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
Yuri Mazzuoli
Rambod Rahmani

May 8, 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, in chapter 2, 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, in chapter 3, a graph-based modelling technique, commonly used in literature, is proposed and some simplified scenarios are analysed.

- In chapter 4 the development of the simulator is described and the results of its validation are presented.
- Chapter 5 concerns the full simulation and performance evaluation of the system.
- Lastly, in chapter 6 conclusions are drawn about the results of the study and some suggestion are given regarding further research that could possibly be done on the subject.

The entire codebase (both the Omnet++ files and the scripts used for data analysis) is available at `https://github.com/MPinna/epidemic_broadcast` .

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 considered point-like 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 single transmission slot obviously has an effect on the total broadcast time, 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 `C`;
- the number of collisions `X`: a collision is detected by a node whenever it receives a message from two or more transmitting nodes during the same slot;

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

More specifically:

- 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.
  R clearly has a great impact on all the performance metrics: the greater this 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.
  Realistic values for R have been taken from Bluetooth Low Energy standard and range from a minimum of 5 metres to a maximum of 20 metres.

- the *per-slot* transmission probability `p`: it is the success probability for the Bernoullian random variable associated to the transmission. As a probability, it can assume values between 0 and 1.
  The higher the transmission probability p, 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;

- the length of the longer side of the floorplan rectangle `L`: a bigger floorplan area, all else being equal, will require a longer time to be covered entirely by the broadcast;

- the *aspect ratio* of the floorplan rectangle `a`, defined as the ratio between the longer side L and the shorter side W.
  A very long and very narrow floorplan will probably cause less collisions than a square one with the same area and the same number of hosts, as the average number of users in the transmission range of a random user decreases; on the other hand, the performance of this type of scenario will be highly influenced by the position of the starting node.
  Because of the radial symmetry of the transmission phenomenon, there is no actual need to consider values for `a` lower than 1.

- the number of users dropped on the floorplan `N`;

- the users population density `d`: it is defined as the number of users per square meter;

For this study, the two main parameters are `R` and `p`. Full sweeps for different values of both are presented.
`a` and `L` were kept constant and equal to 1 and $100m$ respectively (i.e. the floorplan is always a square with a side of 100 metres). `N` was also kept constant and equal to 100. The resulting user density is thus $1/m^2$.

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.
As the broadcast progresses and the message is transmitted among the nodes, the system evolves through different states where each node could be either listening for an incoming message, transmitting it, or stopped (has alreayd trasmitted the message).
The different nodes behaviours were modelled by means of three different states:

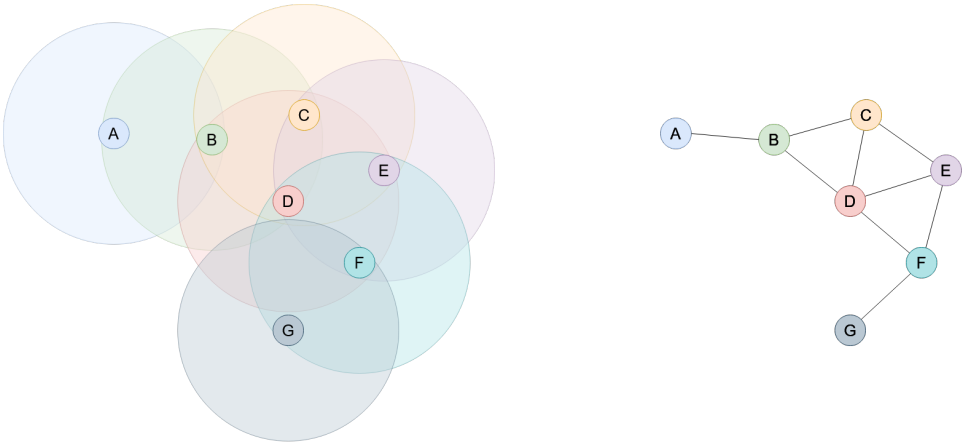
- `listening`: the node has not received the broadcast message yet and therefore it is still listening for incoming messages from other nodes;

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

- `sleeping`: the node has already received and retransmitted the broadcast message; once a node enters the `sleeping` state, it remains 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 pairs of nodes in reach of each other. 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 all the vertices adjacent to a vertex v is called the *neighbourhood of* v.
During the broadcast process, a node can only receive from and transmit to its neighbourhood.
Once a node has transmitted the message and has gone into the `sleeping` state, from the point of view of the graph model, 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* [**hararygraph**].

Modelling the system with graphs also allows for the simple computation of some of their properties, such as the *eccentricity*; in this way, it is possible to find theoretical lower bounds for some of the metrics and can also be useful for validating the simulator or for gaining further insight about the results.
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:

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

Another observation to be made is that the existence of an edge between two nodes

only depends on the ratio between their radius and their distance. This implies that 100 nodes with a radius of 15 m dropped in a 100 m x 100 m floorplan will produce the exact same graph as 100 nodes having a radius of 1,5 m dropped in a 10 m x 10 m floorplan, if all their coordinates are just scaled by a factor of 10.

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 3.2. Each user only has two neighbours, except for the outer ones (nodes A and E) that only have one. Assuming node A is the starting user with the broadcast message, it is the only node in `transmitting` state while all the other nodes are initially in `listening` state.

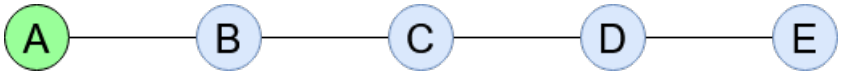


Figure 3.2: Graph of the Single queue configuration

It is clear that, with this configuration, each listening node has a maximum of one active node in its neighbourhood at any time 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 Bernoullian RV with success probability p. The probability of the active node transmitting after exactly k slots is a geometric distribution:

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

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

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

3.2.2 Star configuration with one active node (star 1-to-5)

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 its entire 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.

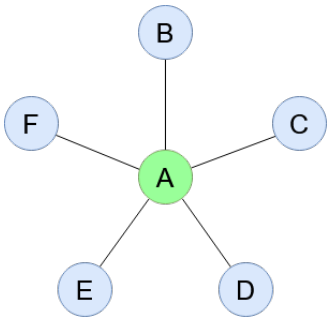


Figure 3.3: Graph of the star configuration with one active node

In this case, the average total broadcast time $E[T]$ is simply equal to the mean value of the geometric distribution, $\frac{1}{p}$, since the central node is placed at one hop of distance from all the remaining nodes.

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 $\mathtt{T}$, its expected value would just be $\frac{2}{p}$, 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 nodes active but one (star 5-to-1)

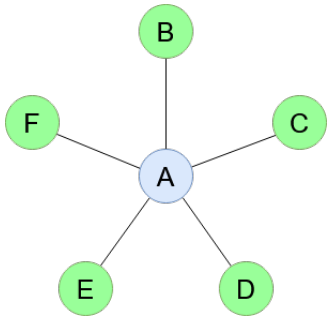


Figure 3.4: Graph of the star configuration with all nodes active but one

This configuration can be seen as the complement of the previous one: all the nodes on the rays of the star are active and trying to transmit the message to the center node. This is the first scenario where collisions might occur and with a non-zero probability 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:

