
University of Pisa
School of Engineering

---

Performance Evaluation of Computer Systems and Networks

# Epidemic Broadcast

**Supervisors**
*Prof. Giovanni Stea*
*Ing. Antonio Virdis*

**Students**
*Marco Pinna*
*Rambod Rahmani*
*Yuri Mazzuoli*

February 14, 2021

# Contents

# Chapter 1

# Introduction

In what follows the study on the broadcast of an epidemic message is carried out. The specifications are detailed in the following:

> ### Epidemic broadcast
> Consider a 2D floorplan with $N$ users randomly dropped in it. A random user within the floorplan produces a *message*, which should ideally reach all the users as soon as possible. Communications are *slotted*, meaning that on each slot a user may or may not relay the message, and a message occupies an entire slot. A *broadcast radius* $R$ is defined, so that every receiver who is within a radius $R$ from the transmitter will receive the message, and no other user will hear it. A user that receives more than one message in the same slot will not be able to decode any of them (*collision*). Users relay the message they receive *once*, according to the following policy (*p-persistent relaying*): after the user successfully receives a message, it keeps extracting a value from a Bernoullian RV with success probability $p$ on every slot, until it achieves success. Then it relays the message and stops. A sender does not know (or cares about) whether or not its message has been received by its neighbors.
>
> Measure at least the broadcast time for a message in the entire floorplan, the percentage of covered users, the number of collisions.
>
> In all cases, it is up to the team to calibrate the scenarios so that meaningful results are obtained.

The work is organized as follows:

- Firstly, an initial overview and a presentation of the problem are given. Here, meaningful parameters to be tweaked and useful scenarios are identified and some considerations about them are made.
- Secondly, a graph-based modelling technique, commonly used in literature, is proposed and some simplified scenarios are analysed.

- Then, in Chapter 4 the complete model is considered and some assumptions about it are made, such as its performance when one or more parameters are set to extreme values or its asymptotic behaviour with different configurations of the parameters.
- In chapter 5 the development of the simulator is described and the results of its validation are presented.
- Chapter 6 concerns the full simulation and performance evaluation of the system.

# Chapter 2

# Overview

To perform the analysis of the broadcast of a message that should reach as many users as possible in a 2D floorplan, the following hypotheses were made:

- the floorplan always has a rectangular shape and it is empty, (i.e. there are no obstacles such as walls or pillars in it);
- each user is thought as a point mass and does not move inside the floorplan;
- the transmission of a message is instantaneous and it happens at the beginning of every time slot; the whole apparatus can therefore be considered a *Discrete Time System*.

Depending on the performance metrics to be analysed, different choices can be made about the parameters to be adjusted. According to the specifications, the main three metrics for this study are:

- the broadcast time `T` for a message to cover as many user as possible in the entire floorplan; the effective duration of the transmissions slots obviously has an effect on the total broadcast time[1], but it is only a scaling factor on the total number of slots. `T` can therefore be measured in terms of slots and converted to units of time accordingly.
- the percentage of covered users `U`;
- the number of collisions `C`; collisions are detected by nodes, when they try to receive more than one message in the same slot; in order to messure the number of collisions, we consider a single event of collision every time a node detect one of them; every statistical analysis is done taking this assumption.

The following parameters have been identified: the transmission range of the users, the *per-slot* transmission probability, the floorplan size and its shape, and the density of users in the floorplan per square metre.

[1] The total amount of time required for a broadcast message to cover the entire floorplan.

To be more in detailed:

- the radius of transmission `R`: it represents the maximum distance between two users such that the message sent from one is detected by the other; it is the same for every user on the floorplan; realistic values for $R$ have been taken from Bluetooth Low Energy standard and range from a minimum of 5 metres to a maximum of 50 metres.
- the *per-slot* transmission probability `p`: it is the success probability for the Bernoulli random variable associated to the transmission; as a probability, it can assume values between 0 and 1;
- the width of the side of the floorplan rectangle `L`;
- the *aspect ratio* of the floorplan rectangle `a`, defined as the ratio between the longer side $L$ and the shorter side $W$; because of the radial symmetry of the transmission phenomenon, there is no actual need to consider values for `a` lower than 1; this parameter is fixed at 1 in our simulation model, so we are considering only sqare floorplans; this will allow us to focus the analysis on other parameters, but still considering a quite common scenario;
- number of users `N`, defined as the number of users in the floorplan;
- users population density `d`, defined as the number of users per square meter. We choose to study two opposite density scenarios (high: $1/m^2$, low $0.1/m^2$).

