UCL Université catholique de Louvain

École polytechnique de Louvain (EPL)

Epidemic Spreading Analysis through Distributed Simulation

Dissertation presented by
Floran HACHEZ
François ROBINET

for obtaining the Master's degree in
Computer Science and Engineering

Supervisor
Prof. Jean-Charles DELVENNE

Readers
Dr. Gautier KRINGS
Prof. Peter VAN ROY

Academic year 2015-2016

UNIVERSITÉ CATHOLIQUE DE LOUVAIN

Abstract

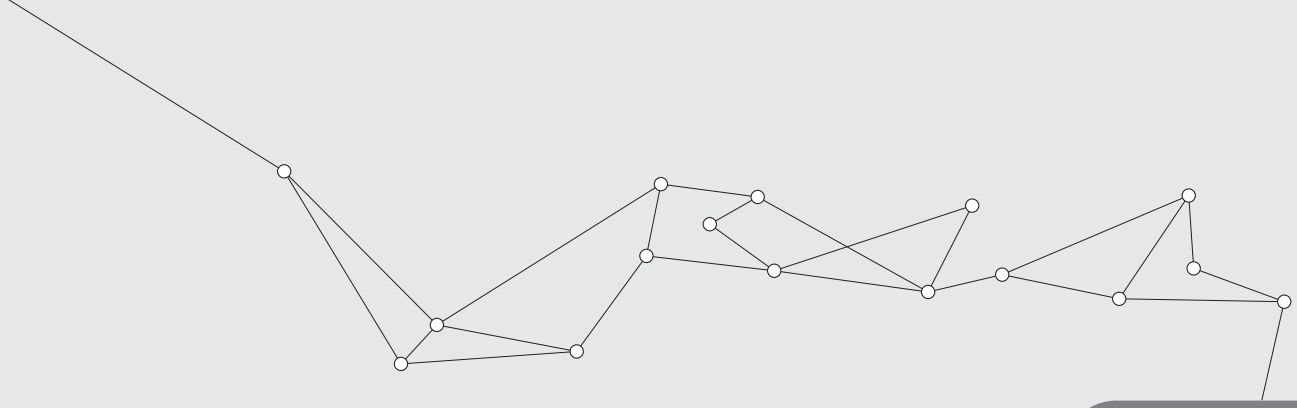
Louvain School of Engineering

INGI Department

Epidemic Spreading Analysis through Distributed Simulation

by Floran Hachez & François Robinet

Given the human and economical impacts of recent disease outbreaks, understanding epidemics has become a societal problem of prime importance. The increased mobility caused by the improvement of transportation has led some researchers to believe that a worldwide pandemic would occur before the end of the century. This thesis starts with a review of various epidemic models and their evolution. To obtain analytical results, these models have to resort to rather strong hypotheses, which are sometimes too unrealistic to be of any practical interest. To face this issue, a simulator was developed in collaboration with *Real Impact Analytics*. This program was designed to allow fine-grained modeling of each individual of the population through customizable behaviours. Using the simulator, we are able to confirm some theoretical results, and to study models for which analytical solutions are hard or impossible to obtain. Understanding the spreading dynamics of such models could be critical in the design of vaccination or quarantine strategies.

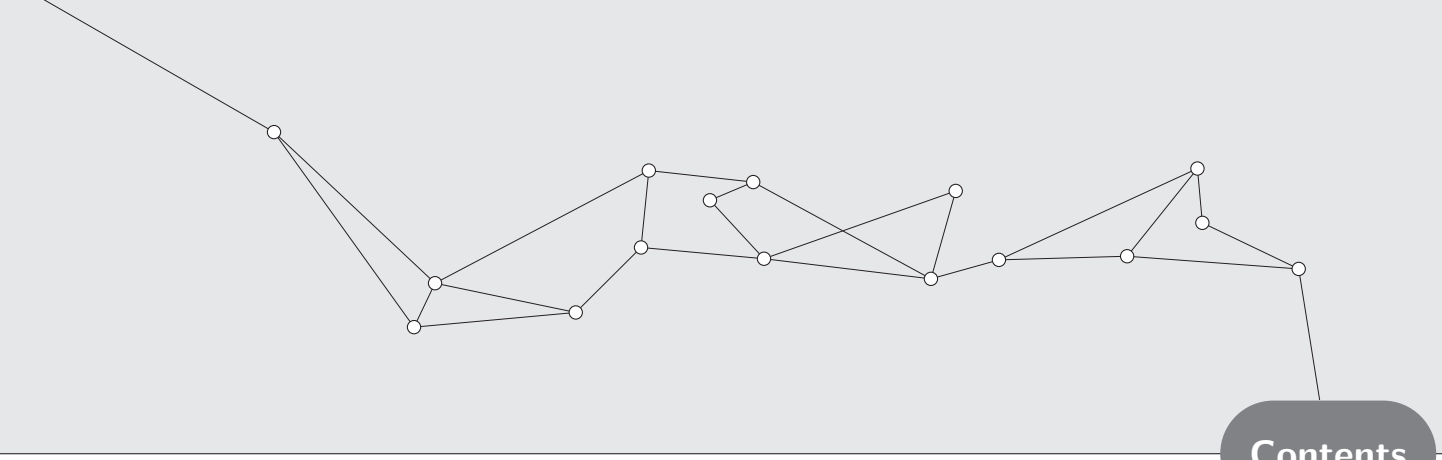


Acknowledgments

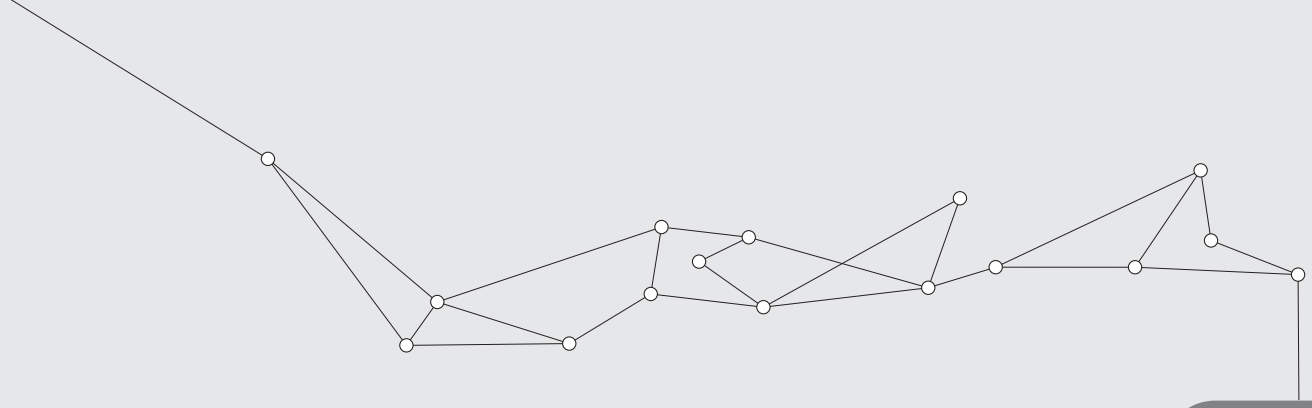
We would first like to thank our advisor, Jean-Charles Delvenne, for giving us the opportunity to freely design and explore our topic of research, as well as for his help during many fruitful meetings held in his office or over Skype.

We are also thankful to all the employees at *Real Impact Analytics* for their warm welcome during our visits in their Brussels office, for interesting lunch discussions and, of course, for many exciting kicker games. In particular, we would like to express our gratitude to Pierre Borckmans and Gautier Krings for their technical and theoretical support.

Finally, we would like to thank Benoit Pairet for proofreading our work, and Adeline Decuyper for giving us a hand in making chapter headers look beautiful.



Contents



Introduction

Spreading is everywhere: from the flu virus to Internet buzzes, it raises many important questions. How does a disease or an idea spread? What are the influencing factors? Can we slow it down or speed it up?

In the past, a pandemic would take years to spread across a continent. For example, as illustrated on figure 1, the bubonic plague took several years to disseminate across Europe [AM91]. Because of modern transportation, a pandemic which starts today at one point of the globe can travel the world in a matter of days or weeks. This was the case with the H1N1 outbreak that emerged in Mexico in April 2009 [Yan+09] and quickly spread across the world as shown on figure 2. According to Bill Gates, it is very likely that the improvement of transport will cause a worldwide pandemic to occur before the end of this century [Gat10].

We can observe a similar evolution for the diffusion of ideas. In the Internet and mobile phone era, information and rumours are spreading ever faster. It would be of public interest to be able to speed up renewable energy adoption by a population, or to slow down the spreading of concepts such as racism. Of course, such manipulations of the spreading of ideas raise major ethical problems.

Even though the spreading of diseases and the spreading of ideas may seem intrinsically different, they can be analysed as different expressions of similar high-level phenomena. Some scientists are more interested in analysing *how* the spreading occurs, rather than exactly *what* (ideas, viruses, ...) is spreading. The dynamics of spreading can sometimes be analysed without loss of generality by considering a network of individuals which are in contact.

To better understand the spreading process, we need to look at the micro-level and analyze individual human behaviours that influence our interactions. How do we move? With whom do we interact? At which frequency do interactions occur? Human behaviours are complex and vary from one individual to another. Understanding how the general complexity emerges from local behaviours is a fundamental question.

Epidemics have been under study for a long time, and the presence of new data sources has made research of insights into human behaviours more accessible than ever. Data gathered by our mobile phones, computers, wearable electronics, and social websites provide more and more insights into our behaviours and the structure of our social network. It is now possible to confront models with the reality to understand their shortcomings.

How can we control viral epidemics? Are there patients that should be cured in priority to help stop them? How should we distribute a limited number of vaccines? What behaviours should be discouraged? Should we try to decrease mobility by shutting down transportation, or cut contacts between communities? To be able to answer these questions and others, we need a detailed and quantitative understanding of spreading dynamics [Vaz+07]. However, as models

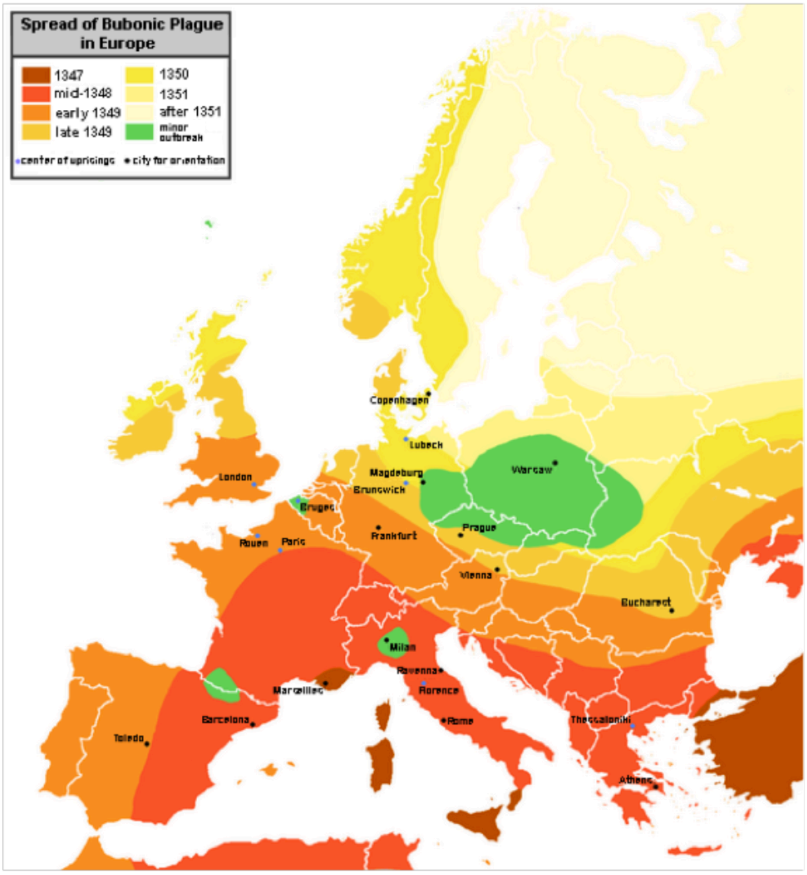


FIGURE 1: Spreading of the Black Death in Europe.
Figure reported from [AM91].

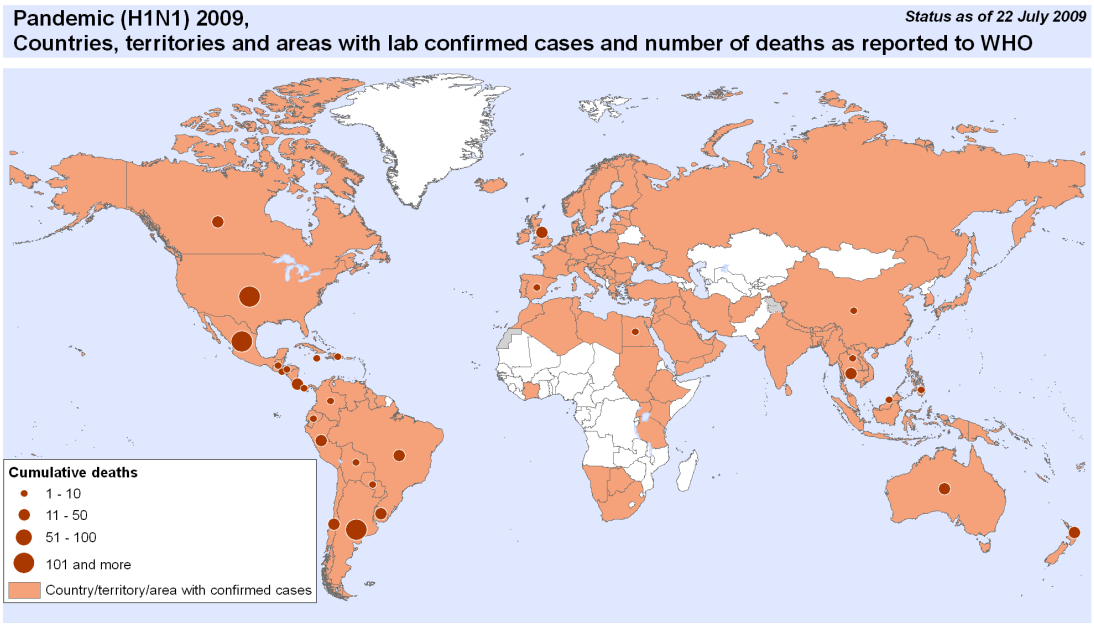


FIGURE 2: Status of H1N1 spreading on 22nd July 2009, as reported by the *World Health Organization*. The disease was first detected on 29th April 2009. Original figure from http://www.who.int/csr/don/Map_20090727.png

get closer to the reality, their mathematical analysis gets more complicated, and analytical solutions are not always achievable anymore. This problem can fortunately be circumvented by resorting to simulations.