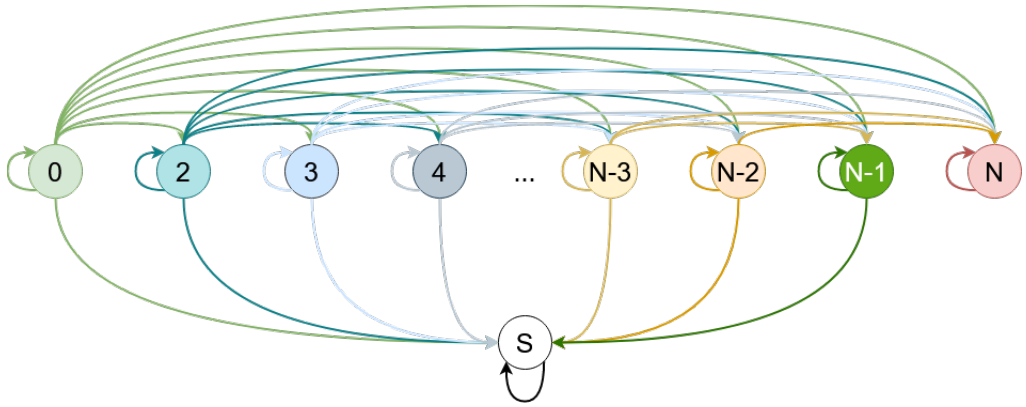


Figure 3.5: Discrete-time Markov chain for a scenario with N active devices

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 `sleeping` devices, namely devices that have already sent the message and have stopped. The number of `sleeping` devices was chosen over the number of `transmitting` devices as the former only allows for non-decreasing sequences of states, since a device cannot become active again once it enters the `sleeping` state.
State S represents the successful transmission state, which the system transitions to when only one device has transmitted the message during the previous time 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 Figure 3.5, where the only outgoing arrow from these states goes back to the state itself).
If the system transitions to state N, it stays in it indefinitely since all the devices are `sleeping` and they cannot become active again. This implies that there will never be total coverage since the target device will forever stay in the `listening` state without ever receiving the message. On the other hand, state S, despite 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 and total coverage has been reached.
Another interesting observation concerns state $N-1$: if the system reaches this state, as there is one and only one remaining node which can transmit, it is 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 given by:

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

which is a binomial distribution given by the sum of the N Bernoulli distributions.

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 sleeping 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 and there are no transitions between states that differ by one, hence
$P_{i,j} = 0 \ \forall \ i > j$ and $\forall \ j = i+1$.

All the other elements of the matrix can be computed using the following formula:

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

and $P_{i,S}$ is given by:

$$P_{i,S} = \binom{N-i}{1} p(1-p)^{N-i-1} \tag{3.6}$$

which is just a particular case of (3.5) when $j = i + 1$.
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.

$$P_k(j) = (x_0 \cdot P^k)_j \tag{3.7}$$

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

$$P_k(j) = \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} \cdot P^k = (P^k)_{0,j} \tag{3.8}$$

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

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

P is now an upper triangular matrix and this allows for faster computation of its powers in 3.8.

Chapter 4

Simulator

In order to obtain experimental results for the presented scenarios, a simulator was built using OMNeT++ with the support of the INET framework. This allowed to reproduce the different scenarios presented in the previous chapters with different values for the identified parameters.

4.1 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. By taking advantage of the C++ compiler optimizations, it can achieve significant speed-ups for the simulations times.

Networks are composed by modules and there are two types of modules: simple modules and compound modules (which can contain other modules themselves). INET, on the other hand, 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 INET framework was chosen as it provides, among many other things, ready-made

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

modules for wireless networks, allowing to reduce the time required for the development of the simulator.

4.2 Network architecture

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

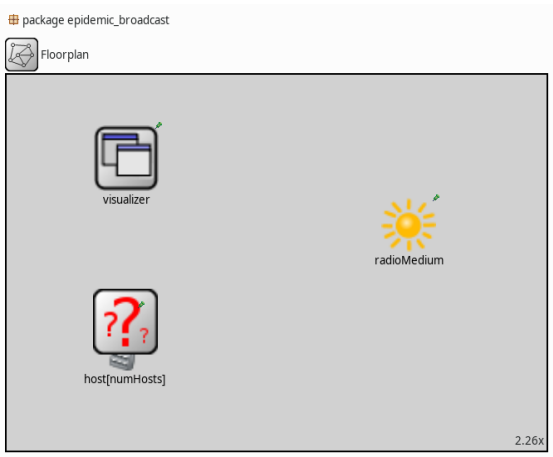


Figure 4.1: Floorplan.ned

- **UnitDiskRadioMedium** is a compound module provided by INET. This radio medium model provides a very simple but fast and predictable physical layer behaviour. 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 is responsible for the visual representation of modules properties and events in the graphic user interface.

- **Host** is the compound module developed to represent a node in the 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. The following diagram shows usage relationships between types:

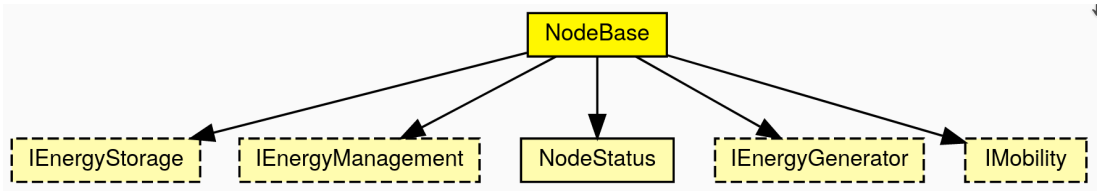


Figure 4.2: NodeBase Diagram.

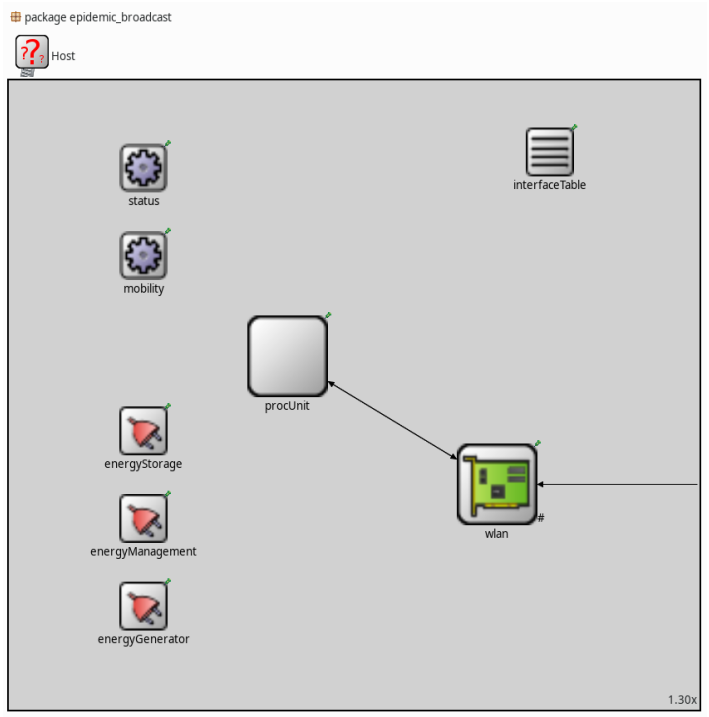


Figure 4.3: host.ned

- the `mobility` module provided by INET manages the position of the parent module `Host`; it allows for various types of movements, but in this study was only used for the initial random placement of the nodes. After being placed, all nodes are stationary for the whole duration of the simulation.

- the `interfaceTable` module is provided by INET and is required for correct operation of the radioMedium module.

- the `wlan` module is the wireless interface that allows nodes to communicate with each other. It is an `AckingWirelessInterface` compound module, which is the simplest wireless interface provided by INET.

- the `status` module is provided by INET as well and is required to shut down and restart network interfaces.

- the `procUnit` module is the custom made processing unit, that implements the node behaviour when a message arrives. It is connected to the `wlan` module in order to be able to receive and send messages.

- `energyStorage`, `energyManagement` and `energyGenerator` are modules inherited from `NodeBase` but they are not instantiated as there was no need to model energy-related behaviours.