The rationales behind the choices of the parameters were the following:

- the transmission radius clearly has a great impact on all the performance metrics; the greater the radius, the faster the message moves across the floorplan and the higher the number of users that can be reached; on the other hand, a greater radius is likely to cause more collisions than a smaller one;
- the higher the transmission probability, the faster a message "moves away" from a user; at the same time, a high transmission probability implies a high collision probability in a local area where two or more nodes are transmitting;
- a bigger floorplan area, all else being equal, will require a longer time to be covered entirely by the broadcast message;
- a very long and very narrow floorplan will probably cause less collisions than a square one with the same area, as the average number of users in the collision range of other users decreases; on the other hand, the performance of this type of scenario will be higly influenced by the position of the starting node;
- in order to exploit various values of population density `d` we choose to fix the total users `N`=100, then we choose the 2 values for $L$: 10 metres and 100 metres.

# Chapter 3

# System modelling

Wireless communication networks have been extensively studied in scientific literature and one of the most used mathematical tools to model them and their behaviour is *graph theory*. In this work the same approach was used.
If we think of the users involved in the broadcasting of the message as the nodes of a graph, as the broadcast propagation goes on and the message is transmitted among the nodes, the system goes through different states where each node could be either transmitting, listening or stopped. We can therefore think of the state of the entire system as a combination of the states of each node. The different nodes behaviours were modelled by means of three different states:

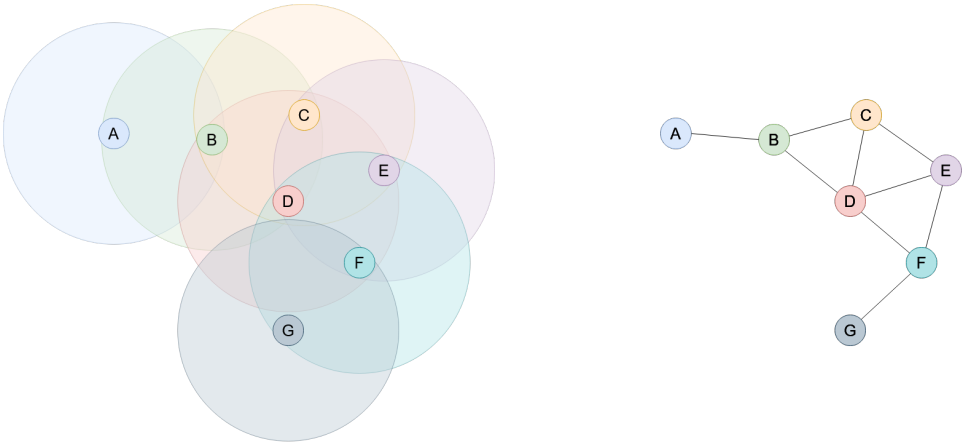
- `listening`: the node has not received the broadcast message yet and therefore it is still listening for incoming messages from other nodes;

- `transmitting`: the node has received the message and during each time slot it is trying to transmit it to adjacent nodes; in what follows, a node in `transmitting` state will be referred to as *active* node;

- `sleeping`: the node has already received the message and retransmitted it; once a node is in a `sleeping` state, it will remain in such state and will therefore have no effect on the system any more.

## 3.1 Graph model for wireless systems

The $N$ users dropped on the floorplan make up the set of vertices $V$ of a graph $G$, whose set of edges $E$ is composed by all the connections between each two nodes in reach of one another. Consider the following as a simplified scenario to exemplify this model. In figure 3.1 (a) devices A, C and D are within device B transmission radius. In the equivalent graph, there will be edges that connect B to A, to C and to D. The same goes for all the other vertices. The resulting graph is shown in figure 3.1 (b).




a) Ranges representation b) Equivalent graph representation

Figure 3.1

In general, the existence of an edge from vertex $i$ to vertex $j$ means that nodes $i$ and $j$ are within reach of each other. Two vertices connected by an edge are said to be *adjacent*. The set composed by the vertex $v$ together with all its adjacent vertices form a subgraph called the *neighbourhood of* $v$.
During the broadcast, a node can only receive from and transmit to its neighbourhood.

Once a node has transmitted the message and has gone into `sleeping` state, it disappears, along with all the edges connected to it. Therefore, the set of vertices $V$ changes with time. This is what in literature is called a *dynamic graph* or, more specifically, a *node-dynamic graph*[1].

