A Tandem Queueing Model for Delay Analysis in Disconnected Ad Hoc Networks

An OMNeT++ simulation

Giulia Cantini

Introduzione

Una rete opportunistica:

- è tollerante ai ritardi (*DTN*)
- implementa il paradigma *store-carry-forward*
- consente la trasmissione di dati solo in presenta di contatto

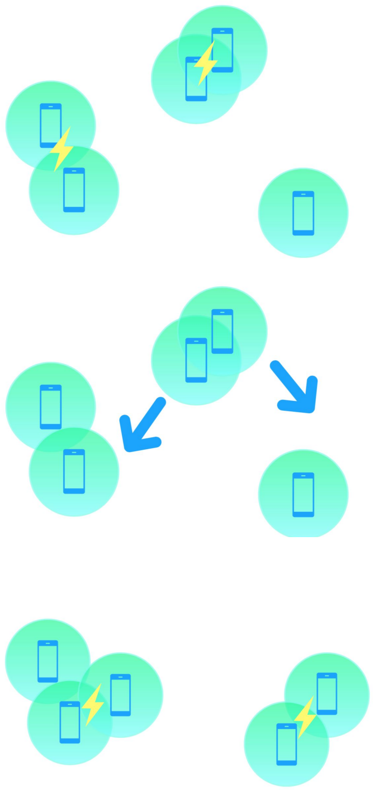

Fig.1: *Store-carry-forward.*

Modello

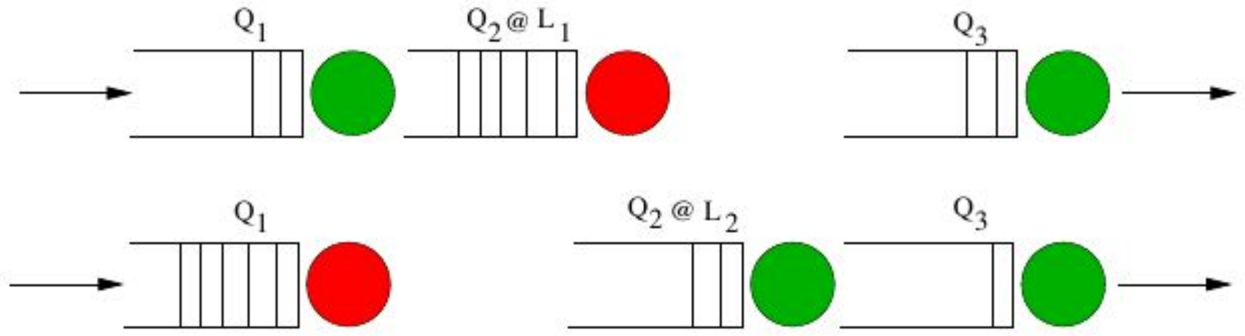

Fig. 2: Modello a tre code con una coda mobile. Quando Q_2 è in L_1, S_2 è down e S_1 è up. Quando Q_2 è in L_2, S_2 è up e S_1 down.

Implementazione

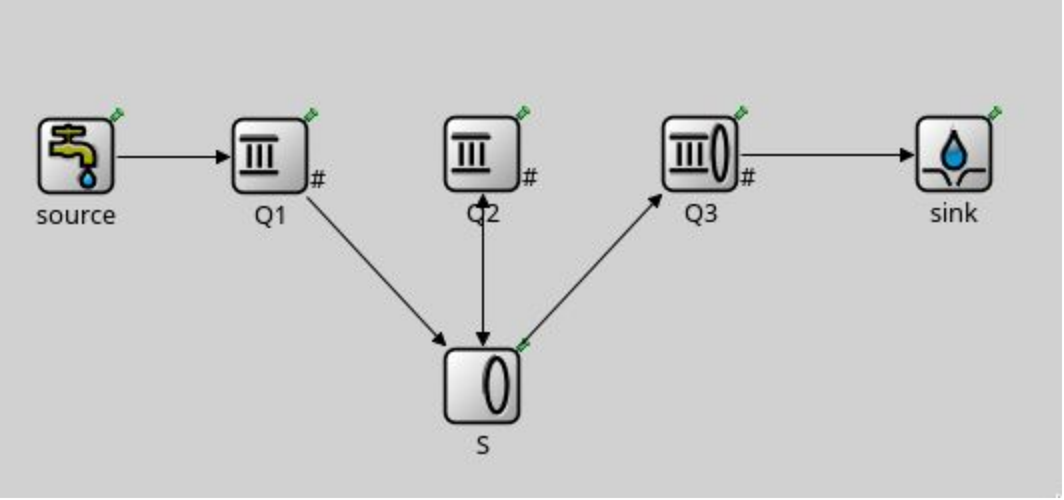

Fig.3: Visualizzazione del file *.ned* in OMNeT.