The `wlan` module is in charge of checking each and every message for collisions and dropping broken packets instead of forwarding them to the processing unit. The processing unit `ProcUnit` implements the behaviour of the nodes; it handles the broadcast message when received, and then its retransmission when the random variable extraction results in a success. Finally it shuts down the network interface, preventing it from receiving any messages or detecting unwanted collisions. The network interface is turned off for the entire duration of the RV extractions.

4.3 Parameters and statistics

During the simulation, signals are used to collect statistics. They are all collected by the `Floorplan` module:

- the `wlan` module emits a signal every time a collision is detected; this signal is collected by the `packetDropIncorrectlyReceived` statistic of the same module; we are interested in the total number of collisions detected by each node.

- the `ProcUnit` module emits two signals when initialized: `hostX` and `hostY`. These are the coordinates of the parent node in the floorplan and they are collected, respectively, by the `hostXstat` and `hostYstat` statistics.

- the `ProcUnit` module emits the `timeCoverage` signal as well, collected in the `timeCoverageStat` statistic; this is a vector containing, for each node that received the broadcast message, the number of the time slot where the broadcast message was received; at the end of the simulation, its size represents the number of covered nodes.

The most significant parameters set up in the initialization file (`floorplan.ini`) are reported below:

- `Floorplan.host[*].procUnit.p = ${p = 0.1..1 step 0.1}`

- `Floorplan.host[*].[...].communicationRange = ${R = 1..20 step 1}m`

Moreover, the X and Y coordinates for each host are set to a uniformly distributed random value by the INET Framework, by default.

4.4 Design Choices and Optimizations

Using the INET framework for the development of the simulator allowed for the use of pre-built modules for modelling wireless communications; for example, collision detection and statistics collection are already implemented by INET modules. During the development, a level of abstraction suitable for the purposes was chosen, but it is possible to model other aspects by just changing the types of the INET modules used, or by adding new ones. Phenomena such as path loss, node movement etc. were intentionally not taken into account and considerations were restricted to a discrete time scenario. However, modelling continuous time scenarios can be easily done by changing few INET modules types and attributes.
INET modules also have pre-built optimization structures, that become indispensable when the number of hosts becomes larger; in order to make the simulator ready for highly complex scenarios, the `neighborCache` structure offered by the `radioMedium` module was used. This module is in charge of storing proximity information of each and every node, in order to speed up the simulation of a single transmission. By setting the type of this module to `GridNeighborCache`, it is possible to reduce the time needed for a simulation with more than 2000 devices dropped on the floorplan, by a factor of 10; it was observed that this type of cache (with the right value for the `cellSize` parameter) is the best trade-off between speed and memory occupancy, for this type of workload[3].

4.5 Validation

To ensure the correctness of the simulator and the meaningfulness of the results, the simulator has been validated by means of two simplified scenarios, namely the single queue configuration and the star configuration, which are discussed in 3.2.1 and 3.2.3 respectively.

4.5.1 Single queue validation

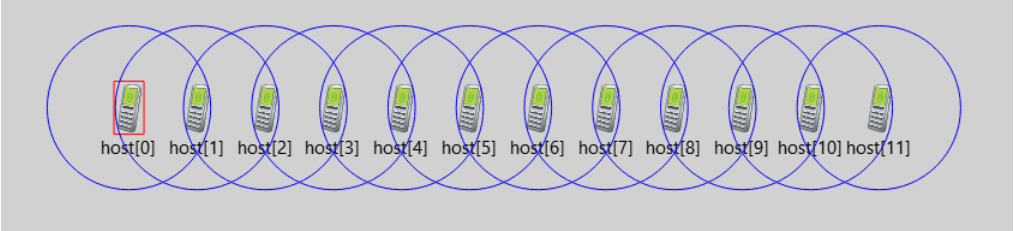


Figure 4.4: Validation configuration with 12 hosts placed on a line

In this configuration, host[0] always broadcasts during the first slot and then the message travels along the queue, with a total of 10 hops. On every hop, the per-slot probability

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

of successful transmission is p, which implies an average number of attempts equal to $\frac{1}{p}$. Therefore, the expected coverage time is

$$E[T] = 1 + 10 \cdot \frac{1}{p} \tag{4.1}$$

Validation was performed with 9 different configurations, one for each value of p ranging from 0.1 to 0.9, with 200 repetitions each. The following results were obtained:

p	Expected value	Experimental value (mean, interval for 95% confidence)
0.1	101.0	**99.68**, 95.55, 103.81
0.2	51.0	**49.9**, 47.93, 51.86
0.3	34.33	**34.17**, 32.88, 35.46
0.4	26.0	**25.79**, 24.93, 26.65
0.5	21.0	**20.84**, 20.24, 21.44
0.6	17.67	**17.42**, 17.0, 17.84
0.7	15.29	**15.2**, 14.87, 15.52
0.8	13.5	**13.48**, 13.25, 13.72
0.9	12.11	**12.14**, 11.97, 12.3

As can be seen in the table above, the experimental results are consistent with the theoretical expected value, with a 95% confidence level.

4.5.2 Star 5-to-1 validation

In this configuration, host[0] is the target to be reached by the broadcast while all the others already have the message and try to broadcast at every slot with probability p.

This system can be modelled by a discrete-time Markov chain, as explained in 3.2.3. The probability of the system to be in the i-th state during the j-th slot is given by taking the j-th power of the stochastic matrix and taking the i-th element of the first row.

Validation was performed with 4 different configurations, one for each value of p ranging from 0.2 to 0.8 with steps of 0.2, with 1000 repetitions each.
For each configuration, data about the state of the system was recorded and statistics were computed, yielding experimental probabilities. These probabilities were then compared with theoretical predictions, obtained by taking

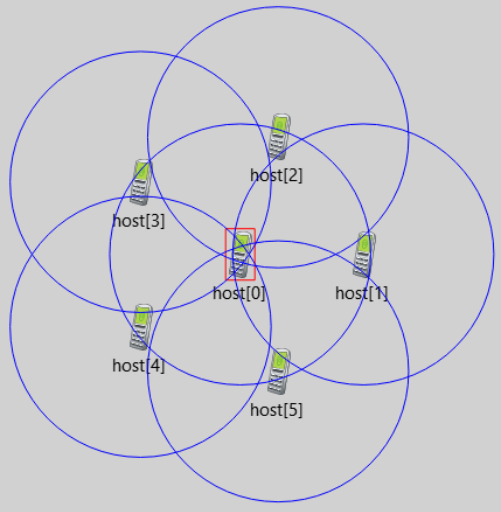


Figure 4.5: Validation configuration with 5 hosts placed on a star and a target in the middle

powers of the appropriate stochastic matrix.
All the experimental results (solid coloured lines in figs. 4.6 and 4.7) proved to be consistent with theoretical computations (black dashed lines).