During our thesis, we collaborated with Real Impact Analytics (RIA). RIA is a data analytics company based in Brussels whose specialty is to develop dashboard applications, providing large Telecom operators with insights about their operations and customers. Aside from this core business, RIA also takes advantage of its access to Telecom data from emergent markets to promote social good. Existing projects include using the high correlation between mobile phone vouchers purchases and food expenditures to predict and help prevent food crises [Ana].

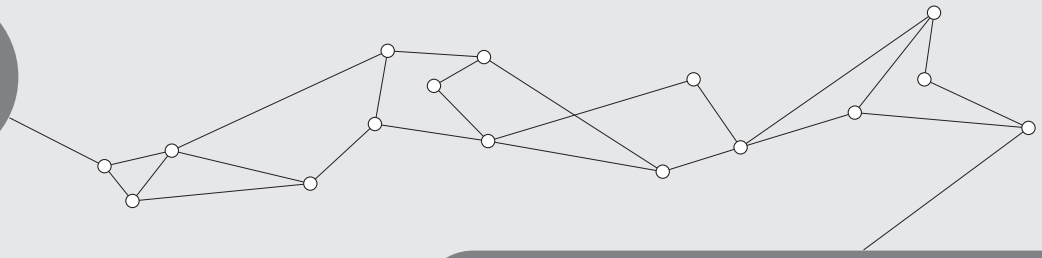
To conduct both industrial and humanitarian studies, researchers sometimes rely on data provided by Telecom operators. The bulk of data collected by Telecom operators consists of Call Data Records (CDRs). These CDRs contain metadata about phone interactions of any sort: call, SMS, *etc.* The identities of the persons involved, the duration of the interaction and the location of the towers through which the interaction was serviced are all logged.

Using CDRs datasets for research has many advantages. First, the datasets offer a size and a degree of granularity that would be impossible to achieve in a survey in which volunteers would record their interactions. Such studies are usually limited to small sample sizes, on the order of a few tens or hundreds of people [REE08; Mos+08]. Secondly, the facts that CDRs are collected through time in a systematic way, and that subscribers mostly maintain their contracts with the same operators, allows researchers to study dynamical aspects. Finally, the data contained in CDRs has proven to be relevant for many applications, especially when combined to extra information about the subscribers, such as age or gender. Amongst other uses, these datasets have been successfully used to estimate the population density in developing countries where census data is old or imprecise [Dev+14], and to study urban mobility and optimize transportation accordingly [Ber+13].

It is clear that Call Data Records are a very useful tool, but they are also hard to come by. Because the CDRs carry very sensitive information about customers, Telecom operators are sometimes reluctant to share even anonymized versions of the datasets. For this reason, part of the work of this thesis was dedicated to the implementation of a simulator able to generate CDRs. Our work is based on a proof-of-concept developed during a 2-day hackaton organized internally at RIA. Although some of the core ideas from this early version remain in our final work, the design has been greatly improved and most of the code has been rewritten. The data generated by the simulator could potentially be used by Real Impact Analytics during testing phases, or to present some realistic examples to a client.

In chapter 1, we present some of the classical models of epidemics. The limitations of these models forced us to reconsider some hypotheses. This led to models that are impossible to solve analytically, and motivated the creation of the simulator we describe in details in chapter 2. Chapter 3 presents an application of the simulator. The CDRs generated by the simulator were used as a proxy for social contacts, to be able to test hypotheses about the dynamics of epidemic spreading. This data-driven approach allowed us to work with some of the non-analytical models mentioned in chapter 1.

1



Epidemic Spreading & Social Networks

In this chapter, we review different models introduced over the years to model the spreading. The chapter starts with a definition of spreading, before presenting some classical models. These models make strong hypotheses about our behavior. We will challenge these assumptions and present more recent models that remove them to try to get one step closer to the reality. Real social networks are complex: some people only maintain a few connections, while others tend to build a large number of social ties. Moreover, these connections evolve through time, and we tend to interact more with some people than other (*e.g.* friends, family, ...). The frequency of our interactions is not constant either: our communications exhibit spikes of activity separated by longer periods of inactivity. Models that relax some of the initial assumptions get more complex, and analytical solutions are not always available. This motivates the creation of a generic simulator allowing us to specify human behaviors in a flexible way. This simulator will be the object of chapter 2.

1.1 What is Spreading?

The concept of spreading in general is very broad. From the pollen that floats into the air in springtime, to the diffusion of a droplet of ink in a glass of water, to the way smoke disperses into the surrounding air, many well-known processes can be seen as some form of spreading.

In this thesis, we will focus on epidemic spreading. Our vocabulary will often reflect this focus, and we will use terms like "susceptible" or "infected". However, we can consider the spreading of any "information" using the same theoretical concepts. The actual kind of information depends on the context. Other examples of interest include computer viruses, rumours or opinions.

One common aspect of the different kinds of spreading models is that they use the concept of interaction. For example, in mathematical models for disease spreading, the infection propagates from infected nodes to the sane nodes when they interact. Just like there are many possibilities for the concept that spreads, the interactions can also represent anything. Diseases often propagate through physical contact, but computer viruses can be transmitted over email or over a local network, and ideas can transit through spoken conversation or in the media.

As opposed to conservative spreading (*e.g.* diffusion of ink in water, or of smoke in the air), epidemics are multiplicative by nature: contaminating someone doesn't make the original infected any healthier. Once the interaction is defined, we can model how the population of infected evolves over time, and then compute the speed of the process. These models provide a way to predict the effect of different prevention and health-care measures. In the next section, we present some classical models of spreading dynamics.

1.2 Classical Epidemic Spreading Models

A classical example of spreading process was studied in the late nineteenth century [Ken75]. Some aristocrats were afraid that their family name would soon die out. To evaluate this possibility, Galton and Watson expressed the number of male descendants of a family after $n+1$ generation M_n as a simple stochastic branching process:

$$M_{n+1} = \sum_{i=1}^{M_n} C_i^{(n)},$$

where $C_i^{(n)}$ represents the number of male children produced by the ith male descendant of the nth generation. The total number of members of a generation is thus simply defined as the total number of children of members of the previous generation. Using rather strong assumptions on the distributions of $C_i^{(n)}$, it is sometimes possible to compute the probability of extinction of a lineage after an arbitrary number of generations. The same idea has been applied to study the development of epidemics [Bar67].

The models we will consider in this thesis use a different approach. In these models, individuals are stratified into compartments based on their health status. The dynamics of the disease is then included by defining transitions between the different compartments. We will start this section by a presentation of the most simple compartmental models, before moving on to the description of more advanced approaches, based on networks.

1.2.1 Basic Compartmental Epidemic Models

In compartmental epidemic modeling, we use the fundamental hypothesis that individuals can be classified into distinct discrete states. Moreover, transitions between states occur depending on time and interactions between individuals from different compartments. Common states considered in classical models are:

Susceptible (S): The individual is not infected yet.

Infected (I): The individual is infected and he is contagious.

Removed (R): The individual either died from the infection or has recovered and is now immune.

We could also imagine modeling the spreading of ideas or opinions using the same formulation with a different vocabulary:

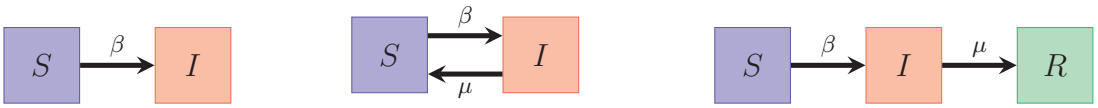
Influenced: hasn't adopted the opinion yet.

Influent: has adopted the opinion and will propagate it.

Converted: has adopted the opinion but will not propagate it.

There exist different compartmental epidemic models that are distinguished by the way states are considered and how transitions between states are defined. Models are often named after the order of the possible transitions. For example, *SI*, *SIS* and *SIR* are named after the transitions illustrated on figure 1.1. The *SIR* model is often used to study diseases for which immunity

follows the infectious period, *e.g.* varicella. Examples of diseases for which no such long-lasting immunity exists include the common cold or the flu. These illnesses are best analyzed using the *SIS* model.



(A) *SI* (B) *SIS* (C) *SIR*

FIGURE 1.1: Transitions between compartments for the *SI*, *SIS* and *SIR* models. A fraction β of susceptible individuals that is in contact with infected ones becomes infected per unit of time. Likewise, a fraction μ of infected individuals spontaneously becomes susceptible again per unit of time.

In order to be able to predict the fraction of the population lying in each state at any point in time, we need to make some assumptions about the state transitions. For the transitions from I to S or from I to R, the usual hypothesis is that a fraction μ of the infected individuals recovers at each time step. Likewise, a fraction β of susceptible individuals that come into contact with infectious ones catches the disease and transitions from the S state to the I state.

The central question is to decide how to model *when* and with *whom* individuals interact. Once this is determined, the model can be understood by simply applying the possible transitions and observing the evolution of the size of each class.

The original models rely on the hypothesis of perfect mixing. Perfect mixing means that any individual can interact with any other, and all these interactions occur with the same probability. The population is completely homogeneous, regardless of compartments. In other words, all individuals have the same chance of interacting with each other. If the population is composed of n individuals, each of them can thus interact with $n-1$ others. To be able to generalize the notations to more advanced models that will be presented later, we will denote the average number of persons with whom an individual is in a social relationship by $\langle k \rangle$. For the remainder of this section, $\langle k \rangle$ will thus be equal to $n-1$.

Another common assumption is the Poissonian activity: individuals interact at a constant rate. In other words, we can say that each individual interacts on average with a constant fraction $\langle c \rangle$ of his friends at any point in time. At a given timestamp, each individual is therefore expected to be involved in $\langle c \rangle \langle k \rangle$ interactions.

Although often unrealistic, these hypotheses have the benefit that they considerably ease the modeling. Indeed, thanks to their use, it is possible to express the variation of the fractions of population belonging to each class: $s(t)$ (susceptible), $i(t)$ (infected), and $r(t)$ (recovered). The different models, reported on table 1.1, are expressed in the form of systems of non-linear ordinary differential equations. The models consider that a fraction β of the susceptible individuals that come into contact with infected ones become infected, and that a fraction μ of infected individuals recovers regardless who they are in contact with. Both β and μ are expressed per unit of time. In the case of the *SI* and *SIS* models, we can express $i(t)$ analytically. Although it can be solved numerically, there is no closed-form solution for the *SIR* model. Figure 1.2 shows the solutions of each model for a given set of parameters.

All three models predict an exponential growth during the first part of the spreading, but their results differ later on. Table 1.1 reports the late time behaviour of the infected fraction of the population. Since no recovery is possible in the *SI* model, the whole population will end up

in the infected compartment. In the *SIR* model, all individuals will either die from the disease or recover, and we have $i(\infty) = 0$ and $r(\infty) = 1$. In the case of the *SIS* model, things are more complicated because individuals that recover are not removed from the population and can become infected again. The process becomes endemic and the proportion of infected reaches a plateau with a value that depends on the values of β, μ, $\langle c \rangle$ and $\langle k \rangle$. Of course, infectious spreading can still be stopped within the *SIS* model. To analyse this possibility, it is useful to consider the basic reproductive number.

The basic reproductive number, also called transmissibility factor, is denoted R_0. It corresponds to the average number of neighbours that a single infected can contaminate by direct transmission during its infectious period, inside of population full of susceptible individuals [Bar15]. Put more simply, R_0 is the average number of infections that an infected individual can cause before getting cured, in a population that is entirely susceptible:

$$R_0 \triangleq \frac{\beta \langle c \rangle \langle k \rangle}{\mu}$$

This quantity is very important. We can define a critical value $R_0^c = 1$, which allows us to predict whether or not an infection will spread. If $R_0 < 1$, an average infected will propagate the disease to less than one other individual before being cured, which will on average cause the infection to die out. The larger the transmissibility, the faster an infection will spread. The goal of a healthcare campaign should then be to try to reduce R_0 to slow down the infection, and possibly eradicate it. Figure 1.3 illustrates the behaviours of the *SIR* and *SIS* models when $R_0 < 1$.

To close this section, we mention one last quantity that is often refered to in epidemic spreading litterature: the spreading rate λ.

$$\lambda \triangleq \frac{\beta \langle c \rangle}{\mu}$$

The spreading rate is directly related to the transmisibility through

$$R_0 = \lambda \langle k \rangle ,$$

but has the advantage that it is independent of the average number of social ties $\langle k \rangle$. We can also relate critical thresholds on both quantity: $R_0^c = \lambda_c \langle k \rangle$.