Modelling the system with graphs also allows for a simple computation of the lower bound for the broadcast time `T`, which can be useful for the validation of the simulator. Given a graph $G(V, E)$ which represents the users in the floorplan, let node $v^*$ be the starting point for the broadcast, i.e. the first node with the message. In the best case scenario, the system evolves with no collisions at all and the message moves along the paths of the graph, reaching all nodes. Let $d(u, v)$ be the *distance* between two vertices $u$ and $v$, i.e. the length of the shortest directed path from $u$ to $v$ consisting of arcs, provided at least one such path exists. Then, the lower bound for the broadcast time is given by the highest distance between $v^*$ and any other vertex. This quantity, in graph theory, is called *eccentricity* of the vertex $v^*$. More formally, the eccentricity of $v^*$ is defined as follows:

$$\epsilon(v^*) = \max_{v \in V} d(v^*, v) \tag{3.1}$$

## 3.2 Simplified models

In what follows, incrementally more complex scenarios are presented showing the reasoning that led us to obtain the final complete model.

### 3.2.1 Single queue configuration

Let us consider a configuration where users are arranged in a line, as shown in figure 5.1. Each user only has two neighbours, except for the outer ones (nodes $A$ and $E$) that only have one. Assuming node $A$ the starting user for the broadcast message, this is the only node in `transmitting` state while all the other nodes are initially in `listening` state.

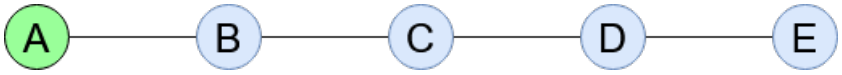



Figure 3.2: Graph of the Single queue configuration

It is clear that, with this configuration, each listening node has a maximum of one active node in its neighbourhood and thus cannot possibly receive the message from two different sources at the same time. This guarantees the absence of collisions. In such a scenario, 100% asymptotic coverage is ensured: during each slot, the active node extracts a Bernoulli RV with success probability $p$. The probability of the active node not transmitting for $k$ consecutive slots is a geometric distribution:

$$P(X = k) = (1 - p)^k. \tag{3.2}$$

The successful transmission of the message from an active node to its neighbour is thus guaranteed since $\lim_{k\to\infty}(1 - p)^k = 0$. As for the total broadcast time `T`, on average it is equal to the mean value of the geometric distribution, $\frac{1-p}{p}$, times the number of hops needed to reach the last node:

$$E[T] = \frac{1 - p}{p} \cdot (N - 1). \tag{3.3}$$

### 3.2.2 Star configuration with one active node

Another useful simple configuration worth analysing is the star-shaped one. In this setup, there is a central node $A$ connected to $N-1$ nodes, all of which are non-adjacent to each other. Let us suppose $A$ to be the broadcast starter. The absence of collisions is ensured in this scenario as well, for the same reason as the previous configuration. At each time slot, $A$ extracts a Bernoulli RV. When the extraction is successful, $A$ broadcasts the message to all of its neighbourhood and total coverage is reached. Hence, 100% asymptotic coverage is ensured in this case too, as the probability of $A$ not transmitting for $k$ consecutive slots is given by Eq. 3.2 and goes to 0 as $k$ goes to infinity.

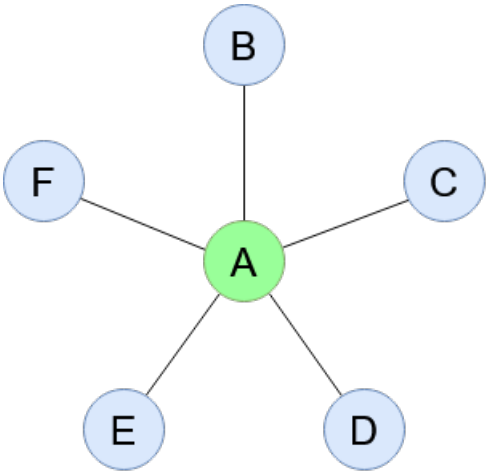



Figure 3.3: Graph of the Star configuration with one active node

In this case, the average total broadcast time $E[T]$ is simply equal to the mean value of the geometric distribution, $\frac{1-p}{p}$.

If the broadcast starter node was not the one placed in the center of the star, there would not be much difference: absence of collisions and total coverage would be ensured as well. As far as it concerns the total broadcast time `T`, its expected value would just be $2 \cdot \frac{1-p}{p}$, since the are now **two** hops involved in the broadcast: one from the starter node to the center node and the another from the center node to all the $N-2$ remaining ones.

