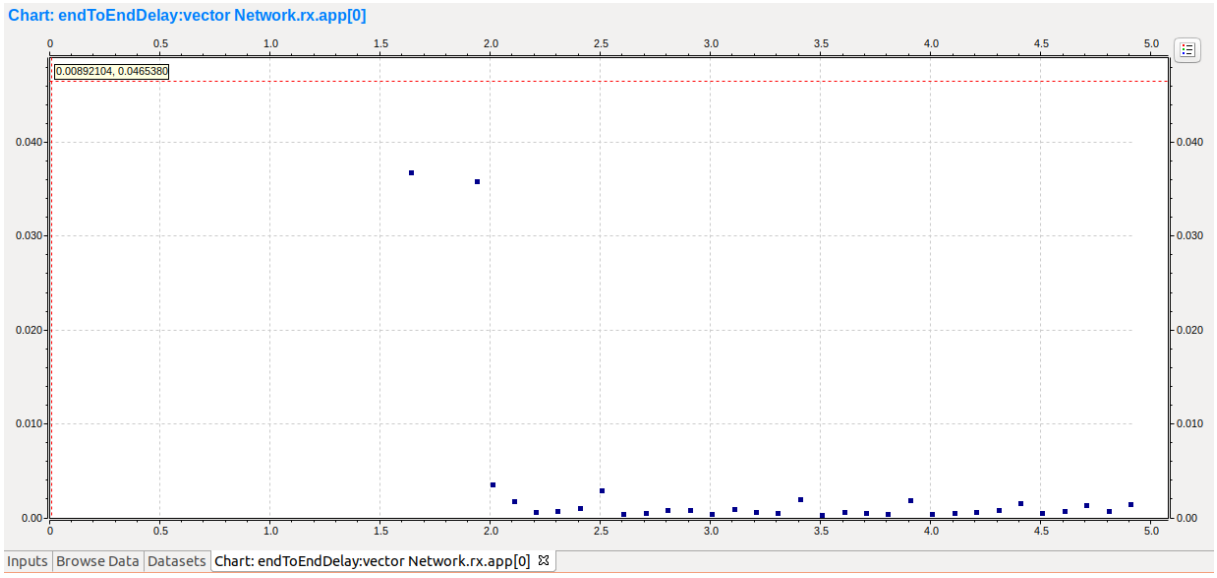


Figura 7 - End-to-end delay Config. Completa α= 4.03, σ= 4.98

Di seguito sono riportate le statistiche riguardanti l'end-to-end delay ottenute in *"configurazione Completa"* con cattive condizioni di canale.

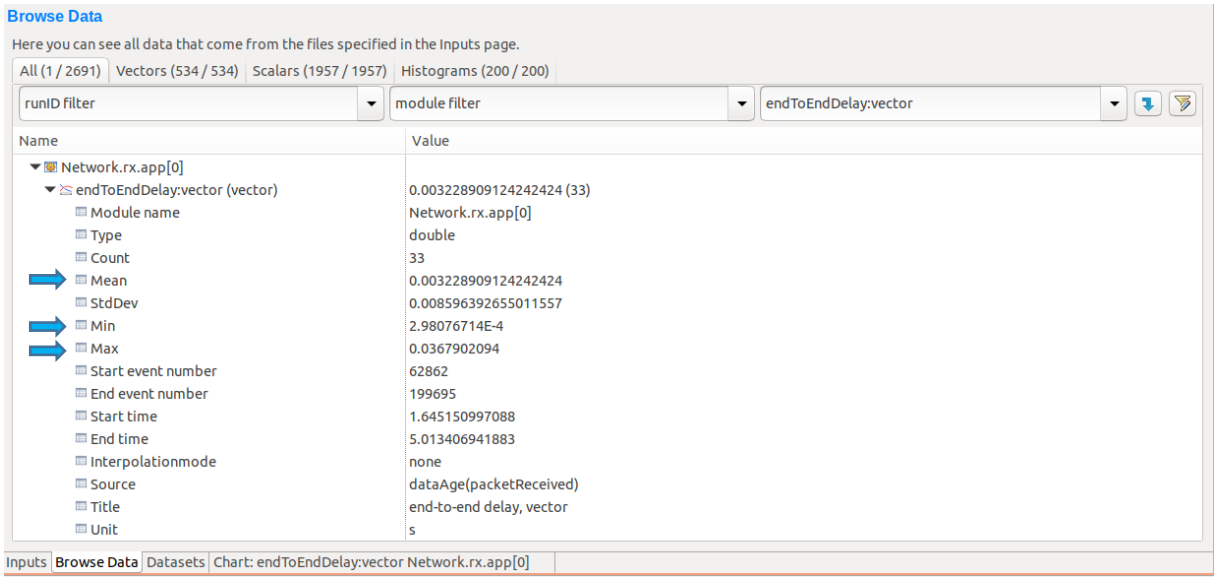


Figura 8 - Statistiche

Configurazione InLineForwarder

In questa configurazione, con lo scorrere del tempo, il mittente e il ricevitore si allontanano dal range di copertura dei Forwarding Node per poi ritornare al suo interno.