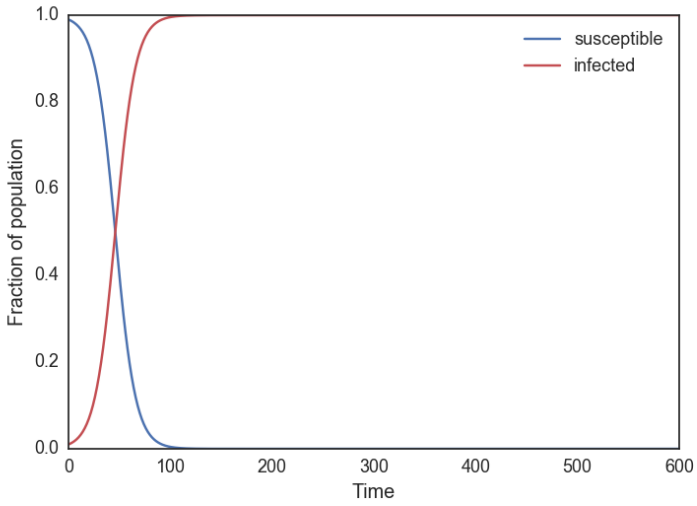
1.2.2 More Advanced Compartmental Models

The models we presented in the previous sections can be extended in many ways, depending on the particular disease and timescale considered. Variations include compartments for sane carriers, or maternally-derived immunity. Some models for long-lasting diseases also take population variations into account. We will limit ourselves to present only the SEIR and SLIR models.

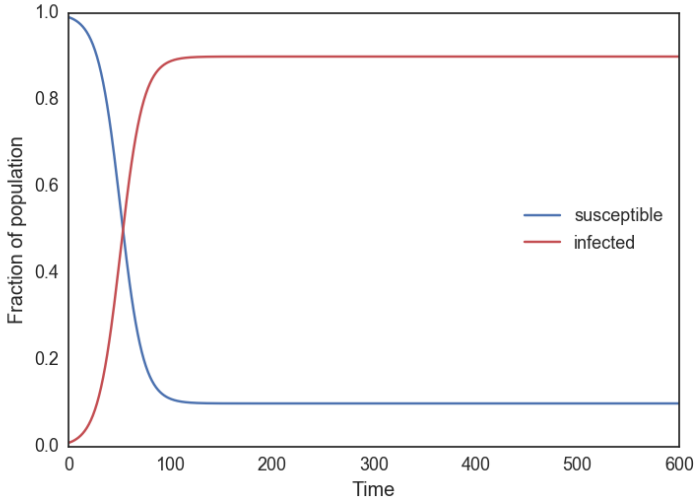
The SEIR model deals with incubation periods present for certain diseases by adding an "Exposed" state, in which an individual has been infected but is not infectious yet. In the SLIR model, the "Exposed" state is renamed "Latent" and the model incorporates vaccination strategies by allowing susceptible and latent individuals to become immune to the disease through vaccines. These two enhanced models are illustrated on figure 1.4.

SI	SIS	SIR
$\frac{di(t)}{dt} = \beta \langle c \rangle \langle k \rangle s(t)i(t)$	$\frac{di(t)}{dt} = \beta \langle c \rangle \langle k \rangle s(t)i(t) - \mu i(t)$	$\begin{cases} \frac{ds(t)}{dt} = -\beta \langle c \rangle \langle k \rangle s(t)i(t) \\ \frac{di(t)}{dt} = \beta \langle c \rangle \langle k \rangle s(t)i(t) - \mu i(t) \\ \frac{dr(t)}{dt} = \mu i(t) \end{cases}$
$i(t) = \frac{i_0 \exp(\beta \langle c \rangle \langle k \rangle t)}{1 - i_0 + i_0 \exp(\beta \langle c \rangle \langle k \rangle t)}$	$i(t) = \left(1 - \frac{\mu}{\beta \langle c \rangle \langle k \rangle}\right) \frac{\frac{i_0}{1-i_0} \exp((\beta \langle c \rangle \langle k \rangle - \mu)t)}{1 + \frac{i_0}{1-i_0} \exp((\beta \langle c \rangle \langle k \rangle - \mu)t)}$	no closed-form solution
$i(\infty) = 1$	$i(\infty) = 1 - \frac{\mu}{\beta \langle c \rangle \langle k \rangle}$	$i(\infty) = 0$
No threshold	$R_0^c = 1$	$R_0^c = 1$

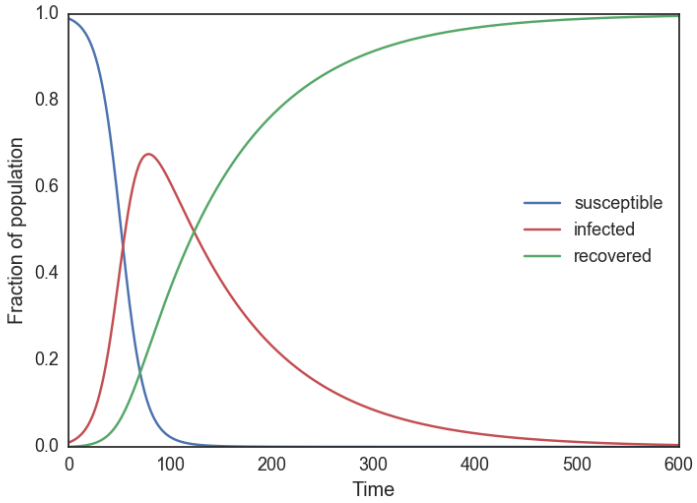
TABLE 1.1: Systems of ordinary differential equations for the SI, SIS and SIR classical models. All three models use perfect mixing and Poissonian activity. The second row presents analytical solutions for $i(t)$. The last two rows give the late time proportion of infected $i(\infty)$ and the critical value R_0^c of the transmisibility. If $R_0 = \frac{\beta \langle c \rangle \langle k \rangle}{\mu} < 1$, the infection dies out for the *SIS* model, and doesn't reach the whole population in the case of *SIR* [Het00; Bar15].



(A) *SI*

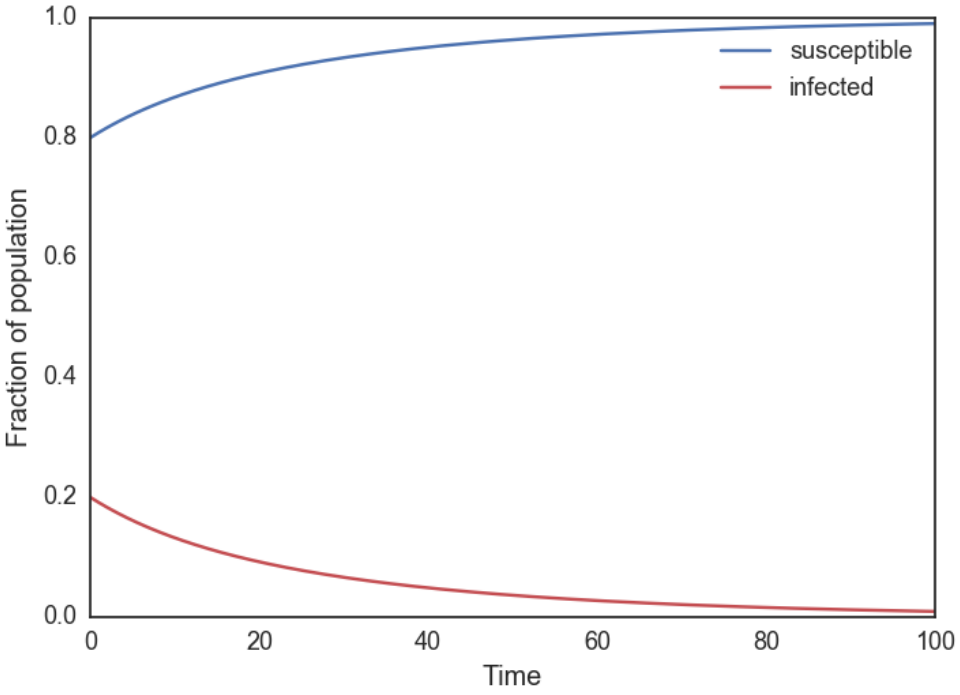


(B) *SIS*

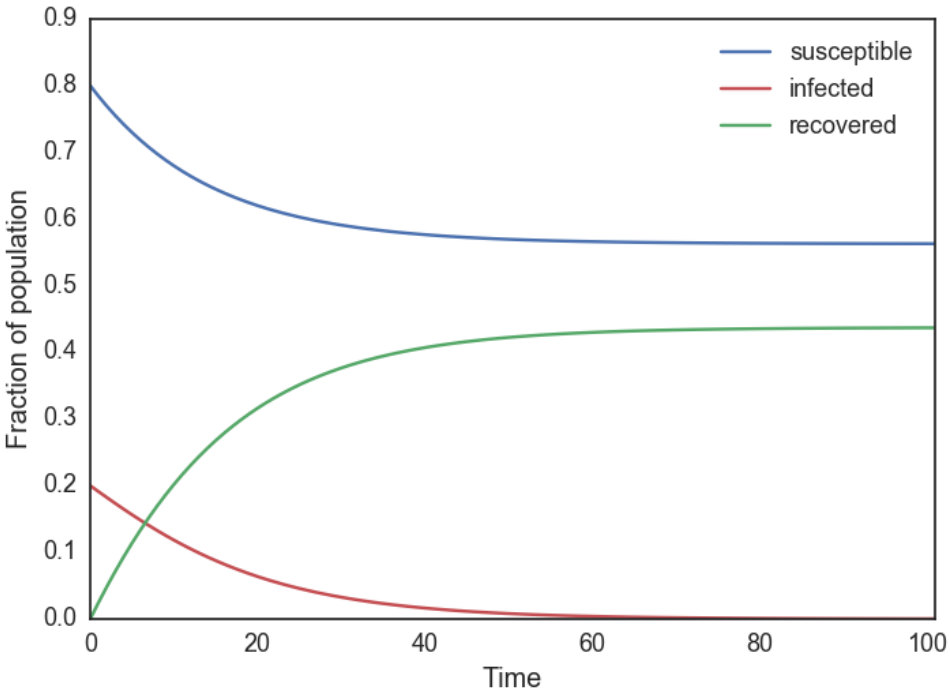


(C) *SIR*

FIGURE 1.2: Analytical solutions of the SI and SIS models. The SIR curve was obtained using the explicit Euler method. Parameters are $\beta = 0.01$, $\langle c \rangle \langle k \rangle = 10$, $\mu = 0.01$ and $i(0) = 0.01$. The transmissibility is given by $R_0 = \frac{\beta \langle c \rangle \langle k \rangle}{\mu} = 10$. Because $R_0 > 1$, the infection does not die out in the *SIS* case and spread through the whole population in the *SIR* case.



(A) *SIS*



(B) *SIR*

FIGURE 1.3: *SIS* and *SIR* models with parameters $\beta = 0.01$, $\langle c \rangle \langle k \rangle = 10$, $\mu = 0.125$ and $i_0 = 0.2$. These parameters yield $R_0 = \frac{\beta \langle c \rangle \langle k \rangle}{\mu} = 0.8 < 1$, which makes the infection vanish very quickly. The *SIR* infection presents with $s(\infty) > 0.5$ and $i(\infty) = 0$, which clearly shows that the infection was eradicated even before it could reach half of the population.

(A) *SEIR* (B) *SEIR*

FIGURE 1.4: The SEIR and SLIR enhanced models. SEIR allows an individual to be in the "Exposed" state (infected but not infectious) for some time before transitioning to the Infected state. SLIR adds a "Latent" state. For some diseases, such as AIDS, this "Latent" state is used to model an individual that is infected and contagious, but does not experience symptoms yet. A "Latent" state also proves useful to model vaccination schemes: susceptible and latent individuals can become immune to the disease through vaccination.

1.2.3 Limitations of the Classical Models

The epidemiological models we presented in this section are quite simple to understand, and may be solved analytically, or numerically using very simple techniques. All these models build upon two major assumptions: perfect mixing and poissonian activity.

Although these two assumptions may provide an acceptable approximation in some cases, they are not realistic in general. We know that humans don't mix homogeneously. Depending on the timescale we are considering, they can't be seen as interacting in a completely regular fashion either. For example, humans clearly exhibit patterns of activity, such as different behaviours during the day and during the night, or depending on the season. To account for this, we sometimes need to resort to more fine-grained models, based on subpopulations or on networks of individuals.

1.3 Challenging the Perfect Mixing Hypothesis

In the real-world, people don't randomly interact with anyone: individuals only maintain social relationships with a small part of the population, and the same contacts are usually repeated.

1.3.1 Considering Subpopulations

Some authors consider the whole population as a set of many subpopulations [SS88; All94; Hes96; Het00]. Within a single subpopulation, perfect mixing is assumed so any individual has the same probability of being in contact with any other. Between subpopulation, we also assume such homogeneous mixing, but the rate of contact between two individuals from different subpopulation is lower than if they belong to the same one. This model generalizes classical compartmental models: if we set the inter-subpopulation contact rate to the same value as the intra-subpopulation contact rate, we get the classical models back.

When the population cannot be globally considered as homogeneous, considering subpopulations can drastically affect spreading results. In January 1989, a measles epidemic occurred on the Texas Tech University campus. The outburst lasted for several weeks, and was finally eradicated thanks to a massive vaccination campaign. Researchers later used data gathered during the epidemic to study it using subpopulation models [All+90].

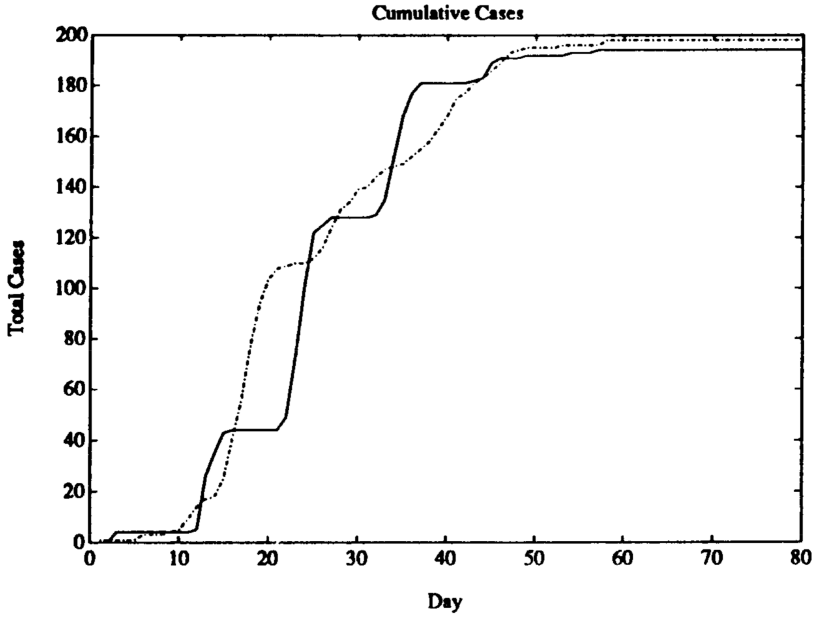


FIGURE 1.5: Cumulative number of measles cases as function of days elapsed since the beginning of the outbreak. The dotted line represents predictions of the SLIR model with subpopulation, and the solid line shows the true value gathered from the data. Figure taken from [All+90].