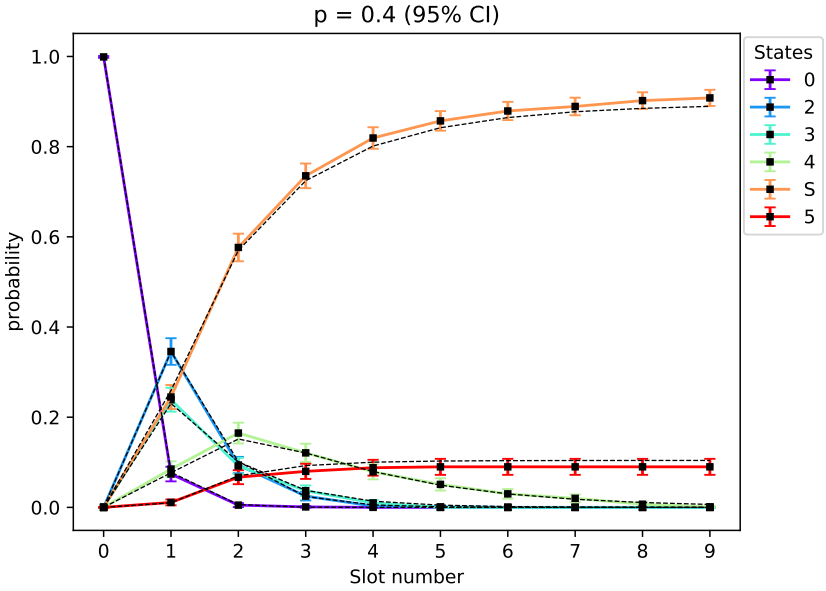


Figure 4.6: Validation data for p = 0.4

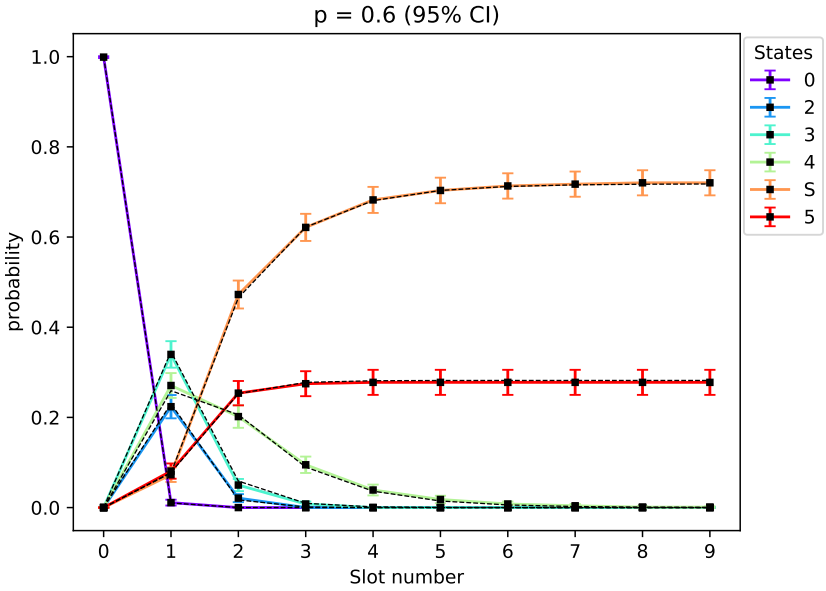


Figure 4.7: Validation data for p = 0.6

Chapter 5

Simulation

As previously described, the scenario considered was the following:

- a 100m x 100m floorplan, with the transmission radius ranging from 1 m to 19 m, with a step of 1 m, and a Bernoullian RV with success probability p ranging from 0.1 to 0.9 with steps of 0.1;

Methodological Note

In order to analyse all the possible combinations for p and R, it is necessary to explore the space generated by the cartesian product of the possible values for each parameter. Every combination identifies a scenario, performance indexes need to be computed; in the model every index is a random variable for which the mean value is sought, with appropriate confidence interval and confidence level (at least 95%). There were not enough details or prior knowledge on the topic to identify the probability distribution of those variables or even to know if they were symmetric.
Nevertheless, some *a priori* considerations could be made about their boundedness and therefore variance: all the indexes are clearly bounded from below, being non-negative. In addition:

- **broadcast time** is bounded from below as the broadcast cannot take less than one slot. Moreover, the worst case scenario is given by a single queue configuration with 100 hosts (see 3.2.1); the probability distribution for the broadcast time in this type of scenario is the same geometric distribution as the one for a single transmission, scaled up by a factor equal to the number of hops required. This distribution bounds the broadcast time from above and guarantees a finite variance for it.
- the **percentage of covered users** is bounded from above by 1 (100%).
- the **number of collisions**, in the case of a floorplan with 100 hosts, is bounded from above by the value 2305. The formula for the upper bound can be found in Appendix B, together with its derivation.

This knowledge allowed for the computation of confidence intervals by means of algorithms that do not require to know the distribution of the samples but only the finiteness of the variance.
For each of the scenarios, the floorplan coverage, the broadcast duration and the number of collisions are measured and plotted in graphs in order to better understand and interpret the behaviour of the system.
In order to achieve statistically significant results with a minimum accuracy of 95%, each scenario configuration – with the same parameters – was run 200 times; then, mean and median values of the performance indexes were computed, along with their confidence intervals.
The analysis of the results showed that mean and median values tend to converge for all the KPIs.

5.1 Coverage

Figure 5.1 shows the floorplan coverage achieved as a function of the success probability p, for different values of the transmission radius R.
As can be observed, the coverage is maximum for lower values of p; this is due to the fact that a low transmission probability minimizes the number of collisions, increasing the number of correct transmissions. For values of $R < 8$ we observe an almost linear behaviour: collisions are almost absent with such a small transmission radius, since the order of the connected component of the graph is quite low; therefore, variations of the value of p do not play a fundamental role as far as total coverage is concerned. For $R \geq 8$ the graph starts getting more and more connected and the achievable coverage decreases with the increase of p, but quite slowly and also it does not degenerate to 0; instead it reaches a value between 10% and 40%.

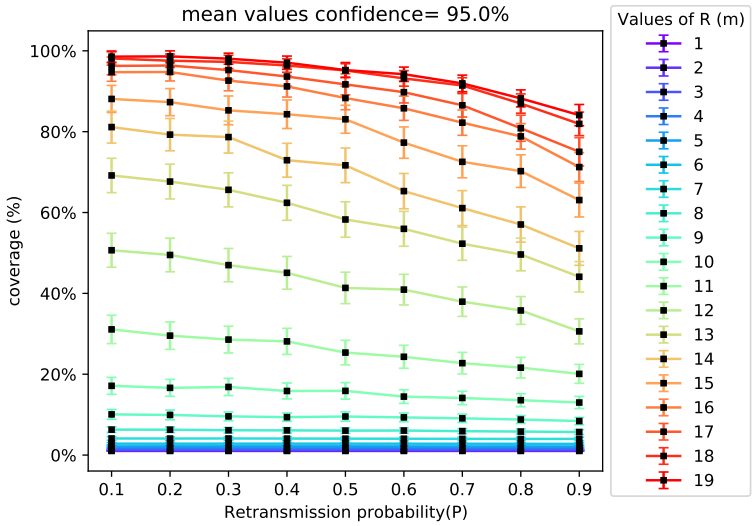


Figure 5.1: Floorplan coverage as a function of p for different values of R

For values of R between 11 and 15, the number of collisions has a strong effect on the coverage percentage; this effect tends to decrease for larger values of R; we know as a matter of fact that for $R \to L\sqrt{2}$ the coverage tends to 100%, independently of any other parameter.

Figure 5.2 shows the floorplan coverage achieved as a function of the transmission radius R, for different values of the success probability p. As expected, accordingly with the previous plot, the final coverage of the flooplan is deeply influenced by the transmission range R.