Meccanismo opportunistico

```
void OppServer::handleMessage(cMessage *msg) {

    EV << "Message name " << msg->getName() << endl;

    // Self-messages
    if (msg == startSwitchEvent) {
         serverIsAvailable = false;
         // If server is not doing anything
         if (!jobServiced)
             scheduleAt(simTime()+switchOverTime, endSwitchOverTimeEvent);
         // If jobServiced = true then the server will process the job and the next event
         // will be a end service event
    }
    else if (msg == endSwitchOverTimeEvent) {
        // Invert isServingQ1 (0 to 1 and 1 to 0)
        serverIsAvailable = true;
        isServingQ1 = !isServingQ1;

        // Schedule next switch event
        if (isServingQ1)
            scheduleAt(simTime()+Q1visitTime, startSwitchEvent);
       else
            scheduleAt(simTime()+Q2visitTime, startSwitchEvent);

        requestJob();
    }
```

```
else if (msg == endServiceMsg) {

    simtime_t d = simTime() - endServiceMsg->getSendingTime();
    jobServiced->setTotalServiceTime(jobServiced->getTotalServiceTime() + d);

    EV << "Deciding where to send the job, isServingQ1 is " << isServingQ1 << "\n";
    if (isServingQ1) {
        // Send the job out to Q2
        EV << "Sending the job out to Q2" << endl;
        send(jobServiced, "out", 0);
    }
    else {
        // Send the job out to Q3
        EV << "Sending the job out to Q3" << endl;
        send(jobServiced, "out", 1);
    }
```

Simulazione

Parametri

- processo di arrivo = exp di media (5s, 7.5s, 10s)
- tempo di servizio di $S_1 = S_2$ = exp di media 1s
- tempo di servizio di S_3 = exp di media 0.8s
- switch-over time = exp di media 1.1s
- tempo di permanenza in L_1 = exp di media 1.3s
- tempo di permanenza in L_2 = exp di media 1.6s

Misure

- tempo medio di risposta del sistema
- numero medio di utenti in coda nel sistema
- throughput medio