At the time, Texas Tech was attended by 6104 on-campus students, and by 17396 students living off campus. On-campus students are split into 11 dormitory complexes, and students of the same complex share the same dining-hall. Because of these shared spaces, there are significantly more interactions between students from the same dormitory complex than between students from different complexes. In the case of this epidemic, global perfect mixing across the whole population should therefore not be assumed. The subpopulation model, however, seems reasonable and was chosen by researchers to model the epidemic.

Each dormitory complex was first divided into four classes corresponding to the compartments of the SLIR model. The SLIR approach was selected to account for the vaccination strategy that was applied at the time of the outburst. Figure 1.5 compares the results obtained through simulation to the real numbers compiled from data gathered by researchers. In this particular case, the predictions of the subpopulation model mostly match the real results. The final size of the epidemic was also correctly predicted.

The subpopulation model also has the benefit of being flexible, because subpopulations can be distinguished based on any criterion. If the goal is to model an epidemic at the scale of a country, we can for example imagine grouping inhabitants by city. When considering the spreading of an idea, it might be relevant to categorize people by age or gender.

1.3.2 Network Epidemics

Subpopulations can be further refined, by recursively splitting them. If we apply this reasoning to the limit, we end up with subpopulations containing only a single individual. We can then represent each member of the population as a node in a social graph. Each node of the graph belongs to one of the usual compartments (*e.g.* susceptible or infected). The fact that two individuals may interact is represented by an edge between the two nodes. Therefore, the

disease can spread only along the edges. The weight w_{ij} of an edge between the node i and j influences the probability that, if the individual i interacts, he interacts with the individual j.

Both the classical models and the subpopulation models can be expressed as particular cases of social network models. When we consider a complete social graph with constant weights w_{ij}, the network respects the global perfect mixing assumption and we get a classical model back. We can mimic subpopulation models by creating a complete graph with weights w_{ij} that depend on whether or not i and j belong to the same population.

Social networks allow to express more general models because the possible interactions are restricted to occur between neighbours of the graph. Since this graph doesn't have to be complete, we can express different patterns of interaction [New03]. Using subpopulations, we could have set the rate of interactions between two populations to 0, but social networks allow us to operate at the individual level. Of course, such flexibility comes at the price of having to generate realistic social networks.

Nowadays, we have access to data sources that can help us understand the topology of real social networks, and try to account for major properties of our social interactions when studying epidemics. The possibilities of fine-tuning the network have given rise to the field of network epidemiology [KN10].

Before we had access to email databases, or datasets from Twitter, some scientists tried to estimate properties of social networks. The "Small-world Experiment" is a famous example of such experimentation that took place in 1967. Researchers tried to estimate the average path length of the social network of the United States [Mil67]. The experiment consisted in sending letters, containing the name and address of a "target", to random citizens of several states. The recipients were asked to resend the letter to the target if they knew him personally, or to forward it to one of their acquaintances which they deemed most likely to know the target. The forwardings of the letters were recorded, and the experiment concluded to an average of between five and six forwardings.

Since this experiment, distances in social networks have been further studied. We define the "distance" separating two nodes as the length of the shortest path that links them. In 2011, a study conducted on data from Facebook revealed that the average distance in the social graph of Facebook was of 4.7 [Uga+11]. Figure 1.6 shows the percentage of pairs of individuals that can be connected as a function of the allocated distance. Another metric of interest is the diameter of the graph. The diameter is defined as the longest distance separating any pair of nodes. In the case of the Facebook graph, its value is around 6. The same paper also shows that the average Facebook member has around 200 friends, with some members (hubs) largely exceeding this amount. This is concordant with Dunbar's number, which the psychologist suggested in 1992 as a limit on the number of social relationships humans could maintain. This number was estimated to lie between 100 and 230 [Her+10].

Over the years, scientists have proposed many models that try to reproduce different characteristics of empirical social networks. The next sections present a few of the most popular ones.

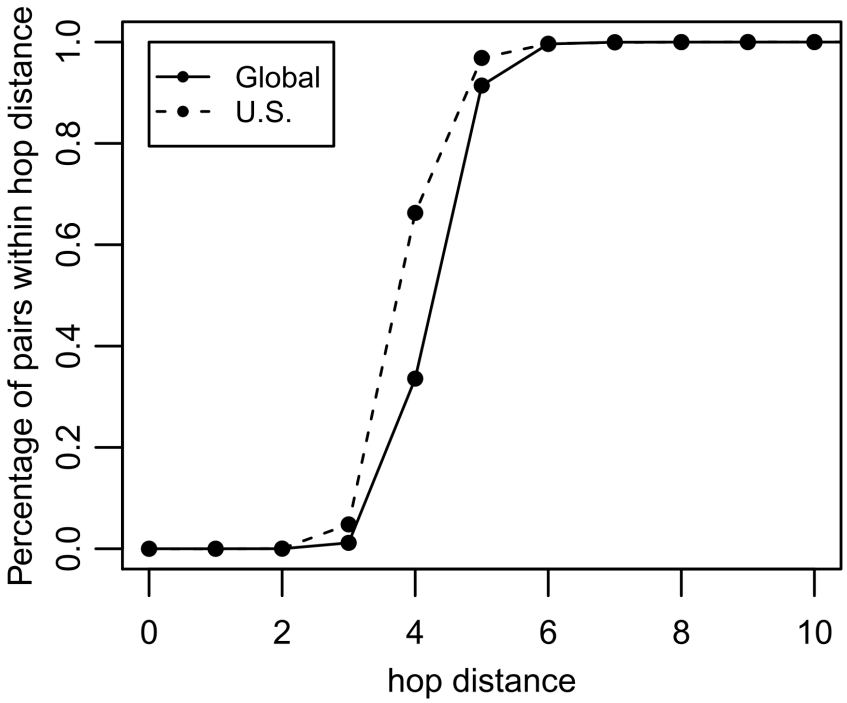


FIGURE 1.6: Percentage of pairs of individuals that can be connected as a function of the allocated distance. A distance of 6 is sufficient to connect every pairs of nodes, which means the diameter is equal to 6. Figure reproduced from [Uga+11].

1.3.3 The Erdős-Rényi Model

In 1959, Erdős and Rényi proposed a random graph model that exhibited the then only theorized small-world property [ER59]. The ER graph is defined by only two parameters: its number of nodes n, and a probability p that any given edge (excluding self-loops) is part of the graph. Because the choice of creating an edge corresponds to a binomial trial, the degree of nodes follows a binomial distribution:

$$deg(v) \sim Bi(n-1,k) \iff P(k) \triangleq P(deg(v)=k) = \binom{n-1}{k} p^k (1-p)^{n-1-k}$$

Despite being very simple, ER random graphs exhibit the small-world property. Indeed, it has been shown that their expected average distance was given by

$$\bar{d} \approx \frac{\log n}{\log \langle k \rangle},$$

where $\langle k \rangle$ is the average node degree [WS98; PS+15]. In terms of epidemics, this small-world property boosts the spreading by providing a short path to different parts of the graph.

Epidemic Thresholds on Network Models

It would be interesting to characterize the resilience of an epidemic over network models. In the classical models presented in section 1.2.1, this resilience was measured using the transmibility

factor R_0. Our goal is to express a critical value R_0^c such that the disease is expected to die whenever $R_0 < R_0^c$. For classical models, it was established that $R_0^c = 1$ by simple inspection of the equations from table 1.1.

Network epidemics are much more complex because of the interactions of large numbers of individuals which can all have distinct characteristics, such as different degrees. One common technique to analyze such complex stochastic systems, called the mean field approximation, is to form groups of similar individuals and to consider that all members of a group adopt the same behaviour. In the case of network epidemics, we can for example make the simplifying assumption that the behavior of an individual depends only on his degree.

This idea was applied by Pastor-Satorras and Vespignani in conjunction with the Poissonian activity assumption, and led to the expression of topology-dependent thresholds on the spreading rate λ and, equivalently, on the transmissibility R_0 [PSV02]:

$$\begin{aligned} &\textit{SIS:} \quad \lambda_c = \frac{\langle k \rangle}{\langle k^2 \rangle} \quad &\longrightarrow \quad R_0^c = \lambda_c \langle k \rangle = \frac{\langle k \rangle^2}{\langle k^2 \rangle} \\ &\textit{SIR:} \quad \lambda_c = \frac{\langle k \rangle}{\langle k^2 \rangle - \langle k \rangle} \quad &\longrightarrow \quad R_0^c = \lambda_c \langle k \rangle = \frac{\langle k \rangle^2}{\langle k^2 \rangle - \langle k \rangle} \end{aligned}$$

In these equations, $\langle k^2 \rangle$ denotes the second moment of the degree distribution of the network. The expression shows that, even for equivalent expected degrees $\langle k \rangle$, changing the topology of the network has an influence on the threshold through variations of $\langle k^2 \rangle$.

In the particular case of sufficiently sparse Erdős-Rényi networks, we can approximate the second moment as $\langle k^2 \rangle = \langle k \rangle (1 + \langle k \rangle)$ [Bar16]. This results in

$$\begin{aligned} &\textit{SIS:} \quad \lambda_c = \frac{\langle k \rangle}{\langle k \rangle (1 + \langle k \rangle)} = \frac{1}{1 + \langle k \rangle} \quad &\longrightarrow \quad R_0^c = \lambda_c \langle k \rangle = \frac{\langle k \rangle}{1 + \langle k \rangle} \approx 1 \\ &\textit{SIR:} \quad \lambda_c = \frac{\langle k \rangle}{\langle k \rangle (1 + \langle k \rangle) - \langle k \rangle} = \frac{1}{\langle k \rangle} \quad &\longrightarrow \quad R_0^c = \lambda_c \langle k \rangle = 1 \end{aligned}$$

For sufficiently sparse ER networks, we thus observe the same epidemic thresholds as defined for classical models, for both SIS and SIR infections. The interpretation of this value is straightforward: the disease will on average survive or reach the whole population if each infected individual manages to infect at least one of its susceptible contacts. We should emphasize that, since network models are stochastic, $R_0 < R_0^c$ does not necessarily imply that the epidemic is guaranteed not to spread. Instead, it simply defines an expected behaviour. In section 1.3.5, we will discuss an example of network model for which $R_0^c \neq 1$.

1.3.4 The Watts-Strogatz Random Graph

In empirical social networks, it was observed that individuals tend to have social relationships with the friends of their friends. This causes triangles to appear in the social network. The presence of these triangles creates densely connected regions in the graph that are called communities. The standard measure of this feature of social graph is the clustering coefficient, which is defined as the following ratio [New03]:

$$C = \frac{3 \times \#\text{ triangles in the network}}{\#\text{ connected triples of vertices}}.$$

Even though ER graphs have the small-world property, their clustering coefficients don't correspond to the high ones that are typically observed in empirical networks. Another model, proposed by Watts and Strogatz, aims at correcting this difference [WS98].

The networks generated by this model exhibit high clustering coefficients. A WS graph is neither completely random nor completely structured. The idea is to start from a completely regular lattice, and to randomly rewire some of the edges.

The small-world property is preserved. As we already mentioned, this preservation is important to study epidemics because it implies faster spreading. As more rewiring occurs, the diameter of the network gets lower. This lowering causes both the epidemic threshold and the time it takes for the infection to reach its peak to decrease, as studied by Watts and Strogatz [WS98].

1.3.5 Scale-Free Networks

Since the degrees of the nodes of both the ER and WS models follow a binomial distribution [New03], which has a small variance, they are generally quite homogeneous. However, in real networks, researchers have discovered the presence of hubs, which are largely connected individuals [AJB99; Uga+11]. This led to the emergence of new models of so-called "Scale-Free" networks (SF).

Research has shown that many real networks, such as the world wide web or the graph of email correspondences, exhibit power law degree distributions [AJB99; Vaz07; Bar05]. The probability that a given node has degree k is given by

$$P(k) = \begin{cases} Ck^{-\alpha} & \text{if } k_{min} \leq k \leq n-1 \\ 0 & \text{otherwise} \end{cases}$$

where $\alpha > 1$ and C is an appropriate normalization factor. This distribution is skewed and heavy tailed. Some individuals, the hubs, are linked to many others. The networks whose degree distribution is a power law are often called Scale-Free, because the power law is the only distribution that is invariant to scaling [New05].

An important characteristic of SF networks is that the variance of their degree distribution can be large. The average value of power law distributions is not defined whenever $\alpha \leq 2$, and neither is their variance for $\alpha \leq 3$. This potential for high variance allows the degrees of the network to be highly heterogeneous, as they are in empirical networks.

To understand the process behind the existence of empirical SF networks, and thus be able to generate such networks, Barabási and Albert suggested a new network model [BA99]. This Barabási-Albert (BA) network is based on two assumptions:

Growth: The network is growing *i.e.* more and more nodes are added to it. The procedure starts with m_0 initial nodes, and each added node is attached to m existing ones.

Preferential attachment: New nodes are more likely to attach to already highly connected nodes. This is sometimes referred to as the Matthew effect: "The rich get richer and poor get poorer".