### 3.2.3 Star configuration with all but one active node

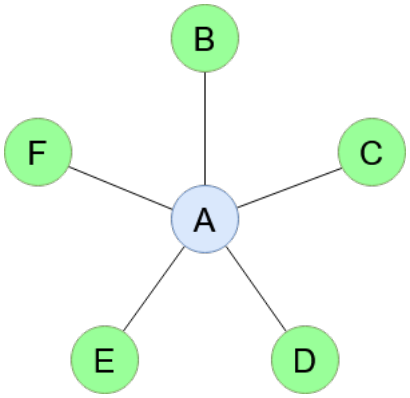



Figure 3.4: Graph of the Star configuration with all but one active node

This configuration can be seen as the complement of the previous one: each node on the ray of the star is active and trying to transmit the message to the center node. This is actually the first scenario where collisions might happen with the possibility that total coverage might never be reached. To simplify the analysis of this system and obtain some insights, it is useful to model it by means of a discrete-time Markov chain.

### Discrete-time Markov chain model for N nodes transmitting to a target

Since the state of the system evolves only once per slot, it can be modelled with a discrete-time Markov chain (DTMC) as follows:

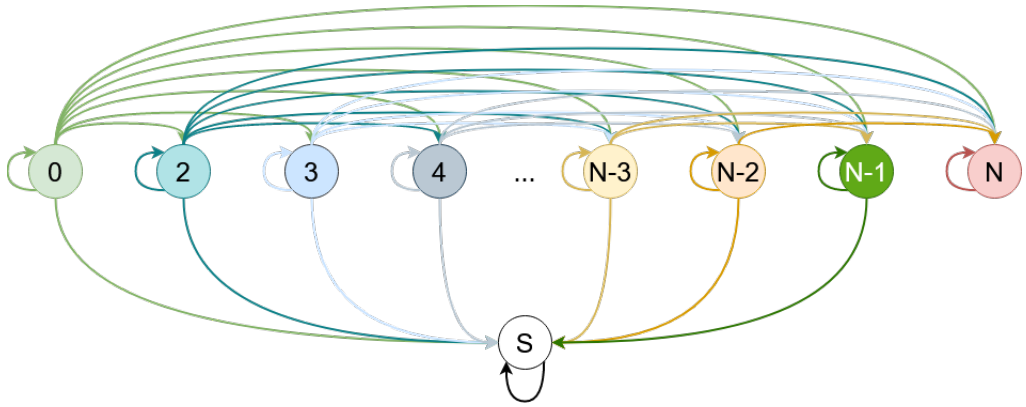



Figure 3.5: Discrete-time Markov chain for a scenario with N active devices

Figure 3.5 shows the DTMC for a generic configuration with $N$ transmitters (transition probabilities are not shown for the sake of clarity).
State 0 and states 2 to $N$ represent the number of `transmitting` devices, namely devices that have already sent the message and have stopped. State $S$ represents the successful transmission state, which the system transitions to when only one device has transmitted the message during the previous slot. The initial state $X_0$ is 0. Any transition from a state $i$ to a state $j$, $j \neq S$, means that more than one device has transmitted the message resulting in the target device detecting a collision. Both state $S$ and state $N$ are *absorbing states*, i.e. states that, once entered, cannot be left (as can be seen in Fig. 3.5, where the only outgoing arrow from each of the two goes back to the state itself).
If the system transitions to state $N$, it stays in it indefinitely since all the devices would be `sleeping` and they cannot become active again. This implies that there will never be total coverage since the target device will forever stay in a `listening` state without ever receiving the message. On the other hand, state $S$, although being an absorbing state as well, should actually be considered as an "exit" state rather than a "sink" state: if the system transitions to this state, the target device has successfully received the message: total coverage has been reached.
Another interesting observation concerns state $N-1$: once the system reaches this state,

it would be guaranteed that the target device will sooner or later receive the message, for the same reason set forth in 3.2.1.

**Transition probabilities**

Let us now address the challenging part: computing the transition probability.
During the first slot, the probability that $j$ devices out of $N$ transmit the message is the following:

$$P_1(j, N) = \binom{N}{j} p^j (1-p)^{N-j} \tag{3.4}$$

Derivation of 3.4 can be found in Appendix A.

As for the probability of $j$ devices transmitting at the same time during slot $k$, be it $P_k(j)$, we can model the system as if it was in the first slot, with the total number of active devices now being equal to $N$ - $t$, where $t$ is the total number of devices that have transmitted up to the ($k$-1)-th time slot.
Carrying on with the computation of the transition probabilities this way, leads to unsatisfactory results which are difficult to interpret.