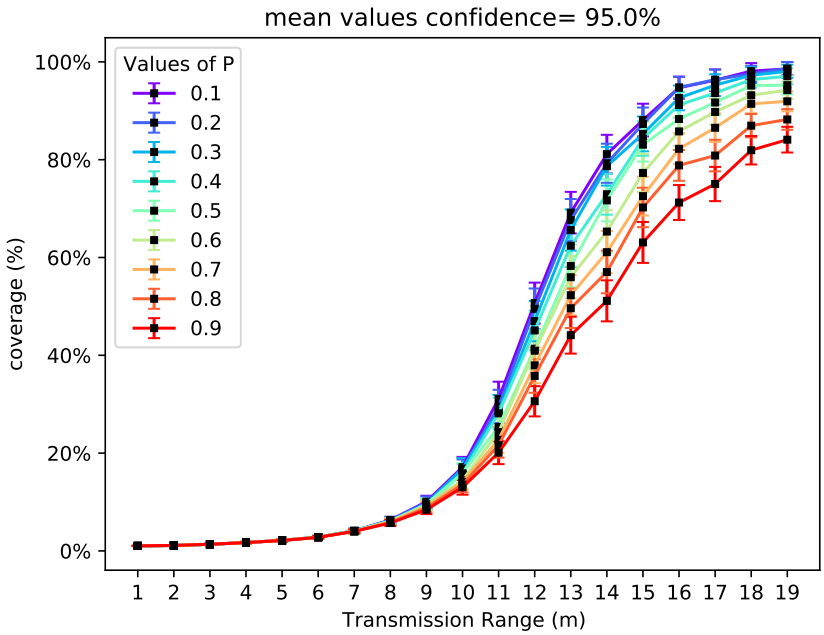


Figure 5.2: Floorplan coverage as a function of R for different values of p

In this second plot we can observe that the coverage increases rapidly with the transmission range, up to about $R = 13$ m. When the transmission range becomes large (> 16m) there are diminishing returns, as can be seen by the decreasing steepness of the curves, and the coverage tends to a value of 80-100%, depending on the value of p. The coverage function would appear to be a sigmoid, at least for low values of R.
A more in-depth analysis is carried out in 5.4.1.

5.2 Collisions

Figure 5.3 shows the mean number of collisions as function of p, for different values of R.
The values seem to have a linear behaviour for low values of p and then start stabilizing as p approaches one.

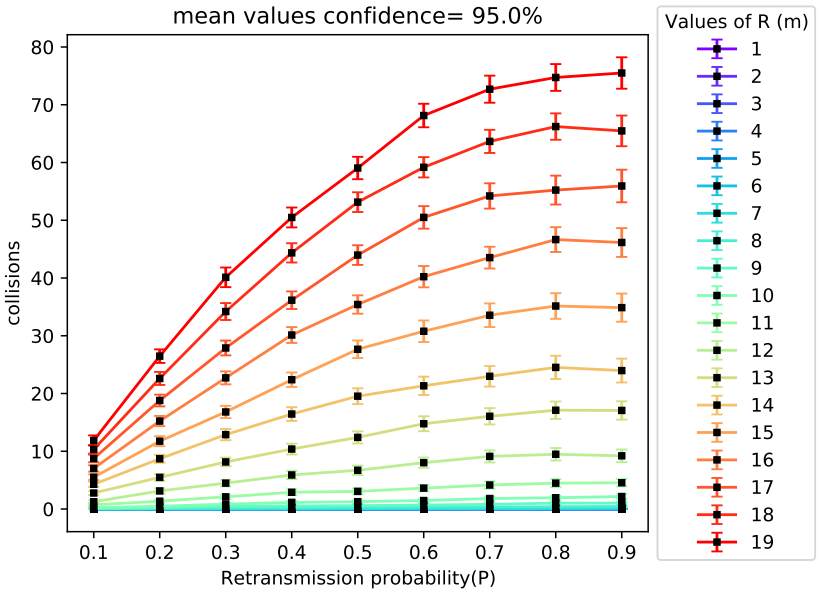


Figure 5.3: Mean value of the number of collisions as function of R for different values of p

The dual plot shown in fig. 5.4 provides further insight: low values for R cause no collisions at all, as can also be seen in 5.3 where the lines for $R \leq 8$ all lie on the horizontal axis, regardless of the value of p.
For bigger values of the radius, the number of collisions grows in a seemingly linear fashion dependent on the value of R, with a slope increasing on the increase of p.
The lines tend to coincide for higher values of p, as it also shows the flatness of the curves in the rightmost part of 5.3.
It would incorrect to assert a general positive correlation between the radius and the total number of collisions: as a matter of fact, the number of collisions has to tend to 0 for $R \to \sqrt{2}L$. This is because the more hosts receive the message from host 0, the more of them will be transmitting during the second slot and the less of them will be listening and able to detect a collision.

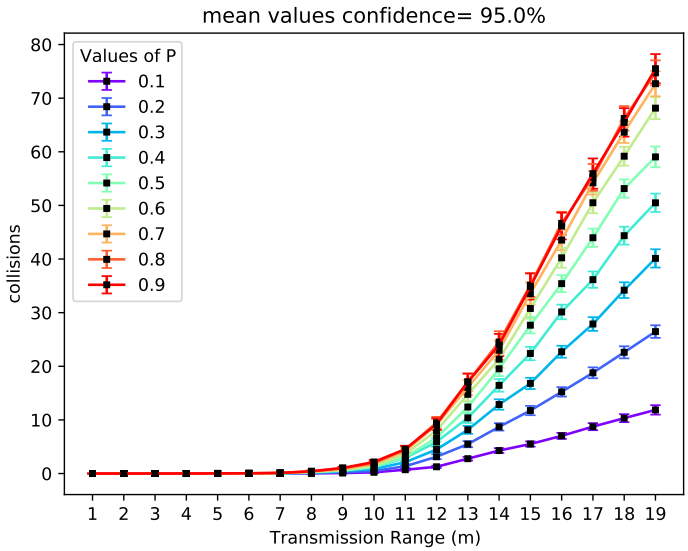


Figure 5.4: Mean value of the number of collisions as function of R for different values of p

5.3 Duration

The following figure shows the plot of the duration of the simulation (in slots) as a function of p, for different values of R.

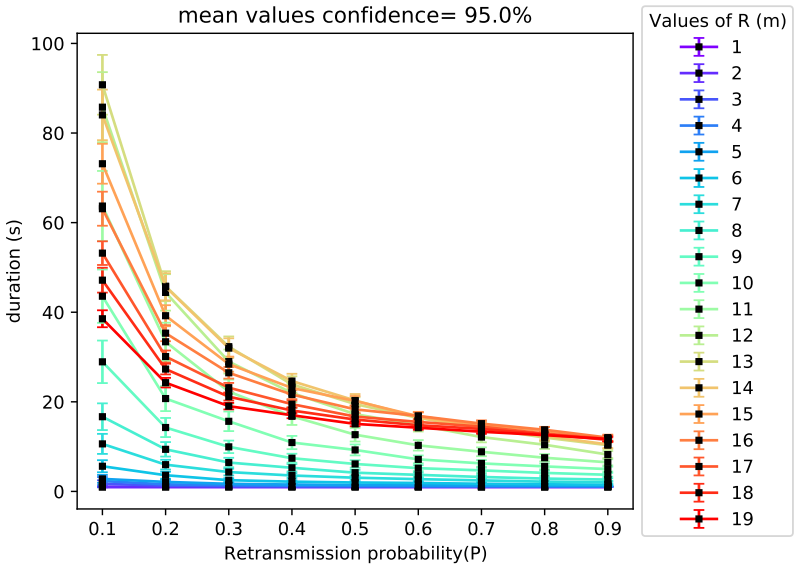


Figure 5.5: Simulation duration as a function of p for different values of R

The duration of the broadcast increases for lower values of p; this behaviour can be explained as a consequence of two main factors:

- the probability of retransmission is low, hence nodes will spend more time slots waiting, without actually transmitting;
- with a smaller number of collisions, fewer paths are "cut off" while the message is travelling along them; this causes the message to stay around for a longer time.