These two properties are inseparable: using only one of them will not result in the generation of a SF network [Bar12]. The BA model was shown to generate random graphs with two interesting properties:

- The degree distribution follows a power law of degree 3: $P(k) \sim k^{-3}$ [BA99].
- The expected diameter is small: $D_{exp} \sim \frac{\log n}{\log \log n}$ [CH03].

The clustering coefficient of SF networks is typically smaller than what is observed in WS networks. Although their clustering coefficient is often larger than in ER networks, BA networks don't completely capture the high clustering property of empirical social networks [KE02; BR02].

The BA model allows the reproduction of the power law degree distributions that are typical in empirical networks. This model also has the small-world property.

Epidemic Thresholds on SF networks

The Scale-Free property is of prime importance when analyzing epidemic spreading on a network. When discussing the SIS and SIR spreading models, we already mentioned the existence of an epidemic threshold R_0 below which an epidemic couldn't spread. As a reminder, under the Poissonian activity assumption, mean field theory can be used to derive a threshold on the spreading rate λ [PSV02]:

$$\lambda_c^{SIS} = \frac{\langle k \rangle}{\langle k^2 \rangle} \qquad \lambda_c^{SIR} = \frac{\langle k \rangle}{\langle k^2 \rangle - \langle k \rangle}$$

In the case of SF networks, the second moment of the degree distribution is expressed as:

$$\langle k^2 \rangle = \sum_{k=k_{min}}^{n-1} k^2 P(k) = \sum_{k=k_{min}}^{n-1} C k^{2-\alpha}$$

When the size of the network tends to infinity, the expression only converges for α larger than 3. In hypothetical infinite SF networks with $2 < \alpha \leq 3$, $\langle k \rangle$ is finite and $\langle k^2 \rangle$ is infinite. This means that, as the size of the network tends to infinity, the epidemic threshold disappears for both the SIS and the SIR models [PSV01a; PSV01b; BPSV03]:

$$\lim_{n \to \infty} \lambda_c = 0.$$

In 2009, Chatterjee and Durrett went a step further and proved that this threshold actually vanishes for any value of α, whenever n grows to infinity [CD09]. These discoveries are important because they mean that diseases can persist on infinite scale-free networks, even when the rate of infection β is small. In chapter 3, we experiment on SF networks and observe that the decrease of the epidemic threshold λ_c is already clearly visible even for moderate network sizes.

The notion that a transmissibility $R_0 < 1$ may still on average lead to a massive spreading may seem counterintuitive. Indeed, we explained that R_0 could be interpreted as the average number of infections caused by a single infected during his infectious period, assuming all his neighbors were susceptible. How could a disease propagate when an infected causes less than one other infection on average? The reason is the existence of hubs in scale-free networks. If the disease is transmitted to a hub, the large number of contact he receives will on average trigger a massive spreading. Even when $R_0 < 1$, infecting one hub from time to time may still be enough to keep the disease from dying.

Control of Epidemics on SF Networks

Intuitively, we know that the more connections a node has, the more likely it is to spread the disease. Since hubs have more connections, they are likely to be infected in the early stage of the spreading and their high degree will cause them to quickly infect many other nodes. The spreading of an epidemic in the early stage is thus faster in a scale-free network [Vaz06].

The scale-free property is also important when studying vaccination strategies. To have the largest possible impact, we want to give the vaccine to nodes that have the highest degrees first. Of course, these individuals are not known in practice. One possible strategy is then to select a random node, and to give the vaccine to a subset of its neighbours. If we only have v vaccines to give, we have a higher probability of targeting hubs using this strategy than by simply vaccinating v random nodes. Intuitively, nodes with higher degrees will have more chance to be selected by the strategy because they have more neighbours [Fel91; CHBA03].

1.3.6 Network Epidemics Summary

In this section, we have seen different models that are closer to the real-world than the classical compartment models. Nowadays, a lot of research focuses on network models. The main properties that are observed in empirical networks are a small average path length, a high-clustering coefficient, and a power law degree distribution. The ER random graph was historically the first network model. It is simple, but it manages to reproduce the small-world property. Watts and Strogatz later proposed an improved model that provides a high clustering in addition to the small average path length. Finally, Barabási and Albert suggested a model that allows the presence of hubs (power law degree distribution), but doesn't have a high-clustering coefficient. The scale-free property was observed to have a large impact on the spreading, both by accelerating the early stage of the epidemic, and by making the threshold vanish for large graphs.

1.4 Challenging the Static Unweighted Graph Assumption

Until now, we have mostly assumed that the graphs we used were static and unweighted. In this section, we explain the importance of reconsidering these assumptions.

1.4.1 Weighted Connections

Although an individual can have many social ties, his social activity will not be uniformly spread among all his connections. We typically interact more often with our family and close friends that with the rest of our acquaintances. As our graph models get closer to real social networks, this difference should be addressed.

In most empirical social networks, we can observe that individuals interact a lot more with people from their community [Onn+07]. We can modify our static graph models to take this into account by adding weights to the edges: the larger the weight, the more active a connection will be. As more data is made available to researchers, it becomes possible to approximate the strength of a relationship between individuals by computing their total interaction time [Vaz+07]. Of course, the data available to researchers only provides a partial view of the underlying, dynamical, social network [OC12]. The size and placement of the window of time covered by the data at hand also plays a role in defining the structure of the aggregated network [Kri12].

Weighted connections can have a drastic impact on the spreading of epidemics. Indeed, since most communication occurs inside communities, diseases will tend to get trapped inside these communities. This phenomenon was for example observed for SI spreading [Onn+07]. Other studies showed that removing the weights or the communities resulted in faster spreading [Kar+10; VK09].

1.4.2 Dynamic Graph

Real-life social networks evolve through time: some friendships change in strength, some disappear or get created. Other external factors include the birth and death of individuals which directly impacts the topology of the network [Ban+10].

During an epidemic, people generally try to avoid interacting with infected individuals. If you get the flu, you will likely stay at home and avoid meeting too many people. This process was shown to have great potential impact: a weak isolation of the infected individuals can be enough to stop a SIS spreading in the long term [ZRG08].

Some of the topological changes may be irrelevant depending on the timescale we consider. For example, the births of new individuals are probably irrelevant when looking at an epidemic spreading over days or weeks [Ste+11]. We still need to be careful not to dismiss some apparently irrelevant events. For example, by creating new ties, individuals can create bridges between communities, helping the disease to spread to other communities and thus accelerating the spreading [VM07].

How do we take topological changes into account? One possibility is to use a temporal network. These networks are dynamic: some vertices and links are only active at given moments, and inactive otherwise [HS12]. In a temporal network, an interaction is represented by an edge that remains active for the duration of the communication. This information can be found in some dataset, *e.g.* in emails or phone logs. We need to be careful when using this kind of dataset, because they only contain interactions that occurred inside a given time window. The smaller this time window, the more partial the observations are [OC12; Kri12].

As we mentioned in the previous section, researchers sometimes make use of temporal data to create a (weighted) static network. The static graph is built by putting an edge between two individuals if they had an interaction during the span of the temporal data we have access to. Of course, temporal information is lost in the process. Some interactions might be correlated with each other. For example, you may call a friend to invite him to a party, causing him to call another friend to know if he wants to attend the event. These correlations between interactions disappear when you build a static network from temporal data. This also means that using different temporal networks may result in the same static aggregation [Sch+14]. For example, in the case of the analysis of CDRs, randomizing the time at which calls were placed has no impact on the static graph obtained by aggregating the data.

If the interactions are highly correlated, using a static network may result in predictions that are far off the reality. It was shown that taking temporal correlations into account speeds up the spreading within a community, but also helps trap the epidemic inside the communities [Kar+10; Kiv+12; RLH11; VK09]. The speed up mainly occurs in the early stage of the epidemic, when the disease is still within the community of the first infected. Correlations can also cause earlier and larger epidemics, as suggested by a study using the SIR model [RLH11].

The different dynamics visible at different timescales are summarized on figure 1.7. These dynamics have an impact on spreading. Temporal correlations can slow it down by trapping the disease inside a community, while tie creation may accelerate the spreading by providing new bridges between the communities. The static network approximation we used so far fail to model topological changes or correlations between interactions. However, all the dynamics are not always relevant, depending on the timescale of the epidemics we consider.

Finally, the order of interactions may sometimes be relevant for epidemic spreading. On the toy example of figure 1.8, we observe that the order of temporal interactions prevents an infected D from ever infecting A, because the neighbours of A that interact with D never contact A afterwards. When using a static network obtained by aggregation from the temporal data, this ordering information is lost, and D will be able to propagate an infection to A via B.

1.5 Challenging the Poissonian Activity

The models we have studied so far all assumed that our social activity was Poissonian, *i.e.* that we interacted at a constant rate. In reality, human behaviour is much more complex: our interactions occur in bursts and are strongly correlated. We have already discussed how correlations affected the spreading in the previous section. It turns out that the burstiness of our contacts also has an impact on the speed of epidemics.

In many real datasets, it was discovered that the inter-event time, *i.e.* the duration between two interactions involving a same individual, generally follows a power-law. This was observed for several different kinds of activities, including emails [EMS03; Bar05; Mal+08], Wikipedia edits [Gan+16], phone calls [MML11], and even old-fashioned letter correspondence [OB05].

Such a power-law distribution of the inter-event time means that many activities will be separated by only a short time interval, and that some longer pauses will occasionally occur. This clustering of activities in time is what we call “burstiness”. Just like using a power-law for the degree distributions avoided the homogeneity of degrees and allowed the presence of hubs

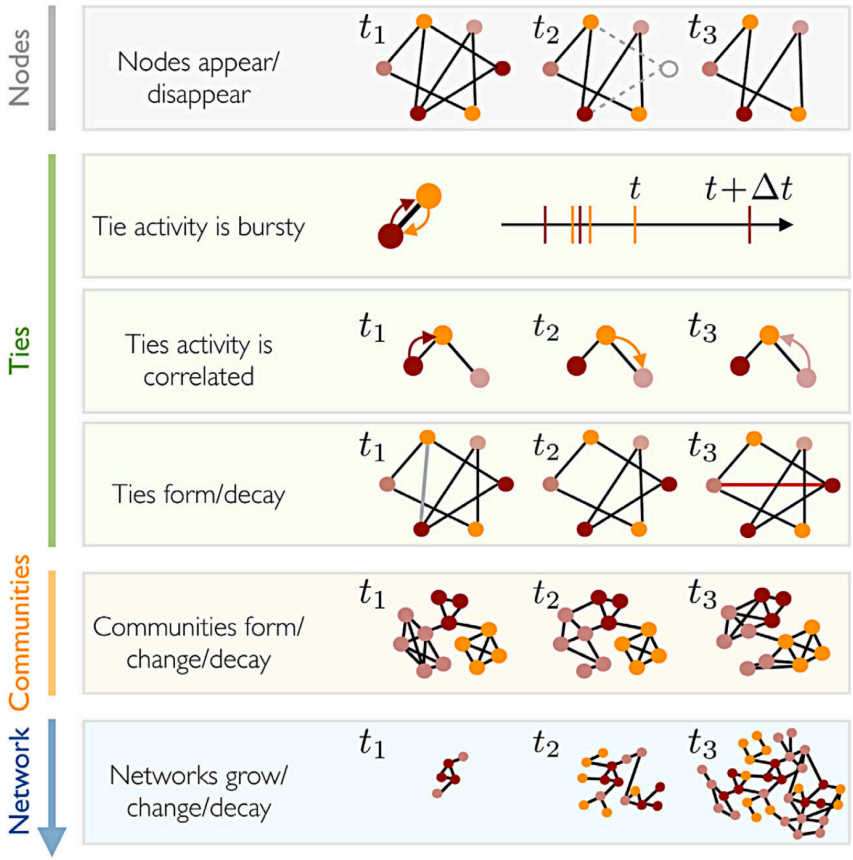


FIGURE 1.7: Visualization of different network dynamics visible on varying timescales. Figure reported from [MS15].

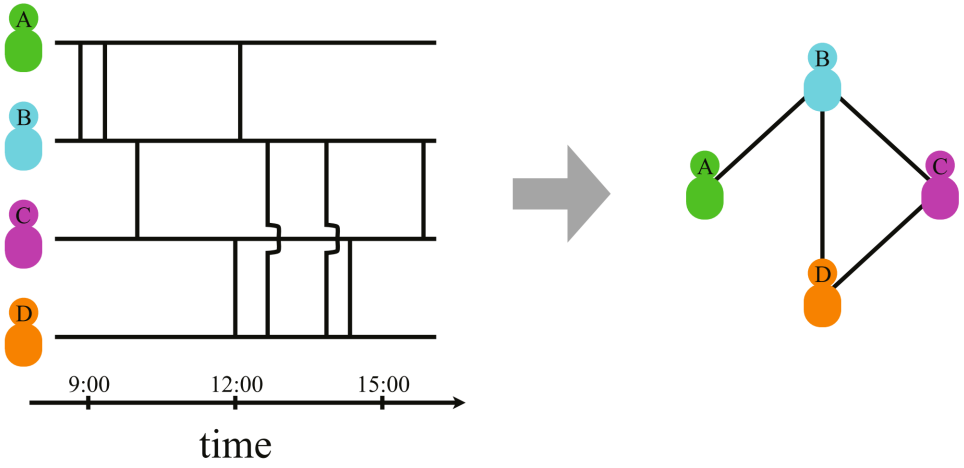


FIGURE 1.8: Examples of interactions on a temporal network. If D is infected, he won't be able to contaminate A through B because B never interacts with A again after having contacted D. When using the aggregated static version of the network, the ordering of interactions is lost and D can spread the disease to A. Figure taken from [MH13]

(see section 1.3.5), using it to model inter-event times helps us capture the heterogeneity of interactions timings. Once again, this comes from the high variance of power-laws.