A better way to compute the probability of having $j$ devices transmitting at slot $k$, is to use the *stochastic matrix* of the Markov chain. If the probability of moving from state $i$ to $j$ in one time slot is $Pr(j|i) = P_{i,j}$, the stochastic matrix $P$ is given by using $P_{i,j}$ as the $i$-th row and $j$-th column element, e.g.

$$P = \begin{bmatrix} P_{0,0} & P_{0,S} & P_{0,2} & \dots & P_{0,j} & \dots & P_{0,N} \\ P_{S,0} & P_{S,S} & P_{S,2} & \dots & P_{S,j} & \dots & P_{S,N} \\ P_{2,0} & P_{2,S} & P_{2,2} & \dots & P_{2,j} & \dots & P_{2,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ P_{i,0} & P_{i,S} & P_{i,2} & \dots & P_{i,j} & \dots & P_{i,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ P_{N,0} & P_{N,S} & P_{N,2} & \dots & P_{N,j} & \dots & P_{N,N} \end{bmatrix}$$

Since S and N are absorbing states, $P_{S,j} = 0$ for j $\neq$ S and $P_{N,j} = 0$ for j $\neq$ N.
Moreover, all the possible transitions can only generate a non-decreasing sequence of states, hence $P_{i,j} = 0 \ \forall i > j$.
All the other elements of the matrix can be computed using formula 3.4:

$$P_{i,j} = \binom{N-i}{j} p^j (1-p)^{N-i-j} \tag{3.5}$$

Therefore the stochastic matrix becomes:

$$P = \begin{bmatrix} P_{0,0} & P_{0,S} & P_{0,2} & \dots & P_{0,j} & \dots & P_{0,N} \\ 0 & 1 & 0 & \dots & 0 & \dots & 0 \\ 0 & P_{2,S} & P_{2,2} & \dots & P_{2,j} & \dots & P_{2,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & P_{i,S} & 0 & \dots & P_{i,j} & \dots & P_{i,N} \\ \vdots & \vdots & \vdots & \ddots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 & \dots & 1 \end{bmatrix}$$

Let $x_0$ be the *initial state vector*, i.e. an $N \times 1$ vector that describes the probability distribution of starting at each of the N possible states.
To compute the probability of transitioning to state $j$ in **k** steps, it is now sufficient to multiply the initial state vector $x_0$ by the stochastic matrix raised to the k-th power, e.g.

$$P_k(j) = x_0 \cdot P^k \tag{3.6}$$

In our case, the system always starts in state 0, so we have

$$x_0 = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

which yields

$$P_k(j) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \cdot P^k = (P^k)_{0,j} \tag{3.7}$$

Calculating the k-th power of a matrix can be an intensive task from a computational point of view. To improve the complexity of the computation, rows and columns of P can be rearranged, moving the S row and the S column as penultimate, thus obtaining

$$P = \begin{bmatrix} P_{0,0} & P_{0,2} & P_{0,3} & \dots & P_{0,N-1} & P_{0,S} & P_{0,N} \\ & P_{2,2} & P_{2,3} & \dots & P_{2,N-1} & P_{2,S} & P_{2,N} \\ & & P_{3,3} & \dots & P_{3,N-1} & P_{3,S} & P_{3,N} \\ & & & \ddots & \vdots & \vdots & \vdots \\ & & & & P_{N-1,N-1} & P_{N-1,S} & P_{N-1,N} \\ & & & & & 1 & 0 \\ 0 & & & & & & 1 \end{bmatrix}$$

P is now an upper triangular matrix and this allows for faster computation of its powers in 3.7 .

# Chapter 4

# Complete Model

# Chapter 5

# Simulator

In order to obtain experimental results for the presented scenarios, a simulator was built using OMNeT++. This will allow us to reproduce the different scenarios with different values for the identified parameters.

## 5.1 Omnet++ and INET framework

OMNeT++ is an extensible, modular, component-based C++ simulation library and framework, primarily for building network simulators[1].
The INET Framework is an open-source model library for the OMNeT++ simulation environment. It provides protocols, agents and other predefined models for researchers and students working with communication networks. INET is especially useful when designing and validating new protocols, or exploring new or exotic scenarios[2].
OMNeT++ is a library and a framework, and can be used with the dedicated IDE. Not only it allows for development of the simulator itself, but also to export simulation results and to inspect simulation behaviour with a graphical user interface. Exploiting C++ compiler optimizations, it can achieve the lowest simulation duration possible. Networks are composed by modules; there are two types of modules: simple module and compound module (which can contain other modules itself). INET is an extension of OMNeT++, dedicated to recreating network simulation environments, with the capability of reproducing the activity of a wireless communication system across multiple nodes. It contains ready to use definitions and implementations of network related modules.