Come mostrato dal grafico, il numero totale di pacchetti arrivati a destinazione è molto basso e i relativi valori di end-to-end delay presentano una distribuzione irregolare ma comunque compresa tra 0 e 0,1 dovuta alla mobilità del mittente e del destinatario che si allontanano dai Forwarding Node.

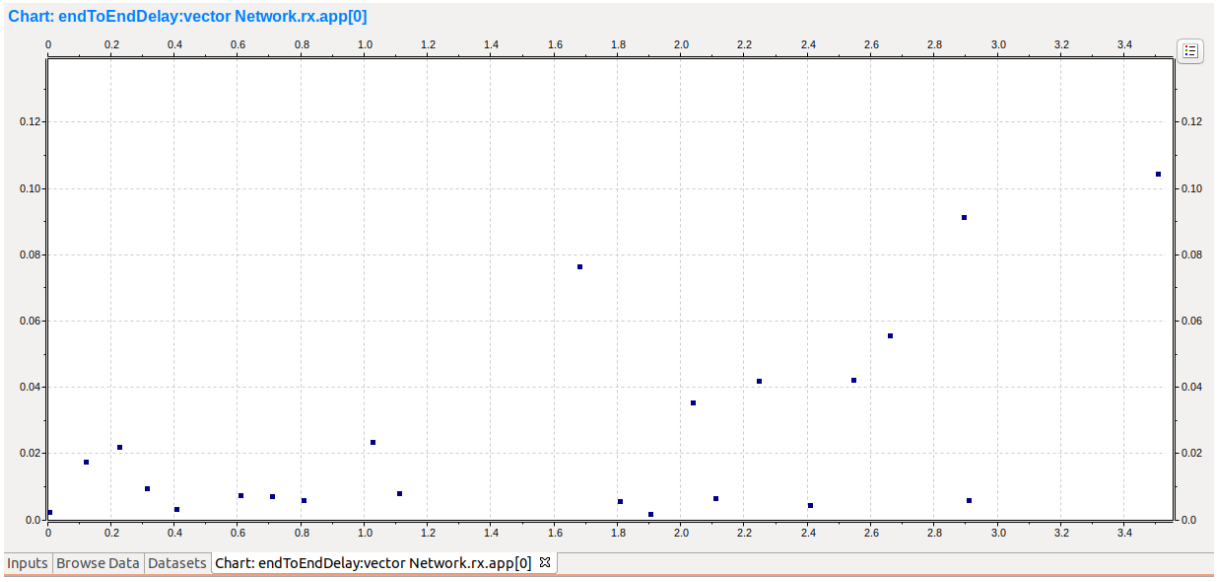


Figura 9 - End-to-end delay InLineForwarder

In conclusione, dato il range di valori che assume la distribuzione in esame, possiamo dire che i pochi pacchetti che giungono a destinazione mantengono un valore basso di end-to-end delay.

Alla fine della simulazione della durata di circa 3 secondi, il ricevitore ha ricevuto 22 pacchetti e il grafico relativo alle statistiche di end-to-end delay è il seguente:

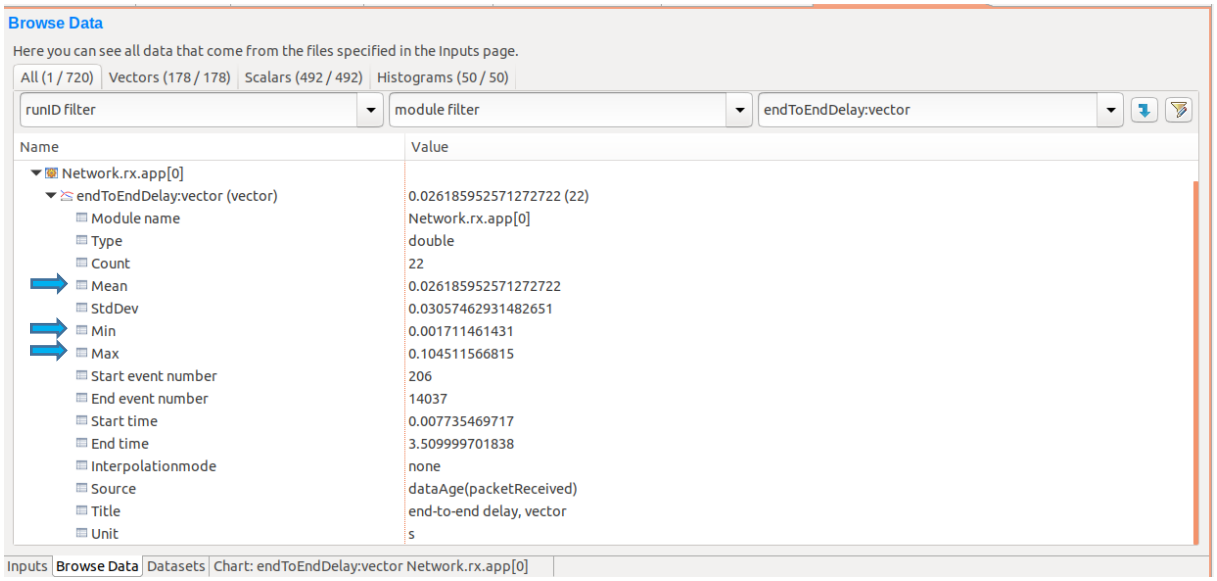


Figura 10 - Statistiche InLineForwarder

Impostando le condizioni di canale ai valori **α**=4.03 e **σ**=4.98 e in base alle posizioni statiche dei Forwarding Node, il numero di pacchetti ricevuti è nullo e quindi non ha senso parlare di end-to-end delay.

Configurazione StaticNodes

Infine, il grafico relativo all'end-to-end delay della terza configurazione, rilevato per valori di canale **α**=3 e **σ**= 7, mostra la presenza di valori bassi di end-to-end delay compresi tra 0 e 0,1.

Comunque, il leggero incremento di end-to-end delay relativo ai pacchetti inviati nello stesso istante di tempo è indicativo del fatto che le code dei due Forwarding Node iniziano a riempirsi, aumentando il tempo necessario per la gestione dei pacchetti.

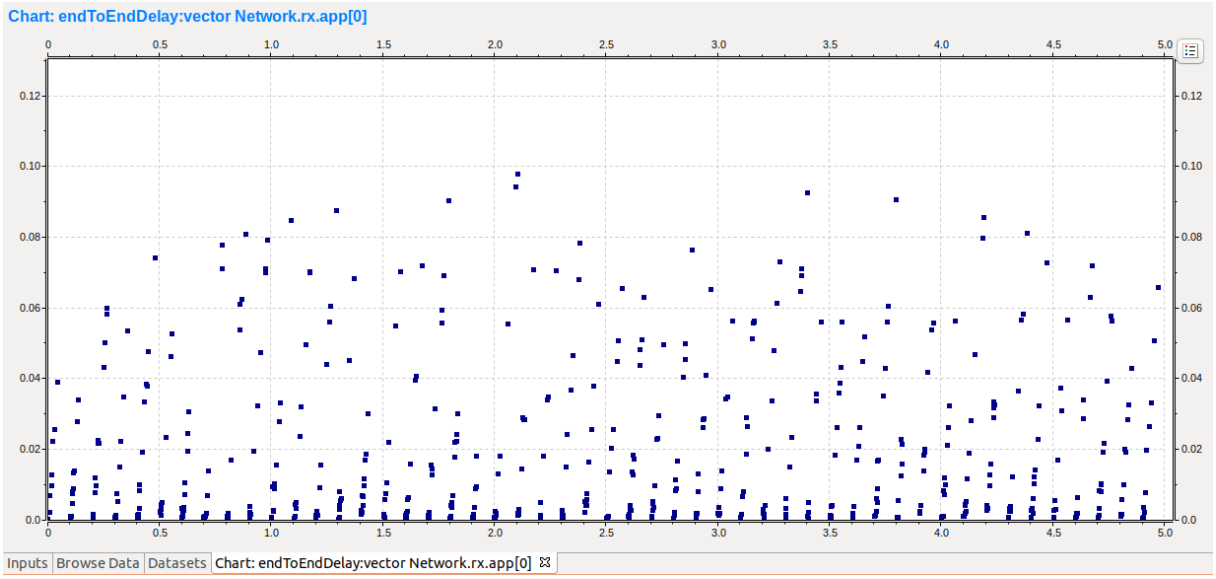


Figura 11 - End-to-end delay StaticNodes

Nonostante ciò, non si ha un ritardo significativo nella trasmissione dei pacchetti tra sorgenti e destinatario, quindi il tempo di trasmissione è accettabile.