Transiente iniziale

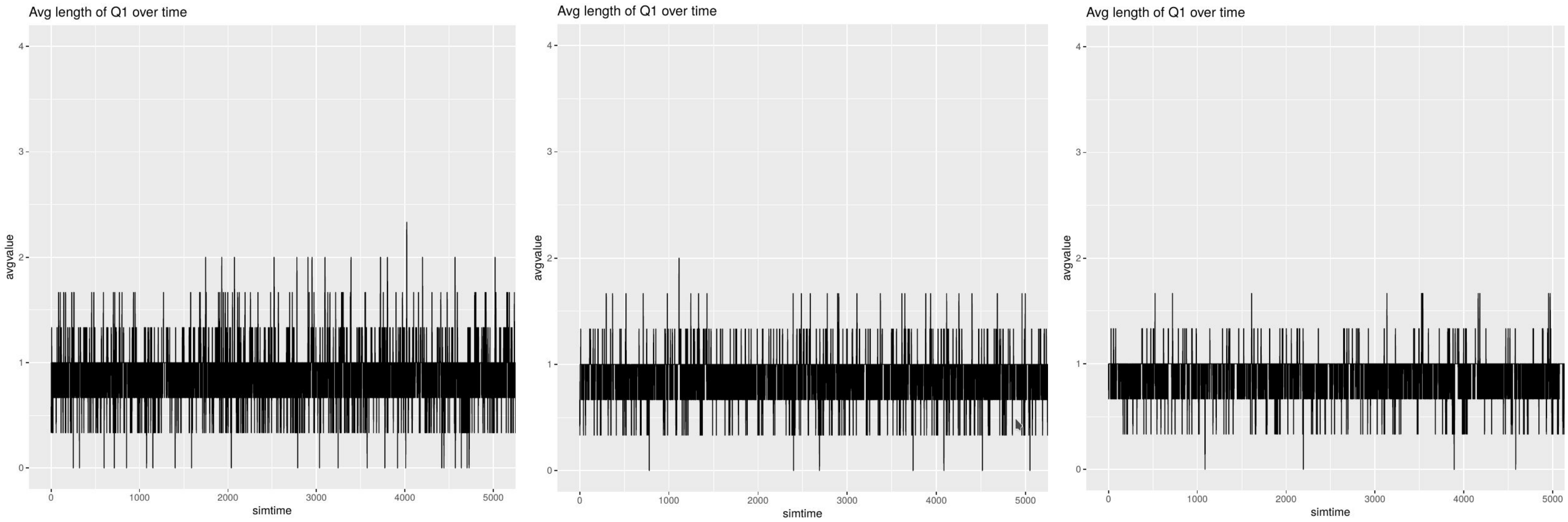

Fig. 4: Lunghezza coda Q_1 per p = (5s, 7.5s, 10s)

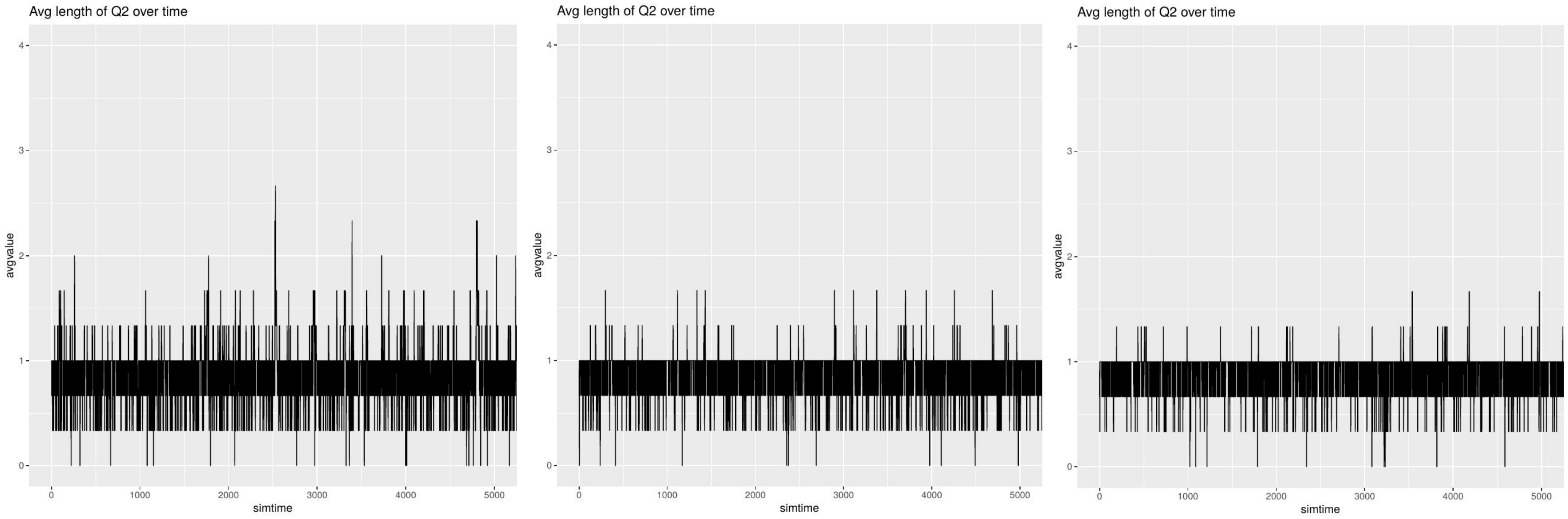

Fig. 4: Lunghezza coda Q_2 per p = (5s, 7.5s, 10s)