The choice was made to use the INET framework in order to avoid spending too much time on the coding side and therefore be able to focus more on other aspects such as the problem modeling and analysis.

[1] https://omnetpp.org/
[2] https://omnetpp.org/download-items/INET.html

## 5.2 Network architecture

The network based architecture is composed by an Array of Host modules, an Integrated visualizer (visualizer) and a UnitDiskRadioMedium (radioMedium);

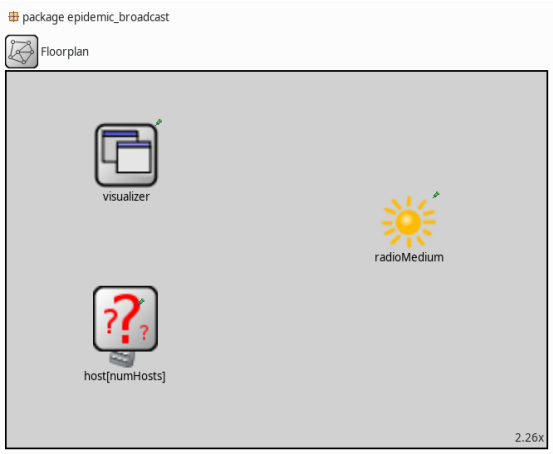



Figure 5.1: Floorplan.ned

- **UnitDiskRadioMedium** is a compound module provided by INET. This radio medium model provides a very simple but fast and predictable physical layer behavior. It must be used in conjunction with the `UnitDiskRadio` model. It can simulate the behaviour of the wireless communication channel with various levels of abstraction.

- **Integrated visualizer** is a compound module provided by INET. It's resposible for the visual representation of modules properties and events in the graphic user interface.

- **Host** is the compound module developer for representing a node in our network environment.

The `Host` module extends the NodeBase module defined by INET. This module contains the most basic infrastructure for network nodes that is not strictly communication protocol related.

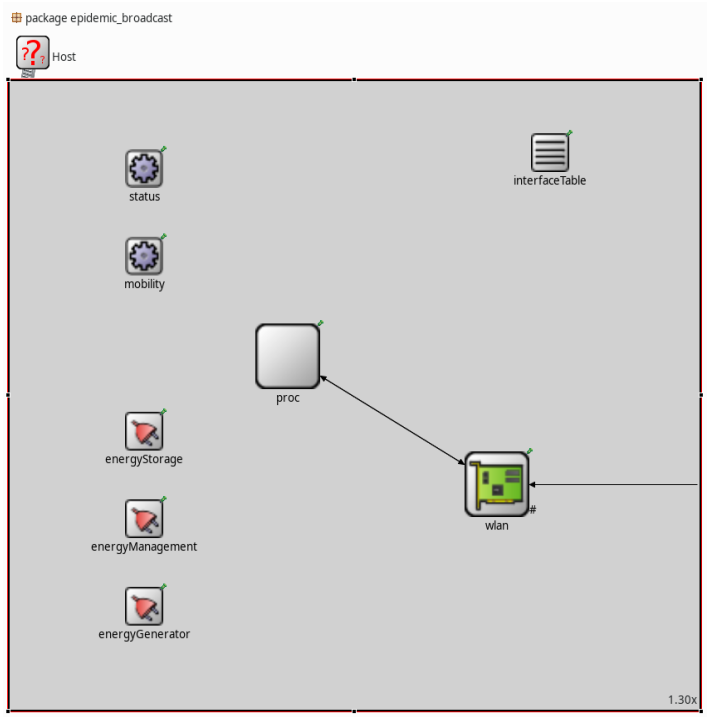



Figure 5.2: host.ned

- **Mobility** module privided by INET manage the position of the parent module (Host); it allow various types of movements, but we are going to use it only for the initial random placement of the nodes, then they will remain static.
- **interface Table** module is provided by INET and is required for correct operation of radioMedium module.
- **wlan** module is the wireless interface that allow nodes to communicate with each others. It's a compund module of type **AckingWirelessInterface**, which is the simplest wireless interface provided by INET.
- **status** module is provided by INET and is required to shutdown and restart network interfaces.
- **ProcUnit** is the processing unit, that implement the node behaviour when a message arrive. It's connected to the wlan module in order to receive and send messages through that.
- energyStorage, energyManagement and energyGenerator are modules inherited from NodeBase but they are not instantiated, because we don't need to model energy related behaviours.