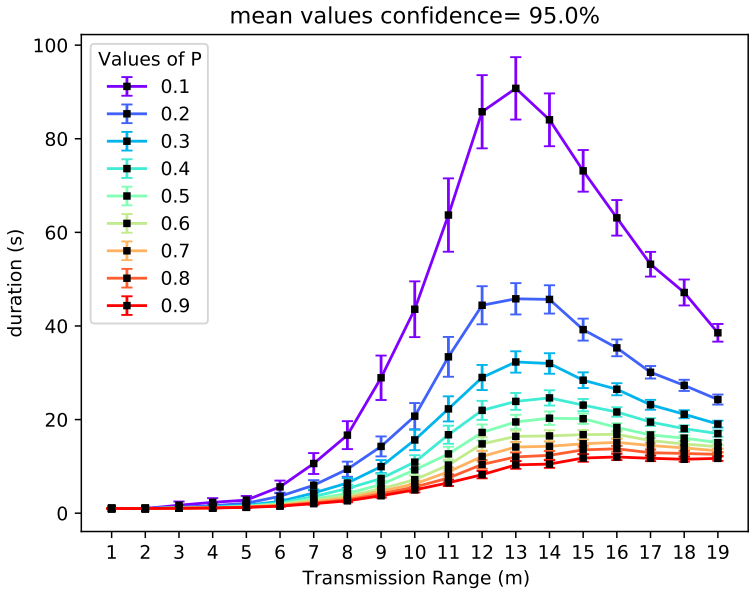


Figure 5.6: Simulation duration as a function of R for different values of p

The plot in fig. 5.6 is the dual of the one shown in 5.5 and shows the duration of the simulation (in slots) as a function of R, for different values of p.
The duration of the simulation tends to increase with the transmission range, reaching a peak around 13 m; the maximum value depends on the probability of retransmission (p) and the distance between the peaks would suggest this dependency to be an inverse proportionality; this would also be consistent with the trend exhibited by the curves in fig. 5.5, where the topmost ones resemble hyperbolas.
The bottom ones are much flatter, since very low values of R often do not even allow the starting node to be connected to any others. This supposedly inverse proportionality with respect to p is an interesting result and can be explained by observing that the same inverse proportionality is found in formula 3.3.
Further observation about this and relationship with some graph properties are discussed in 5.4.2.

Not all the curves in 5.5 exhibit a hyperbolic trend: very low values of R cause them to have a degenerate trend, while for high values of R the deviation from the hyperbola is likely to be due to the increasing influence of the collisions.

5.4 Comparison between KPIs and graph properties

Some of the results shown in the previous section can be better understood when compared with some properties of the graphs associated with the configurations of the hosts. In particular, three properties of the graphs have been studied:

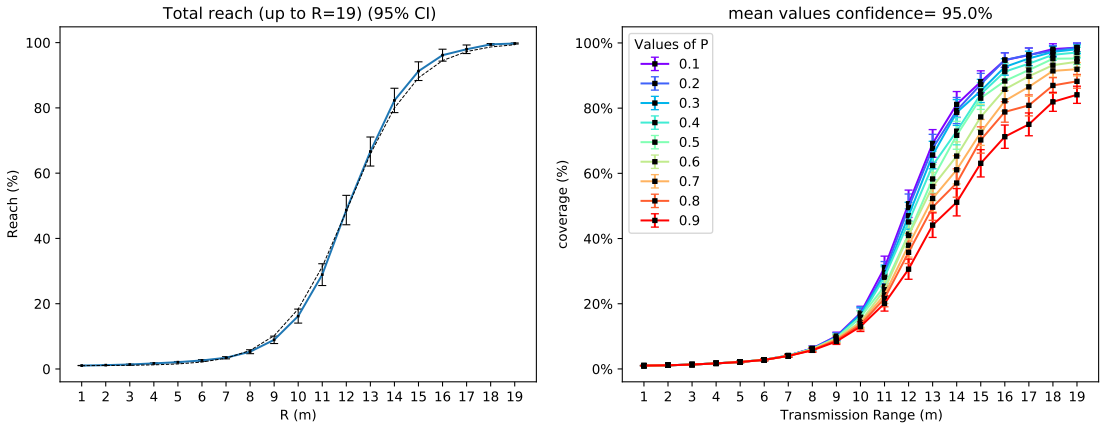
- the `reach`, defined as the number of nodes that can be reached from host (vertex) 0. More formally, let H be the connected component containing host 0 of the initial graph G, i.e. the induced subgraph in which any vertex is connected to vertex 0. The `reach` is the *order* of H.

- the `eccentricity`, already defined in 3.1

- the number of `safe nodes`, which has been defined for simulations with $p = 1$ as the number of nodes that correctly receive the message. Systems with $p = 1$ are completely deterministic and therefore the number of *safe nodes* only depends on the topology of the network.

5.4.1 Coverage, reach and safe nodes

In 5.1 results about the coverage were shown and it was hypothesized that this index, as function of R, might have a sigmoidal trend.
Let us consider the limit case where $p \to 0$.
As p approaches 0, during each slot it becomes less and less likely for an active node to transmit the message. On the other hand, smaller values of p lower the probability of collisions even more.
For values of p really close to 0, it is easy to imagine that there will be basically no collisions; at the same time, every node belonging to the connected component of node 0 will sooner or later receive the message with a probability of 1.



a) b)

Figure 5.7

Figure 5.7 a) shows the average reach from node 0, as function of R, in a graph with 100 nodes and figure 5.7 b) the experimental data for the coverage, for convenience of comparison. The plot in 5.7 a) plateaus at a value of 100 for values of $R \geq 24$ metres.
Various sigmoid functions have been tested to fit the experimental values for the *reach*, and the most appropriate one was found to be

$$S(R) = \frac{1}{N} + \frac{N-1}{N} \cdot \frac{1 + tanh(b(R-a))}{2} \tag{5.1}$$

where N is the number of nodes in the graph, a controls to the abscissa of the inflection point and b controls how steep the increase is.
The correct value for a can be found examining the data in fig. 5.7 and observing that they show an inflection point for $R = 12$ m. With regard to parameter b, trial and error showed that its value lies between 0.36 and 0.37. In fact, a value equal to $e^{-1} \approx 0.36788$ fits the data very well.
A closed form for the *reach*, and therefore for total coverage when $p \to 0$, in a square floorplan with $L = 100$ would then be

$$C_0(R) = \frac{1}{100} + \frac{99}{100} \cdot \frac{1 + tanh(e^{-1}(R-12))}{2} \tag{5.2}$$

This formula, however, models the coverage for a fixed values of `L` and N, as mentioned above.
A more complete closed form, that also takes in account the length of the side of the

floorplan and the number of hosts, might be the following:

$$C_0(R, L, N) = \frac{1}{N} + \frac{N-1}{N} \cdot \frac{1 + tanh(e^{-1}(10\sqrt{N}\frac{R}{L} - 12))}{2} \tag{5.3}$$

The presence of the quantity $\frac{R}{L}$ follows from the observation that the overall connectivity of a graph should only depend on the ratio between the transmission radius and the coordinates of the hosts (see section 3.1).
The rationale behind the $\sqrt{N}$ term is that the average number of nodes per unit area goes with the **square** of N, and supposedly also does the average number of nodes per transmission circle.
Another way to visualize this is to imagine the entire graph in a LxL floorplan, divide the floorplan in four $\frac{L}{2}$x$\frac{L}{2}$ squares and take the graph that lies in one of these squares: on average it will have a fourth of the initial number of nodes but its level of connectivity should presumably be the same as the original graph.
The value for a, as mentioned above, was found by observing experimental data and it is, almost certainly, not an exact value.
Lastly, coverage is also likely to depend on the shape of the floorplan itself, which is not taken in account in the formula. Further research is needed

Let us now consider the limit case where $p \to 1$.
The closer p gets to 1, the more the system approaches a deterministic behaviour where each active host broadcasts the message as soon as it receives it.
When $p = 1$ the number of node that correctly receive the message during the broadcast can be computed by knowing only the topology of the network.