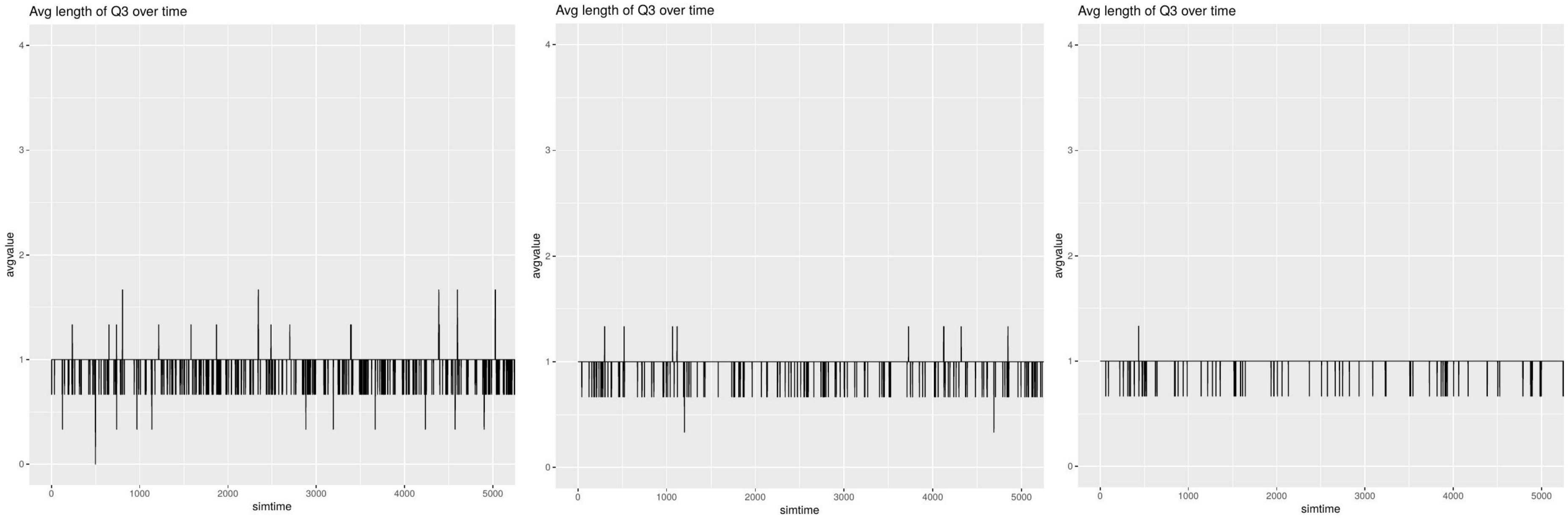

Fig. 4: Lunghezza coda Q_3 per p = (5s, 7.5s, 10s)