The wlan module check every message for collisions,and drop broken messages instead of forwarding it to the processing unit. The processing unit realize behaviours of nodes;

it handle the broadcast message when received, and then retransmit it when the random variable give a positive result. Finally it shut down the network interface, prevent it to receive any message (or collision). The network interface is also turned off durign the RV extraction.

## 5.3 Parameters and Statistics

During the simulation this signals are emitted and this statistics are collected:

- The wlan module emit a signal every time a collision is detected; this signal is collected by the **packetDropIncorrectlyReceived** of the same module; we are intrested in the total (count) number of collisions detected by each node.
- The ProcUnit module emit 2 signal when initialized, hostX and hostY, collected rispectively by **hostXstat** and **hostYstat**; those are the coordinates of the parent node in the floorplan.
- The ProcUnit module emit timeCoverage singal, collected by the Flooplan module in the **timeCoverageStat** statistic; this is a vector, containing the number of the slot when each node has been covered; it's dimension represent the number of nodes covered at the end of the simulaton.
- The ProcUnit module emit coverage singal, collected by the Flooplan module in the **coverageStat** statistic; this sum also represent the number of nodes covered at the end of the simulaton.

Most significative parameters set up in the initialization file (floorplan.ini) are reported belaw:

## 5.4 Design Choises and Optimizations

The utilization of INET framework for simulator developement allow us to exploit a pre-builted model for wireless communications; for example, collision detection and statistic collection about it is already implemented in INET modules. During the developement we choise for every aspect the optimal level of abstraction for owr purpose, but it's possible to model other aspects just changing typse to the INET modules, or adding new ones. We voluntarily omitted phenomena like pathloss and node movement, and we restrict owr considerations to a discrete time scenario, but it will be very easy to model this complex things just changing few modules types and attributes.
INET modules have also pre-builted optimizaton structures, that become indispensable when the number of modules become large; in order to make the simulator ready for high complex scenarios we used an the **neighborCache** structure offered by the radioMedium module. This module is in charge of store the proximity information of every nodes, in order to speed up messages delivery. Setting the type of this module to

`GridNeighborCache` it's possible to reduce the time needed for a simulaton with more than 2000 devices, by a factor of 10; we also found that this particulare type of cache (with the right cellSize) is the best trade-off beetween speed and memory occupancy, for this type of workload[3].

[3]https://doc.omnetpp.org/inet/api-current/neddoc/inet.physicallayer.contract.packetlevel.INeighborCache.html

# Chapter 6

# Simulation

# Chapter 7

# Appendices

## Appendix A

Given a scenario with $N$ transmitter devices and a target device $T$ in reach of all the transmitters, let us define the probabilities $P_1(j, N)$ as the probability of $j$ devices out of $N$ transmitting at the same time during slot 1 and $P_i(j)$ as the probability of $j$ devices transmitting at the same time during slot $i$.
By specification, the successful reception of the message by device T happens if and only if **one** of the transmitters sends the message during the slot. Furthermore, the successful transmission of a device is "*a Bernoullian RV with success probability* p *on every slot, until it achieves success*"; therefore we can model $P_1(j, N)$ as follows:

$$\left.\begin{aligned}
&P_1(0,N) = (1-p)^N \\
&P_1(1,N) = Np(1-p)^{N-1} \\
&P_1(2,N) = \binom{N}{2}p^2(1-p)^{N-2} \\
&P_1(3,N) = \binom{N}{3}p^3(1-p)^{N-3} \\
&\dots \\
&P_1(N-1,N) = \binom{N}{N-1}p^{N-1}(1-p) \\
&P_1(N,N) = \binom{N}{N}p^N
\end{aligned}\right\} P_1(j,N) = \binom{N}{j}p^j(1-p)^{N-j}$$

As for $P_i(j)$ we can model the system as if it was in the first slot, with the total number of active devices now being equal to $N - t$, where $t$ is the number of devices that have transmitted in the ($i$-1)-th slot.