Our bursty behaviour has an impact on epidemic spreading. Compared to Poissonian activity, burstiness yields a slow-down of spreading, although it can also accelerate it in the early stages [Vaz+07; Kar+10]. This is justified by the fact that longer inter-event times are allowed, which can stall the disease for longer than using uniform activities. Another interesting fact is that adding bursty activity to a scale-free network results in the presence of an epidemic threshold [MGK13]. As a reminder, this threshold vanished for large scale-free networks when using the Poissonian activity assumption. These particularities will be studied in more details in chapter 3.

1.6 Summary

This chapter started with the presentation of classical epidemic spreading models. The standard version of the most common models that are SI, SIS and SIR were first presented. These versions use only up to 3 compartments: susceptible, infected and removed. These simple models have the benefit that they can easily be solved analytically (or numerically in the case of SIR spreading), but their assumptions of perfect mixing and poissonian activity are sometimes too strong for them to provide useful results.

We therefore presented alternative models that dropped the perfect mixing assumptions, first by introducing subpopulations, and then by considering each individual as a separate node of a full-fledged social network. Three important network models were introduced. All three models generate small-world networks, but only the model proposed by Watts and Strogatz exhibits a high clustering coefficient, while the presence of hubs is only encouraged by the Barabási-Albert model.

Improving static graph models is important, but we can get even closer to the real-world by taking social dynamics into account. Real social networks evolve through time. First, individuals are born or die and the strength of social ties varies. Secondly, our social activities are bursty and often correlated (an interaction may trigger several other ones). Once again, it was shown that these aspects have an impact on spreading properties.

1.7 Evolution of Epidemic Spreading Analysis

As shown by the diversity of the literature, the study of epidemic spreading over social networks can be approached in many ways. Part of our work for this thesis was to understand the existing research, and to identify the key aspects of the domain. To summarize the evolution of spreading studies and models, we have classified 34 of them according to 4 major questions:

- Does the study consider a scale-free network?
- Does the study take network dynamics into account?
- Does the study consider the burstiness of our activities?
- Does the study rely on a real dataset (data-driven approach). For example, data can be used to infer a network, whether temporal or static.

This compilation of papers is too dense to be entirely described here and is instead available in appendix A. Table A.1 lists the papers themselves, while the results of the comparison are presented on table A.2. We present a visual summary of this survey work on figure 1.9 where the articles are numbered from 1 to 34, this ordering corresponding to the chronological order of publication. As shown on the Venn diagram, the first historical studies didn't consider any of the four aspects we listed. The scale-free question was the first one to be addressed in the first years of the 21st century. In the last ten years, researchers have analyzed the impact of different combinations of epidemic spreading properties.

The Venn diagram from figure 1.9 also carries an important information about the use of data in scientific studies of social networks, and of network epidemiology in particular. The studies that manage to combine several properties typically use a data-driven approach. As we have already mentioned, obtaining relevant data for such studies is not easy. This is one of the reasons behind our decision to build a simulator able to generate such datasets. The generated datasets can then be used to test some assumptions or validate theoretical conclusions through systematic simulation.

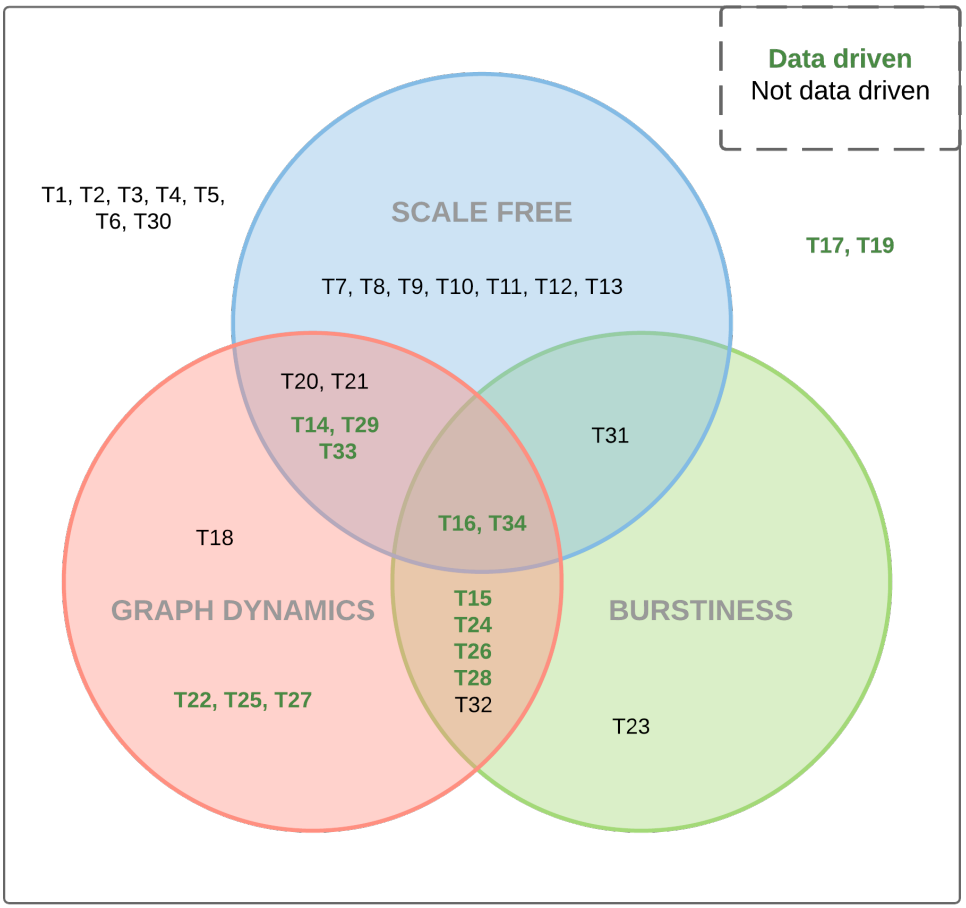


FIGURE 1.9: Visual summary of the comparison presented in table A.2. The labels reference articles from table A.1. Studies are numbered from 1 to 34 according to chronological order of publication. The older articles typically ignored burstiness and social network dynamics, and were generally not data-driven. In the early 21st century, models started to take the scale-free property into account. In the last few years, most publications have begun combining multiple aspects at the same time, often using a data-driven approach.

1.8 What's Next?

The models become more and more complex and can't always be solved analytically. A first solution is to start from the data. This is less flexible but allows to stay close to the reality. Another approach is to resort to simulation. In the next chapter, we introduce a flexible simulator allowing the study of epidemic spreading according to different assumptions.

In order to study the questions discussed in the previous section, different paths are possible. Figure 1.10 graphically illustrates several of these approaches in the form of a flow diagram. In particular, the simulation approach we have chosen to explore in this thesis is highlighted.

Studies usually start from a static network that is either generated from data, or directly from a theoretical model. In the simulator that will be presented in chapter 2, this generation of the world of the simulation is handled by a module called *"God"*.

A study can choose to ignore dynamic aspects of social interactions and run an epidemic spreading simulation using the static network directly. As we have already discussed, social dynamics may have a non-negligible impact on the behaviour of epidemics. This is why we use the second part of our simulator to generate a temporal network. Because this module generates social interactions in the world created by *God*, we named it *Nature*. Of course, if temporal data (*e.g.* tweets, emails, ...) is available from the start, it can be used to create the dynamic network directly. Another possibility is to use this data to calibrate some parameters of the simulator.

Once a temporal network is available, we can simulate an epidemic spreading on it. This time, the dynamics of social interactions will be taken into account.

In chapter 2, we will detail the inner workings of *God* and *Nature*. Because of our collaboration with Real Impact Analytics, *Nature* was designed to output a temporal network in the form of an ordered list of Call Data Records.

The reason that the epidemic spreading simulation is not included into *Nature* is that we didn't want to limit the simulator to epidemiological studies. Our goal was to build a generic tool that could be used whenever no data was available, or to test hypotheses.

In chapter 3, the flexibility of the tool will be put to good use to analyze how using different settings impacts epidemic spreading. We will first generate a static network with *God*. After this, we will create a temporal network representing the interactions using *Nature*. The last step will be the simulation of the spreading of a disease over the temporal network.

Using the simulator, we will study the impact of the diameter of an ER network on the spreading. The simulation will also allow us to track differences in epidemic thresholds between BA and ER networks, confirming the theoretical expectation that R_0 should decrease a BA networks get larger. Finally, we will investigate the impact of a specific kind of correlation between activities on the spreading and on the inter-event time distribution.

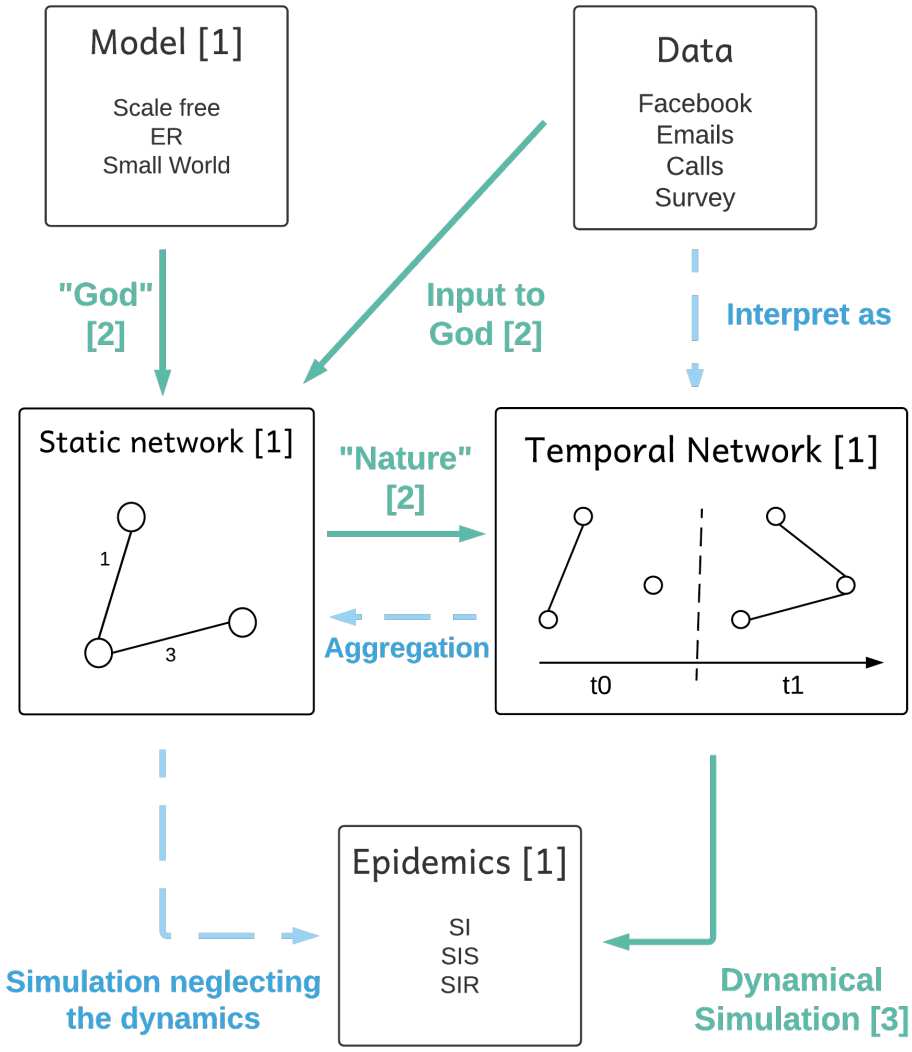
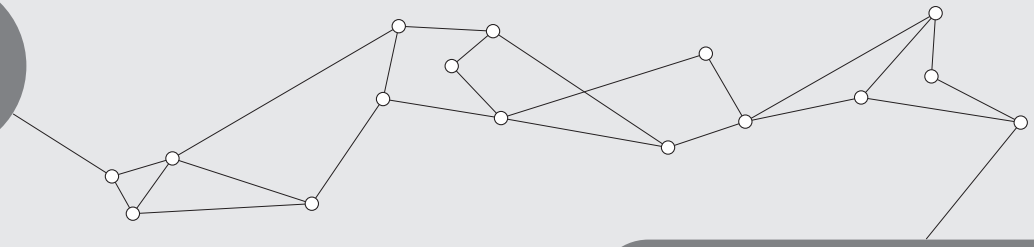


FIGURE 1.10: Flow diagram of different approaches to study epidemic spreading over social networks. The path followed in this thesis in represented in green, and the number in brackets denotes the chapter in which the subject is treated.

2



Architecture of the Simulator

2.1 Introduction

Epidemiology models have some shortcomings and, as they get closer to the reality, the mathematical models become harder or even impossible to solve analytically. However, they can still be simulated. The main difficulty lies in modeling and generating the interactions between individuals. In this chapter, we present the challenges of building a flexible distributed simulator. This simulator aims at generating contact records in the form of a temporal graph of the interactions over the social network.

This simulator allows to take different assumptions into account. The user can, for example, specify different types of static or dynamic graphs, different inter-event time distributions, different individual behaviours, and different kinds of correlations of interactions. In chapter 3, this simulator will prove to be a useful tool to study the impact of different hypotheses on the spreading of epidemics.

The simulation is decomposed into two steps. The first part handles the generation of the initial network and the determination of each individual's behaviour. The initial network may be generated using a predefined social network model, or directly from external data. This first module was named *God* because it creates the "world" of the simulation.

Once we have defined the network, we enter the second and last phase: *Nature*. This module generates the interactions between individuals, based on the world configuration provided by *God*. *Nature* is an event-driven distributed simulation, where each individual is represented by a distinct actor. This allows fine-grained behaviours, making individuals reactive to events and generating correlated interactions. For example, individuals could too dynamically change their acquaintances or switch between different interaction time patterns.