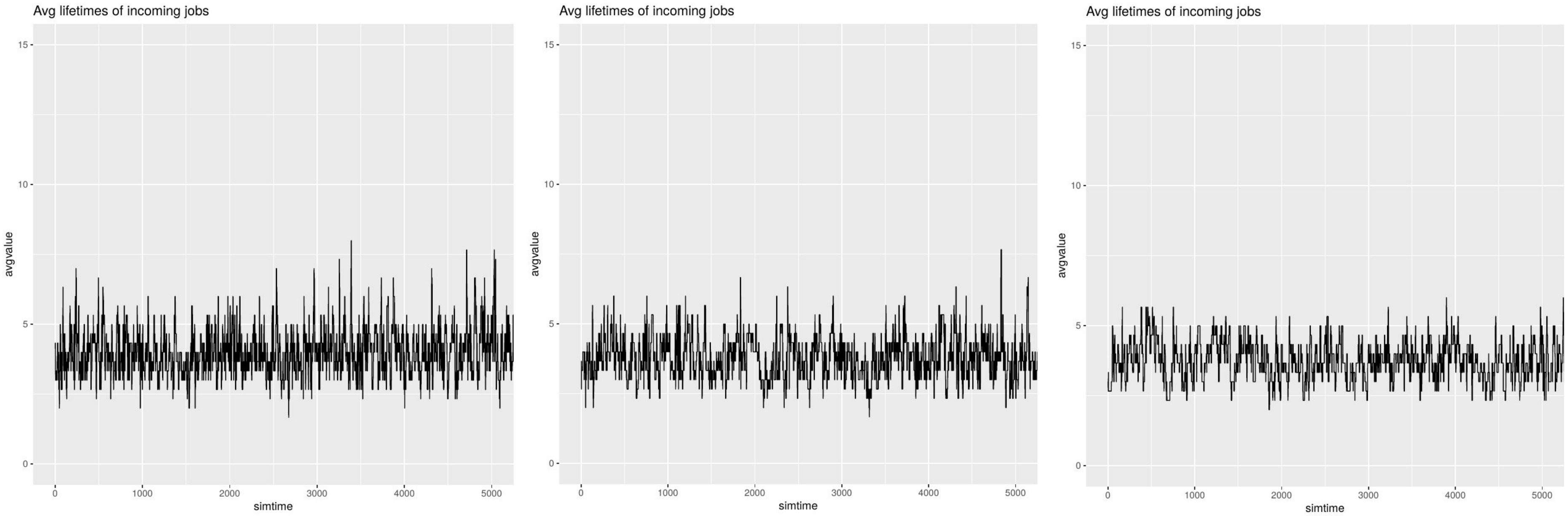

Fig. 4: Lifetime per p = (5s, 7.5s, 10s)

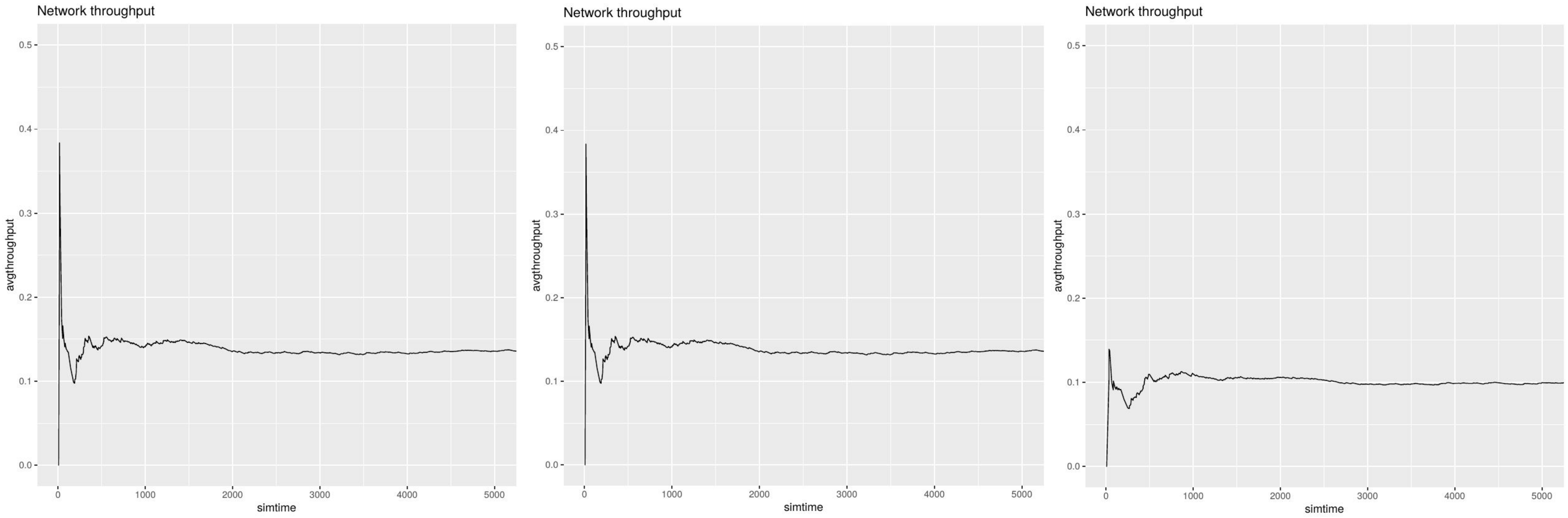

Fig. 4: Throughput per p = (5s, 7.5s, 10s)