Therefore, for $i = 2$ we have:

$$
\begin{aligned}
P_2(0) &= P_1(0,N)P_1(0,N) + P_1(2,N)P_1(0,N-2) + ... + P_1(N-1,N)P_1(0,1) = \\
&= P_1(0,N)P_1(0,N) + \sum_{k=2}^{N-1} P_1(k,N)P_1(0,N-k) = \\
&= \sum_{k=0}^{N-1} P_1(k,N-k)P_1(0,N) - P_1(1,N)P_1(0,N-1) \\
P_2(1) &= P_1(0,N)P_1(1,N) + P_1(2,N)P_1(1,N-2) + ... + P_1(N-1,N)P_1(1,1) = \\
&= P_1(0,N)P_1(1,N) + \sum_{k=2}^{N-1} P_1(k,N)P_1(1,N-k) = \\
&= \sum_{k=0}^{N-1} P_1(k,N)P_1(1,N-k) - P_1(1,N)P_1(1,N-1) \\
P_2(2) &= P_1(0,N)P_1(2,N) + P_1(2,N)P_1(2,N-2) + ... + P_1(N-2,N)P_1(2,2) = \\
&= P_1(0,N)P_1(2,N) + \sum_{k=2}^{N-2} P_1(k,N)P_1(2,N-k) = \\
&= \sum_{k=0}^{N-2} P_1(k,N)P_1(2,N-k) - P_1(1,N)P_1(2,N-1) \\
P_2(3) &= ... = \sum_{k=0}^{N-3} P_1(k,N)P_1(3,N-k) - P_1(1,N)P_1(3,N-1) \\
&... \\
P_2(N-2) &= \sum_{k=0}^{2} P_1(k,N)P_1(N-2,N-k) - P_1(1,N)P_1(N-2,N-1) \\
P_2(N-1) &= \sum_{k=0}^{1} P_1(k,N)P_1(N-1,N-k) - P_1(1,N)P_1(N-1,N-1) \\
P_2(N) &= P_1(0,N)P_1(N,N)
\end{aligned}
$$

which has the general form:

$$
P_2(j) = \begin{cases} \sum_{k=0}^{N-1} P_1(k,N)P_1(0,N-k) - P_1(1,N)P_1(0,N-1) & \text{j} = 0 \\ \sum_{k=0}^{N-j} P_1(k,N)P_1(j,N-k) - P_1(1,N)P_1(j,N-1) & 0 < \text{j} < \text{N} \\ P_1(0,N)P_1(N,N) & \text{j} = \text{N} \end{cases} \tag{7.1}
$$

where the term with the minus sign is due to the fact that, if only one device transmitted during slot $i$, the target device T will have correctly received the message and therefore,

starting from slot $i+1$ onwards, it will not be listening for incoming messages any more but it will be itself transmitting instead.

# Appendix B

Given a generic convess shape as a floorplan, we can compute it'area ($A$); if we imagine to put a (puntiform) transmitter (trx) in a random point of the floorplan, it's easy to compute the probability for the trx to be located in a given point $x$ as:

$$P(x) = 1/A$$

Given that the trasmssion raduis in $r$, we can define a circle centered in x, as the transmission range of the first trx ($R$). If the trx in far enough from the border of the floorplan, all it's transmission range fall into the floorplan; in this case, the probability for another (random positioned) trx to be placed inside this transmission range is equal to:

$$P(x') = \int_R P(x) = \frac{2\pi r^2}{A}$$

If the firts trx is placed at a distance less than $r$ from one or more borders of the floorplan, the probaility to randomly place another trx in it's range is less than the previous case; this is because part of the area of the transmission range will fall out of the floorplan, and, by construction, trxs can't be placed there. Depending on the floorplan' shape, we can identify the partition of the shape where the transmission range will fall out of the shape itself, and we will do specific computatin for this area; In general, by the middle value theorem for integrals, and the total probability theorem, the probailiy for 2 transmitters (let $x$ and $y$ be their coordinates) randomly placed to be close enough to communicate it's:

$$\rho = P(|x - y| < r) = \frac{1}{A} \int_A P(x')$$

Since we know that $P(x')$ is not constant for all the area of the shape, this integral might be difficult to compute, even for a square floorplan; in any case it's value, it's constant for a given floorplan and a given $r$. Taking in to consideration only one trx, we can compute the probability density function for the number of trx in its transmission range, in function of the number of trx in the entire floorplan $C(N)$. Every time we put a trx in the floorplan, the probability that it falls in the transmission range of the highlighted one is $\rho$. This is true for every $N > 1$, because of indipendance.

$$C(N) = \left(\frac{1}{p}\right)^{(N-1)}$$

This is a Bernullian distribution, and we can use this result only taking one trx at time; in other words we can't use this distrubution for all the trx in a random configured floorplan, because they are not indipendent. The distrubution for one trx will be influenced by the position of others. Anyway, we can apply this distribution to many transmitters, each one taken from a different floorplan; in this way we can validate the indipendanceof different random configurations.

# Bibliography

[1] Frank Harary. *Graph theory.* Addison Wesley, 1969.