Nature was originally written in the context of our collaboration with *Real Impact Analytics*. A first proof-of-concept it was implemented at RIA during an internal 2-day hackaton. The final design of *Nature* is based on this early version, but has been improved in many ways. The output of *Nature* is composed of Call Data Records (CDRs), which contain metadata about interactions such as calls or text messages. In the real world, these CDRs are continuously logged by Telecom operators. However, most of the code is not specific to mobile phone networks, and can be reused to simulate any social network. Without loss of generality, we will consider CDRs as general interactions on a social network, in the sense that a call or a text message represents a contact between two individuals.

The chapter is organized in five parts. We start by introducing the world generation with *God*. The next section gives an introduction to distributed simulation and exposes the concepts needed to understand the choices made in the design of *Nature*. A reader familiar with the Actor model and with distributed systems may skip section 2.3 and go on to read the following section,

which details the design and architecture of *Nature*. Finally, we provide some benchmarks of our simulator to evaluates its performances.

2.2 God Architecture

God's role is to generate a description of each subscriber which provides information about the world to *Nature*. Among others, each subscriber is associated with a list of friends and an activity pattern. *God* can either generate this social network using one of the implemented models, or can read it from an external file.

2.2.1 Apache Spark Presentation

Our implementation of *God* relies on Spark. Spark is a project maintained by the Apache Foundation that was realeased in version 1.0 in May 2014. Its goal is to ease distributed computations by providing an API whose building block is the Resilient Distributed Dataset (RDD). RDDs are the primary abstraction provided by Spark and they support two kinds of operations: actions and transformations. Actions are used to perform computations on a RDD and get a single result back, while the output of a transformation is another RDD [Zah+10].

A typical use-case of Spark is to read data from a distributed file system into the cluster in the form of a RDD, before performing in-memory Map-Reduce computations on it and writing the result back to disk. In the case of *God*, we rely on the Hadoop File System (HDFS) for storage [Bor10].

2.2.2 Application to Network Generation

God provides built-in support for the generation of Erdős-Rényi networks and for networks with a LogNormal degree distribution. Since one of the key feature of the simulator is its flexibility, it is also able to read edge lists from external files. This allows users to use any arbitrary network in the simulator. In particular, we have used this feature to perform experiments on Watts-Strogatz and Barabási-Albert networks, generated using the *NetworkX* library.

Once a RDD representation of an edge list is available, we apply a Map-Reduce computation which is illustrated for a toy example on figure 2.1. A flatmap transformation is first used to make sure that both directions of each edges are added to the undirected graph. In this case, we only consider undirected edges. For some applications, simulating bidirectional social contacts could be too restrictive, and support for directed graphs is also possible. The final operation on the RDD is a reduce action, which results in an adjacency list representation, while also removing potential duplicated edges.

As explained in the introduction of this chapter, the simulator supports specifying custom behaviours for each individual of the network. These behaviours are specified in a JSON configuration file given to *God* on startup. A behaviour is associated to each node before persisting the whole network to HDFS where it can be read by *Nature*.

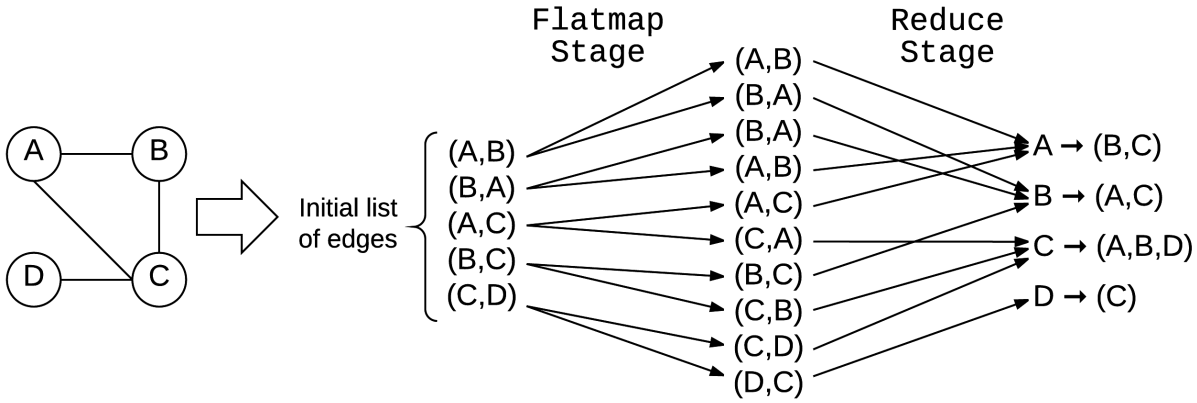


FIGURE 2.1: Toy example illustrating the Map-Reduce computation of an adjacency list representation from an arbitrary list of edges. The initial edges can either be generated directly by *God* using one of the supported network models, or taken as an input to support arbitrary networks.

2.3 Distributed Discrete Event Simulation: Theory & Concepts

Discrete event simulation is concerned with the modeling of systems represented by discrete states that are modified through discrete events. In the case of this thesis, the goal is to create a satisfying model of social interactions by representating each individual as a separate entity, that can communicate with other entities based on predefined behaviours.

When implementing a large-scale program today, a programmer cannot avoid dealing with concurrency. This is in part due to the rise of multi-core processors which has made parallel execution necessary to make full use of the hardware. Another factor is the generalization of distributed programming, arising from the fact that adding more computers to a cluster (horizontal scalability) now comes cheaper than increasing the power of a central machine (vertical scalability). This trend is likely to increase in the coming years, given that the semiconductor industry is scheduled to abandon its pursuit of Moore's law [Moo75; Wal16].

In the first part of this section, we introduce the Actor Model as a mean to manage concurrent execution. We then compare it with the popular alternative of Shared-State. The second part of the section gives an overview of the most important concepts in distributed discrete event simulation, with a particular focus on synchronization. A reader familiar with one of these topics should be able to skip the corresponding section without impacting his understanding of the rest of the text.

2.3.1 The Actor Model

Actors were introduced in [HBS73] as a universal formalism for computation, based on message-passing. In this thesis, we are interested in their use in addressing the synchronization problems that occur in concurrent or distributed programming. The building blocks of the models are actors, entities that exchange messages. An actor can send a message to any other actor for which it knows a reference. In most implementations, messages can contain any serializable data, including references to other actors. Upon receiving a message, an actor can update its internal

state, send one or more messages to other actors, and specify a new behaviour for handling the next incoming message.

Synchronous and Asynchronous Message-Passing

Messages can be sent synchronously or asynchronously. In the first case, a message is sent and the sender waits for a reply before processing anything else. In asynchronous message-passing (AMP), the sender never waits for a reply, and this reply will be sent as another message later on. AMP thus allows the processing of other messages while waiting for the arrival of an answer.

Synchronous message-passing (SMP) can in fact be thought of as a particular case of AMP. This was shown rigorously in [CBMT96] and an intuitive illustration is provided on figure 2.2. Using SMP, actors suspend when waiting for an answer, which prevents them from reacting to any incoming message. We can see that the order in which the exchange of messages occurs in the synchronous case is only one of the possible orders of the asynchronous one.

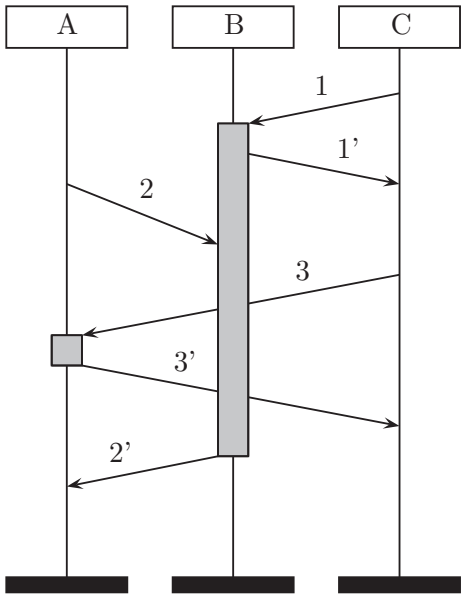
Any asynchronous actor can be transformed to a synchronous one by forcing the sender to block until its message is completely processed by the recipient. Van Roy and Haridi suggest a clean implementation of such synchronized actors which adds a dataflow variable to the message [VH04]. The dataflow variable is initially unbound, and the sender blocks until it becomes bound, which only occurs once the recipient is done processing the message.

SMP is often seen as conceptually simpler, because sending a message and waiting for an immediate answer is analogous to performing a standard method invocation. However, using synchronous message-passing exclusively is neither practical nor safe. Indeed, actors may end up spending most of their time waiting for other actors to process their messages. A single slow actor (maybe one running on slower hardware, or on a machine with high network latency) could become a bottleneck and slow the whole system down. Furthermore, synchronous messages can lead to deadlocks if there are cycles in the wait-for graph. In contrast, AMP prevents slow processing and deadlocks by never needlessly blocking.

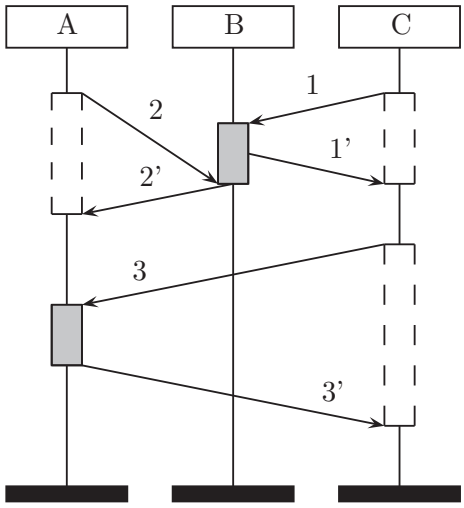
The asynchronous protocol is not perfect either. A first drawback is that it often makes the agents a bit more complex because some internal state has to be kept to remember which answer we are waiting for. Secondly, actor systems have to rely on "mailboxes" to handle messages that arrive when an actor is already busy. These mailboxes are buffers that obviously have a finite size, which can vary depending on the specific actor and the available hardware on a specific machine. Whenever a buffer becomes full, the actor system can either choose to make the sender wait or to discard incoming messages, therefore loosing transmission reliability. The second problem should be dealt with when building a fault-tolerant distributed system.

Comparing the Actor Model With Shared-State Concurrency

In the shared-state concurrency model, the different execution threads share part of the memory. This kind of sharing allows concurrent modifications of variables, which leads to wrong results. To prevent these problems, the model must provide constructs to perform atomic actions. Threads can use locks to ensure exclusive access to resources, or higher-level constructs such as monitors or transactions. One drawback of shared-state concurrency is that the programmer must explicitly manage critical sections of the program, which is error-prone. Moreover,



(A) Asynchronous Example



(B) Synchronous Example

FIGURE 2.2: Two possible message sequences illustrating the difference between synchronous and asynchronous message passing. Time flows downwards and single quotes are used to denote answers. "Activation periods" during which an actor is processing a message are filled in grey. In the synchronous case, actors wait for answers to their messages before they process anything else. These "suspension periods" are represented using dotted rectangles. Actors can receive messages during both activation and suspension periods, but these messages will be stored in mailboxes to be processed later on.

failures and deadlocks that ensue can be very hard to detect as they may not occur on each run of the program.

Actor's mailboxes are race-free by design [HO07], and most actor systems guarantee that successive messages from one actor to another that reach their destination always do so in the same order they were sent. This means that the communication channel between two actors guarantees that messages are delivered at most once in a first-in, first-out fashion. Because an actor treats messages sequentially, actors may safely maintain internal mutable state. The internal state of an actor is then only accessed through the exchange of messages, and the payload of each message is immutable to prevent any race-condition.

From a software architecture point of view, actors often provide a natural way of separating independent components, while remaining highly testable. Indeed, an actor can be tested in much the same way any function can by just sending messages and replying to queries. To prove correctness, one can simply show that each actor's behaviour is correct in isolation, and that interactions between actors occur as expected. Because the non-determinism of actors comes from the ordering of exchanged messages, it is possible to treat messages sequentially. The internal behaviour of an actor may therefore be written as a pure function, which makes correctness easier to prove [VH04].

Actors are often easier to reason with, because sending and receiving messages makes the causality between events more explicit, *i.e.* there is a clear distinction between the send and receive operations. Since messages can be lost, this distinction also has the benefit of making fault-tolerance more explicit. This is discussed in the section 2.3.1.

We don't mean to blindly advocate the use of the actor model over shared-state concurrency in every situation. As the Law of the Instrument pleasantly puts it: 'If all you have is a hammer, everything looks like a nail" [Mas66]. The two models promote very different kinds of programming techniques. Shared-State concurrency will be a good fit for data-centered programs, wherein the same data is accessed and altered by many threads. Actors will really shine when used to model autonomous entities that need coordination [VH04]. Our simulator is composed of many such entities, so our choice of the actor model makes perfect sense.

Available Implementations

The use of the Actor model has been popularized by the Erlang implementation, which was created in the late eighties at Ericsson to build highly concurrent and reliable systems. Many other languages, such as Oz and Scala, also have support for this kind of message-passing concurrency [Arm+96; VH04; OSV08].

Our simulator is implemented in Scala and relies on Akka, an actor system implemented on top of the Java Virtual Machine. Akka provides the following features [Lig16b]:

- Management of all underlying threading complexity and resource allocation: each agent runs into its own lightweight thread and the system manages a threadpool inside which all actors are scheduled.
- Lower level control, *e.g.* over the threadpool, is provided through configuration files.
- Network transparency: the programmer doesn't need to deal with machine addresses, or with the transmission of messages over the network.

- Lightweight actors: up to a million actors for each gigabyte of heap memory. Of course, these numbers are valid for Akka's raw actor. Our actors will have internal memory which will result in much lower numbers.
- Provide testing framework to test each actor independently
- Fault-tolerance through hierarchy of actors
- Optional state persistence allows actors to restart after a node migration or a JVM crash.

Fault-tolerance

Even though fault-tolerance is not the focus of this work, it remains a very important problem to consider when dealing with distributed systems. There are many possible sources of failure: JVM crashes, network failures, power outtages or faulty electronic components, to name only a few.