Di seguito sono elencati tra gli altri, i valori di massimo, minimo e medio end-to-end delay.

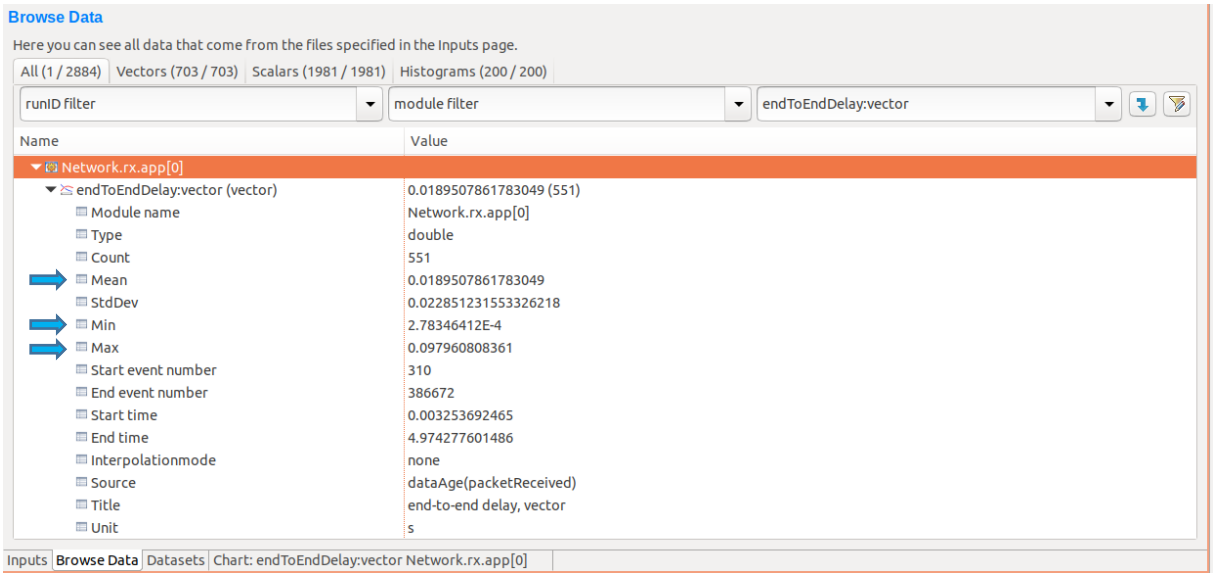


Figura 12 - Statistiche

Peggiorando le condizioni di canale, si nota che viene persa la quasi totalità dei pacchetti e l'unico ad essere ricevuto è quello proveniente da uno dei due nodi più vicini al Forwarder come evidenziato nella seguente figura che mostra le specifiche del pacchetto: in particolare il nodo sorgente è ***tx[13]***.

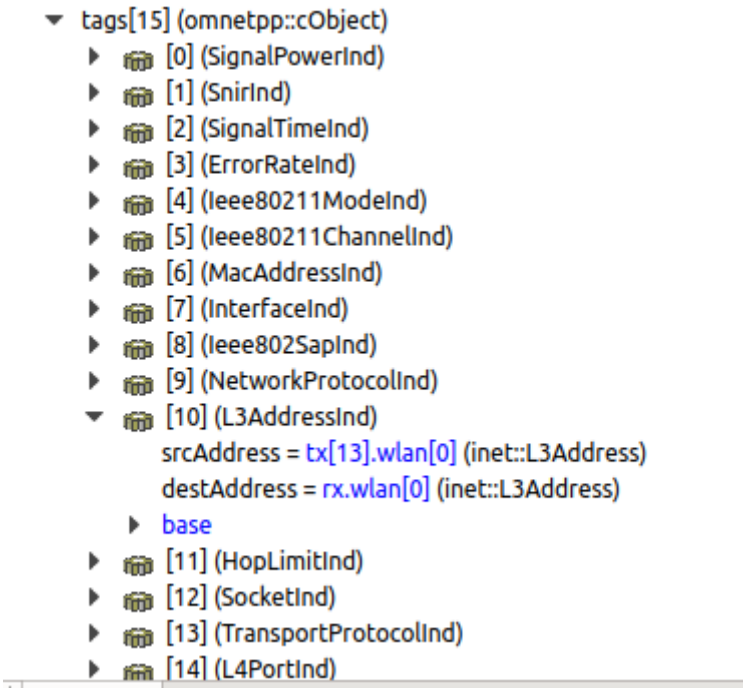


Figura 13 - Specifiche di pacchetto

6. Indice delle figure

Andrea Calabretta

Alessandro Mauro