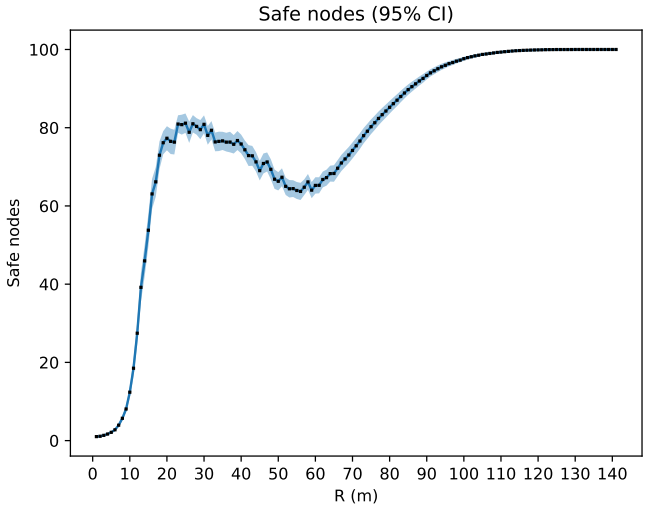


Figure 5.8: Average number of safe nodes of a graph as function of R

Figure 5.8 shows the average number of safe nodes during a broadcast starting from node 0, as function of R, in a graph with 100 nodes.
It is clear that this function does not have a sigmoidal trend, except for low values of R, up to about 25 metres, where the number of safe nodes reaches about 80. This is consistent with the trend of the curve for $p = 0.9$ in Figure 5.2 shown in the previous section.
For $R > 25$, increasing the radius yields negative returns: the average number of safe nodes decreases until R reaches about 55 metres. This is likely to be due to the fact that the increase in the number of collision takes over on the increase of reachable nodes. This is consistent with the hypothesis made in Chapter 2 when analysing the impact of R on the KPIs.
For $R > 55$ the number of safe nodes starts increasing again, probably because host 0 starts having more and more neighbours when the broadcast begins (and none of them detects a collision as host 0 is the only one transmitting during the first slot).
The quantity asymptotes to the total number of nodes and for $R > \sqrt{2} \cdot L$ it is identically equal to it since, wherever the starting node would happen to be placed, it would cover the whole area during its first transmission slot.

These considerations show that the coverage as function of R, although it might resemble a sigmoid for different values of p (and could probably be approximated well enough by it for realistic value of the transmission range), is probably more complex than that; for a certain range of values for R, the increase in the number of collision starts having a heavier role in the number of reached hosts.
A closed formula for the number of safe nodes when $p = 1$ has not been found yet but, assuming it is $C_1(R, L, N)$, then a reasonable form for the coverage function could be the following:

$$C(R, L, N, p) = (1 - p) \cdot C_0(R, L, N) + p \cdot C_1(R, L, N) \tag{5.4}$$

5.4.2 Broadcast time and eccentricity

In 3.1 the eccentricity ϵ of a graph was introduced, defined as the distance from node 0 to the furthest connected node, let it be u.
If we imagine the message spreading from host 0 outwards along the various paths of the graph, it is easy to guess that, on average, the path that will take the longest to be covered will be the one that leads to node u and this path will have, by definition, a number of edges equal to the eccentricity.

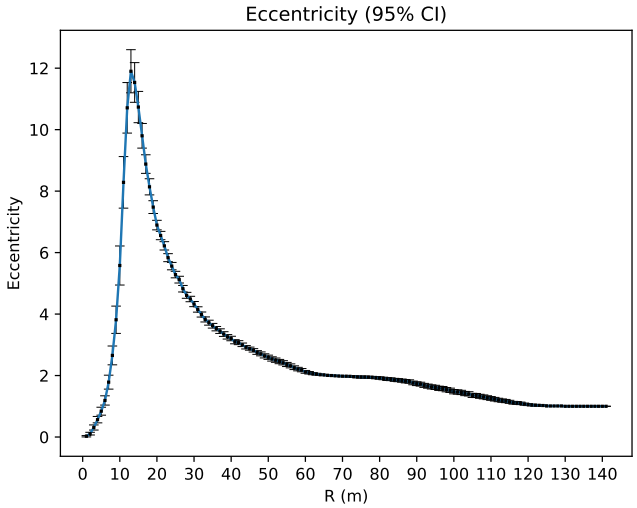
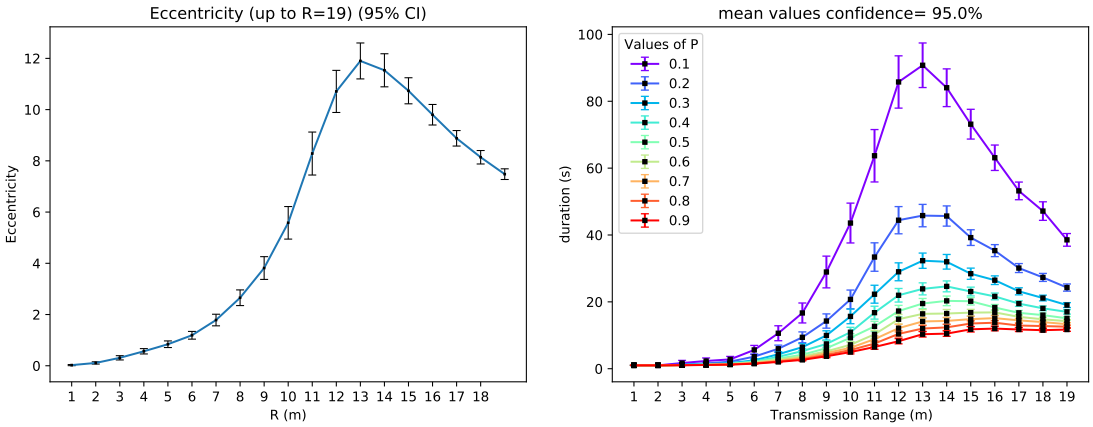


Figure 5.9: Average number of safe nodes of a graph as function of R

Figure 5.9 shows the average eccentricity of a graph with 100 nodes, as function of R. The function has a maximum around $R = 13$ m and then starts slowly decaying until it reaches 1 as R gets closer to $\sqrt{2}L$; this makes complete sense since, for high values of R, node 0 is quite likely to be connected to all nodes in the floorplan.
The peak at $R = 13$ m has the following physical interpretation: for values of R between 0 m and 13 m, increasing the radius increases the order of the connected component of the graph containing node 0; by further increasing the radius, more edges start to appear in the graph creating shorter paths between nodes that were already connected.
Another interesting observation concerns the behaviour of the function around $R = 70$ m: the eccentricity seems to stabilize at a value of 2, with very narrow confidence intervals, for values of R close to $\frac{1}{2}\sqrt{2}L$.
Fig. 5.10 shows the previous function up to $R = 19$ m one one side, and again the experimental data for the duration, for convenience of comparison.



a) Eccentricity as function of R b) Broadcast time as function R

Figure 5.10

The curves on the right clearly resemble the one on the left, minus a scaling factor, especially for lower values of p.
As the value of p approaches one, the curves tend to flatten and this might be due to the fact that the number of collision increases and more paths are cut off (see 5.3).
The spacing between the duration curves might suggest that this scaling factor is the inverse of the corresponding p, as mentioned earlier. This would make complete sense because the average time a message takes to travel along a path of n hops is $n \cdot \frac{1}{p}$ (see 3.2.1).
In fact, while the spacing does indeed suggest an inverse dependency on p, the actual values turn out to be about 20% smaller. The existence of this additional ~ 0.8 scaling factor could be motivated as follows:
the eccentricity of a graph gives information about the longest path from a node to the furthest node u, but does not say anything about the number of paths of that length. In the graph there might very well be different paths of length ϵ that connect node 0 to u, either completely disjunct or with some nodes in common and "forks", "branch-ins" and "branch-outs". If these paths happens to be disjunct, node u will receive the message as soon as it arrives from the "fastest" one.