Risultati

p	q1 length: mean	q1 length: var	q1 length: int l	q1 length: int r
5	0.74333	0.01582	0.70431	0.78235
7.5	0.67500	0.00944	0.64486	0.70514
10	0.62167	0.00839	0.59326	0.65008
p	q2 length: mean	q2 length: var	q2 length: int l	q2 length: int r
5	0.76667	0.02764	0.71510	0.81824
7.5	0.69500	0.01196	0.66107	0.72893
10	0.63833	0.00977	0.60767	0.66899
p	q3 length: mean	q3 length: var	q3 length: int l	q3 length: int r
5	0.77000	0.04217	0.70630	0.83370
7.5	0.69333	0.01771	0.65205	0.73461
10	0.65000	0.01276	0.61496	0.68504
p	lifetime: mean	lifetime: var	lifetime: int l	lifetime: int r
5	3.83960	0.06389	3.76119	3.91801
7.5	3.67062	0.04146	3.60745	3.73379
10	3.62975	0.02790	3.57793	3.68157
p	throughput: mean	throughput: var	throughput: int l	thoughput: int r
5	0.19764	0.00009	0.19470	0.20058
7.5	0.13377	0.00001	0.13279	0.13475
10	0.09729	0.00004	0.09533	0.09925

Tabella 1: numbatch = 30, numobs = 40

p	q1 length: mean	q1 length: var	q1 length: int l	q1 length: int r
5	0.73417	0.00857	0.70951	0.75883
7.5	0.66500	0.00689	0.64289	0.68711
10	0.62500	0.00494	0.60628	0.64372
p	q2 length: mean	q2 length: var	q2 length: int l	q2 length: int r
5	0.76083	0.01435	0.72892	0.79274
7.5	0.67667	0.00810	0.65269	0.70065
10	0.63750	0.00616	0.61659	0.65841
p	q3 length: mean	q3 length: var	q3 length: int l	q3 length: int r
5	0.77083	0.01733	0.73576	0.80590
7.5	0.66500	0.00928	0.63934	0.69066
10	0.63000	0.00609	0.60921	0.65079
p	lifetime: mean	lifetime: var	lifetime: int l	lifetime: int r
5	3.80783	0.03773	3.75608	3.85958
7.5	3.68995	0.01755	3.65466	3.72524
10	3.63671	0.01900	3.59999	3.67343
p	throughput: mean	throughput: var	throughput: int l	thoughput: int r
5	0.19933	0.00004	0.19765	0.20101
7.5	0.13324	0.00005	0.13240	0.13408
10	0.09750	0.00002	0.09631	0.09869

Tabella 2: numbatch = 40, numobs = 60

p	q1 length: mean	q1 length: var	q1 length: int l	q1 length: int r
5	0.73300	0.00247	0.70419	0.76181
7.5	0.66000	0.00156	0.63710	0.68290
10	0.62800	0.00100	0.60967	0.64633
p	q2 length: mean	q2 length: var	q2 length: int l	q2 length: int r
5	0.76000	0.00549	0.71705	0.80295
7.5	0.67300	0.00225	0.64550	0.70050
10	0.64100	0.00074	0.62523	0.65677
p	q3 length: mean	q3 length: var	q3 length: int l	q3 length: int r
5	0.77900	0.01012	0.72069	0.83731
7.5	0.67800	0.00393	0.64166	0.71434
10	0.63100	0.00399	0.59438	0.66762
p	lifetime: mean	lifetime: var	lifetime: int l	lifetime: int r
5	3.80277	0.01357	3.73524	3.87030
7.5	3.69224	0.00295	3.66076	3.72372
10	3.63064	0.00268	3.60063	3.66065
p	throughput: mean	throughput: var	throughput: int l	thoughput: int r
5	0.19866	0.00002	0.19607	0.20125
7.5	0.13312	0.00005	0.13129	0.13495
10	0.09734	0.00001	0.09551	0.09917

Tabella 3: numbatch = 10, numobs = 200

Riferimenti

- A. Al-Hanbali, R. de Haan, R. J. Boucherie, J. van Ommeren, *A Tandem Queueing Model for Delay Analysis in Disconnected Ad Hoc Networks*
- *OMNeT++ Documentation*, https://docs.omnetpp.org/
- *OMNeT++ User Guide*, https://doc.omnetpp.org/omnetpp/UserGuide.pdf
- *OMNeT++ Simulation Manual*, https://doc.omnetpp.org/omnetpp/manual/
- Emmanuel Paradis, *R for Beginners*, https://cran.r-project.org/doc/contrib/Paradis-rdebuts_en.pdf.