If the paths happen to have nodes in common, as it is the case in fig. 5.11, every time there is a "branch-in" ($[1,2]\rightarrow 3$ or $[4,5,6]\rightarrow 7$ in the figure) there is a possibility that the message will have a speed "boost" with respect to its average speed along a single queue and also a possibility that its progress will be interrupted by a collision.

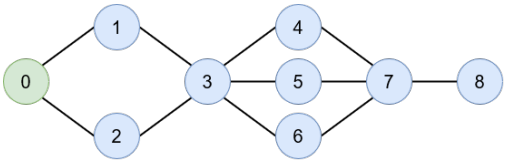


Figure 5.11: Example of graph with multiple shortest path of lenght ϵ

5.5 Performance optimization

The ultimate goal of this study is to find the optimal configuration of parameters to ensure the highest coverage in the lowest possible time, since the message should ideally reach all the users as soon as possible.
The trivial solution is to make the transmission radius so big that it covers the entire floorplan, thus ensuring total coverage in one single slot. This is clearly not viable if the floorplan size gets really big.
If one wishes to only maximize the coverage, regardless of the amount of time it would take to reach the last user, they should choose the biggest R the devices allow for and a low value of p: this configuration corresponds to the top right corner of figure 5.2. The downside of this choice of parameters is that the duration of the broadcast will be very long, as shown in the corresponding curve in fig. 5.6.
If, instead, one wishes to minimize the duration of the broadcast, possibly at the expense of not having full coverage, they should opt for a higher value of p. In this case, the choice of R is not as obvious: provided that R cannot be large enough to cover the whole area, it should be ensured that chosen value does not fall within the range that yields negative returns, as discussed in the second part of 5.4.1 and shown below in fig. 5.12. In practical terms, referring to fig. 5.8, if R can be greater than ~ 76 metres, it should be as high as possible; if it cannot be greater than ~ 76 metres, then R should be kept at ~ 25 metres, since higher values would result in a poorer performance.

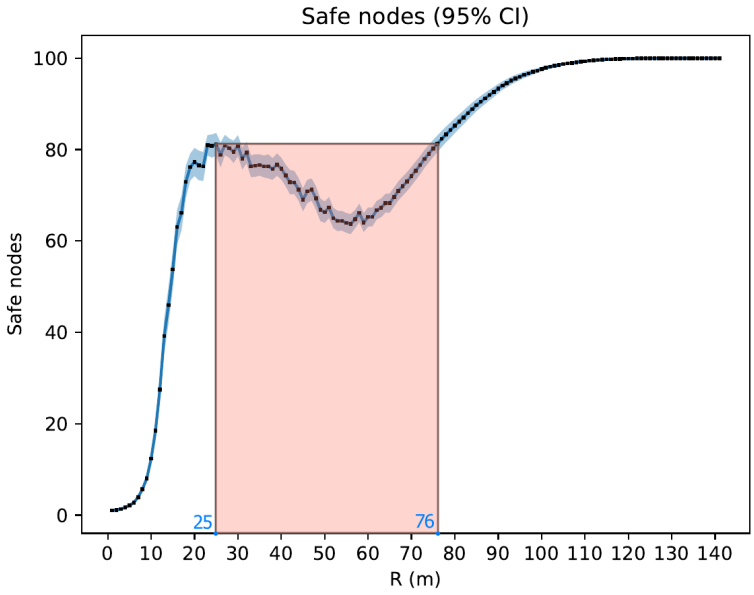


Figure 5.12

Chapter 6

Conclusions

Modelling the broadcast of an epidemic message in a floorplan with wireless hosts is not a trivial task. There are many factor that have an impact on the system and only few of them have been studied in this work.
Additional interesting result can be obtained by studying the system adjusting the parameters that in this study were kept fixed, such as the shape of the floorplan and the density of the users, or by exploring a wider range of parameters to confirm or dispute the assumptions made for the limit cases.
All the simulations which have been run for this study had static users with coordinates generated by a uniform random distribution on an empty floorplan, which might not be realistic. It would be worth analysing a floorplan with moving users, obstacles and/or users not uniformly distributed on the area, e.g. having some spots where users would crowd that may model shops or information points.

Chapter 7

Appendices

Appendix A

An example of stochastic matrix for a system with $N = 5$ and $p = 0.4$:

$$P = \begin{bmatrix} P_{0,0} & P_{0,2} & P_{0,3} & P_{0,4} & P_{0,S} & P_{0,5} \\ & P_{2,2} & 0 & P_{2,4} & P_{2,S} & P_{2,5} \\ & & P_{3,3} & 0 & P_{3,S} & P_{3,5} \\ & & & P_{4,4} & P_{4,S} & 0 \\ & & & & 1 & 0 \\ 0 & & & & & 1 \end{bmatrix}$$

Here with numerical values (rounded to 4 decimal places):

$$P = \begin{array}{c} \\ 0 \\ 2 \\ 3 \\ 4 \\ S \\ 5 \end{array} \begin{array}{c} \begin{array}{cccccc} 0 & 2 & 3 & 4 & S & 5 \end{array} \\ \begin{bmatrix} 0.0778 & 0.3456 & 0.2304 & 0.0768 & 0.2592 & 0.0102 \\ & 0.216 & 0 & 0.288 & 0.432 & 0.064 \\ & & 0.36 & 0 & 0.48 & 0.16 \\ & & & 0.6 & 0.4 & 0 \\ & & & & 1 & 0 \\ 0 & & & & & 1 \end{bmatrix} \end{array}$$

Appendix B

Below is a demonstration of the upper bound for the number of collision in a scenario with N hosts.

The necessary condition for a collision to happen is that at least two hosts transmit during the same slot. Each host placed in the intersection of the transmission circles of the two transmitting hosts detects one collision.
Therefore, to maximize the number of collisions, it is necessary in every slot, both to a) maximize the number of hosts that detect the collision and b) minimize the number of transmitting hosts (while still assuring that a collision can happen) so that during the next slot the number of hosts available to detect a collision is the highest.

Let us suppose to have a floorplan with 100 hosts.

1. host 0 transmits and, being the only host transmitting, absence of collision is ensured. Let us suppose that the message is received by host 99 and host 98.
2. host 99 and host 98 broadcast the message and, out of the 97 active hosts, only host 97 and host 96 receive the message correctly. The remaining 95 detect a collision.
3. host 97 and host 96 broadcast the message and, out of the 95 active hosts, only host 95 and host 94 receive the message correctly. The remaining 93 detect a collision.
4. And so on...

Following this line of reasoning, the number of listening hosts decreases by two every slot, and so does the number of collisions.
When only 5 hosts are remaining on the floorplan, 2 of which are transmitting and the other 3 are receiving, 2 of the latter 3 receive the message correctly and the last one detects a collision.
Finally, the last 2 out of 3 broadcast the message and the last one detects a collision again.
The total number of collision is therefore $95 + 93 + 91... + 3 + 1\mathbf{+1} = 2305$.

It is trivial to prove that if $N \leq 3$, no collisions can happen.
In general, for a configuration with N hosts, $N \geq 4$, the upper bound for the number of collision is given by the following formula:

$$C_{max}(N) = \begin{cases} 1 + (\frac{N}{2} - 2)^2 & \text{if N is even} \\ 2 + (\frac{N-1}{2} - 2)(\frac{N-1}{2} - 1) & \text{if N is odd} \end{cases} \tag{7.1}$$