Akka takes inspiration from Erlang on the fault-tolerance front. It uses supervision and applies the "Let it crash" philosophy [Lig16a]. When an actor spawns a child actor, it becomes its supervisor. The top-level actors have a single supervisor called the guardian. Upon failure of an actor, the actor system notifies the parent which can react according to a predefined strategy. Depending on the kind of failure, the parent actor may take appropriate action and then simply resume the actor, restart it, stop it completely, or fail itself and propagate the error to its own supervisor. In some situations, the failure of a child may also force the supervisor to restart all its children, which is referred to as the "All for one" strategy.

During this thesis, we didn't implement all the mechanisms required for fault-tolerance to work smoothly. We did, however, make sure to follow Akka's guidelines. Supervisor's default strategy when dealing with the failure of a child is to restart it. To avoid cycling between restarts and failures, a child is only restarted a given number of times before the supervisor gives up and kills it completely. This scheme is sensible, but may not work in all cases. Our design, presented in section 2.4.4, defines a clear actor hierarchy. Since the supervision hierarchy is already established, it is possible to refine this strategy in the future in places where it seems too crude, without breaking the rest of the logic.

2.3.2 Approaches to Discrete Event Simulation

We should be careful when discussing time in event-simulations. Indeed, the term is ambiguous because time might refer to several different concepts [Fuj00]:

Physical Time: The time in our physical system which, at least for the purpose at hand, can be considered absolute.

Wall clock Time: Analogous to the physical time, but only starts elapsing from 0 when the simulation begins.

Simulation Time: Abstract discrete time used by the simulation to represent the physical time. The duration of the intervals in simulation time has a correspondence to the durations in the physical time.

Although the simulation time represents the physical time, simulating a unit of simulation time can be faster or slower than a unit of time. For example, simulating one second of a Tsunami could take more than one second.

There are many possibilities to handle simulation time in distributed systems. In this section, we compare time management in distributed discrete simulation systems along three distinct axes: the progression of time, clock centralization and synchronization.

Time-driven *vs.* Event-driven

When it comes to the way time progresses, we can distinguish two types of discrete event simulations [FT98].

Some simulations are time-driven: they advance the timestamps by a fixed, predefined increment Δt. High value of Δt will result in a fast simulation with poor precision, whereas low value will increase precision by trading off performances. Despite the need for a sensible value for Δt, which is not always easily determined, another problem occurs for simulations that consist of events that are irregularly spread over time. In this case, actors will spend a lot of time needlessly incrementing their timestamp.

A better solution would be for such actors to directly increment their timestamp to the time of their next event. This is exactly the idea behind event-driven simulation. In this approach, the simulator maintains a priority queue of events to occur, ordered by ascending timestamps. The simulation proceeds by repeatedly popping and processing the next event, until the maximal timestamp is reached, or the queue becomes empty. While processing an event, other events may be added to the queue, and some events from the queue can be discarded.

Event-driven simulations often yield better performances because they avoid useless polling of actors, thereby reducing both actors load and network traffic due to constant time increment messages. The cost of performing an event-driven simulation amounts to the cost of maintaining a priority queue of future events, which is generally manageable. Time-driven simulations will still be more efficient in the rare cases where most actors are expected to react at every time steps.

Centralized *vs.* Decentralized Clock

When a single entity is responsible for giving and maintaining time in the distributed system, we call it the central clock. Even though all actors share the same time scale, it doesn't mean that a centralized clock is a requirement. It is also possible for each actor to maintain its own clock, and to decide how to progress through time on its own. A major advantage of this design is that the failure of a single actor doesn't bring the whole system down, as would be the case if a centralized clock actor crashed. Besides this fault-tolerance aspect, this approach can also speed the whole system up in cases where the centralized clock would have been a bottleneck.

Conservative *vs.* Optimistic Synchronization

It is getting clear that managing time is a major problem in distributed simulations: we want events that occur in a specific order in the physical system to take place in the same order in the simulation. In other words, we want each actor to process each event in timestamp order, or at least appear to do so from the outside. This requirement is referred to as the Local Causality Constraint (LCC), and it is the role of synchronization mechanisms to ensure it is respected. As was shown mathematically in [Mis86], if local causality is established, the results produced by a concurrent execution will be identical to those produced by a sequential execution. There are two main approaches to synchronization: conservative simulation or optimistic simulation [Fuj99; Zim08].

The goal of the conservative approach is to ensure that causality is respected at all times for all actors by making each of them wait until we are certain that all messages that he might receive have been processed. This blocking of all actors at a same point in simulation time is called a synchronization barrier [Fuj00]. Conservative agents are safe, because they avoid violating the LCC in the first place, but they can lead to slow processing: the system is only as fast as its slowest actor. This drawback will especially manifest when actor loads are unbalanced, since most actors will be idle while waiting for the highly loaded actors to finish processing their events [FT98].

The optimistic approach is based on the fact that, although respecting the local causality constraint is sufficient to avoid causality errors, it isn't strictly necessary. Indeed, some events that occur for the same actor may be completely unrelated and can thus be simulated sequentially in any order, without causing any causality glitch. In optimistic systems, actors try to take steps in the simulation even though they may lead to causality errors. This means that an actor will try to simulate steps with timestamp $t' > t$, subsequent to their current timestamp t [Zim08].

An error will be detected if the actor receives an old event with timestamp $t \leq t_{old} < t'$. This event can't be processed directly because it is out of order. In order to process the event, the actor first has to rollback its state, and undo all operations with timestamps $t' > t_{old}$. These rollbacks can be very costly, because they imply sending "anti-messages" to cancel any message that was sent by the actor while it was too optimistic. Actors that received such messages must in turn perform a rollback. Such causality error chains can have a strong negative impact on performances. To be able to perform rollbacks, actors are also forced to maintain some history of their state which increases memory pressure. However, this is generally not that much of a problem because of the existence of techniques that are able to safely erase most of the history on a regular basis [Fuj99]. To avoid suffering from excessive optimism, which would cause frequent rollbacks, most systems limit themselves to a simulation window of $[t, t+W]$, which means that a process can advance at most W timestamps above its last safe timestamp t.

To sum things up: conservative methods avoid LCC violations at the cost of making faster actors wait for slower ones, while optimistic techniques take the bet that no ordering mismatches will occur, and risk being forced to undo simulations steps in order to recover from such causality mistakes [FT98]. In general, the optimistic approach is very efficient in simulations where actors are mostly independent, and causality errors will be rare. In other cases, being overly optimistic will come at the cost of painful rollbacks that will end up being slower than using the safer conservative approach.

2.4 Architecture of Nature

To understand spreading processes over a population equivalent to a small country and to study how global complexity emerges from local behaviours, *Nature* has to be able to simulate the interactions of hundreds of thousands of individuals. In order to scale reasonably, the system was designed as a distributed system running a discrete event-driven simulation. We will start by exploring the different choices that were made, in light of the theoretical concepts presented in the previous section. Subsections 2.4.4 through 2.4.6 will then detail the communication of actors and the scheduling of events. As all the presented diagrams translate almost directly into code, this second part can serve as a high-level reference to whoever wants to read and understand the Scala implementation of *Nature*. We finally end our presentation of the architecture of the simulator with some benchmarks of the performances of *Nature*.

2.4.1 The Real World Problem

Actor-based modelling is a natural choice when it comes to simulating many independent entities, as it maps particularly well to the real-world where individuals act independently from most others. Indeed, each individual only has social relations with a tiny fraction of the entire population, and ignores the remainder. This is probably the reason why the actor model has been one of the most successful in social science simulation [Cas+09].

As was mentioned before, our goal is to be able to generate Call Data Records (CDRs) by simulating each subscriber and making him interact with other individuals. We want our work to be as generic as possible, so it can be extended for purposes other than generating CDRs. This is why we don't just simulate subscribers, but also operators, and the towers that relay communications. One could then modify our simulator to analyse the load of towers in a given region, to model the consequences of the failure of a tower on the network, to help design marketing campaigns for an operator, or to research mobility patterns. This need of flexibility caused us to aim at genericity rather than raw performances. In chapter 3, we discuss how the generated CDRs can be used to study epidemic spreading.

2.4.2 General Purpose Actors

In this section, we briefly present the different actors that are needed to model the world of the simulation, regardless of the synchronization techniques we use.

Individuals represent the subscribers of an operator. They maintain a state composed of their own immutable id, their current tower's id, their operator's id, and a description of their current friendships, *i.e.* the set of subscribers they can interact with. We also allow to specify call patterns, mobility patterns and a dynamic model of how the friendships will evolve.

Towers have an id, a name and a geographical position. To model reality, each tower also bundles a set of cells, which roughly correspond to chunks of land area it services. Multiple towers are, of course, allowed to service a single cell.

Operators currently maintain no state, and don't interact with towers or subscribers. We added them for completeness because they do play a role in the real world. They already

participate in the synchronization mechanism, and could thus be easily modified for any experiment that would require them.

Many parts of the different actors can be customized. In particular, one can set probability distributions for the different behaviours such as calls or mobility for actors, and probability of dropping calls for towers (see section 2.4.6). Some default behaviours have been defined, and can be set up using a JSON configuration file. The design is flexible: it is easy to modify the code to add new custom behaviours or actors. For example, we could imagine randomly creating new social relationships between subscribers that are geographically close to each others.

2.4.3 Design Decisions

Now that the different design possibilities and the context have been presented, we will explain our design choices. We chose to build *Nature* as an event-driven, conservative simulation system with a centralized clock.

The reason we opted for the event-driven approach is very simple: in the context of phone communication, we expect most actors to stay idle at each iteration. For the sake of the argument, let's assume that each one of the n actors places 10 calls a day, uniformly spread through time. Since we use time steps of one minute, there are 1440 ticks a day. An actor is thus expected to place a call only $\frac{10}{1440} = 0.7\%$ of the time. In practice, polling each subscriber at each timestamp is therefore completely useless, and an event-driven simulation will yield much better performances.

Synchronization techniques are well-studied, but there is no clear consensus as to which of conservative or optimistic one should choose in general [Fuj99; FT98]. This choice is in fact largely case-specific. As we've said before, the conservative approach has the disadvantage that it forces all actors to wait for the slowest one. Optimistic synchronization allows actors to advance at their own pace, at the cost of having to rollback sometimes to fix causality errors. In *Nature*'s case, we know that causality will play a central role. Indeed, once a subscriber accepts a call, he cannot take or place any other ones for the entire duration of the said call. This means that one subscriber placing a call will influence the behaviour of the callee, possibly preventing him from calling someone else. If we used optimistic synchronization, we should expect such causality chains to be frequent, and we know it would be inefficient to undo them. Furthermore, optimism comes at the price of the memory needed to store sufficient information to be able to restore the state of actors. Using this approach, we would need to deal with IO operations very carefully, as they cannot be undone as easily as state modifications that occur in RAM. For our purpose, the optimistic approach is thus much harder to implement correctly and maintain. We expect that it wouldn't be more effective than the conservative counterpart, which is why we opted for the latter. However, from a performance viewpoint, only experimenting with both techniques could designate a clear winner.

One major drawback of conservative approaches is that they make fast actors wait for slower ones. This can be a problem when the event rates are very unbalanced between individuals. However, this problem can be mitigated if we use conservative synchronization in conjunction with event-driven time progression. In this case, only actors that must process an event are contacted at a given timestamp. The actors that are active during a time step generally have comparable processing to perform, which greatly helps reduce imbalances in event frequencies.

Our last design choice was to choose centralized clocks over decentralized ones. The major

drawback of using both a decentralized clock and conservative synchronization is that it forces us to use a fixed topology for the social network. In the decentralized approach, a subscriber A maintains his own clock and only waits for his friends before skipping from timestamp t_A to timestamp $t'_A > t_A$. The problem is we can't dynamically add new a new friend B to subscriber A, because the timestamp t_B may be larger or lower than t_A at the time of the creation of the friendship. This would cause a causality error which is forbidden by conservative synchronization. Our only choice would be to use optimistic synchronization, with the drawbacks we mentioned previously.

Two main advantages of decentralized clocks are performance (no central clock that could become a bottleneck) and fault-tolerance. In *Nature*'s case, the fact that we use event-driven simulation results in the master clock doing very little computation at each timestamp, only sending one tick to each node of the cluster and waiting for the corresponding `Done` message to come back (see section 2.4.4). Fault-tolerance can also be dealt with using a centralized clock. In case the master clock dies, it can be restarted by the general supervisor of the cluster (the guardian mentioned in the section 2.3.1). Finally, the centralized clock approach is far easier to test and program with.

2.4.4 High-level Architecture

The high-level architecture of *Nature* is presented on figure 2.3. The master clock actor acts as the centralized clock we introduced in section 2.3.2. The master communicates with other clocks, called the slaves. There is one slave clock on each node of the cluster, and it is responsible for the synchronization of all actors that are located on the node. As figure 2.3 shows, slave clocks send time messages to specific clocks, which relay them to subscribers, towers and operators. All messages are exchanged asynchronously.

The synchronization technique used by *Nature* is called a tree barrier, also sometimes referred to as Beta synchronization [Lyn96]. Figure 2.4 illustrates its inner workings. At a given timestamp t, the master issues a `Tick(t)` message to all the slave clocks, which broadcast it to the specific clocks. Because we use an event-driven approach, the specific clocks maintain an ordered list of the events scheduled by their children, and are able to send the `Tick(t)` only to actors that will be active during timestamp t. The work done by active actors is discussed in the section 2.4.5. When an actor is done for the timestamp, it replies with a `Done` message. Whenever all their supervised actors are done, the specific clocks send a `Done` message to their slave clock, which relays it to the master clock. Once all the nodes of the cluster are done, the master clock asks all the slave clocks for the time of their next event. The reason for this message exchange will become clear in section 2.4.5. The master then finally skips to the next timestamp and the simulation continues. The "tree barrier" name comes from the fact that each level of the tree implements a barrier, waiting for its children to complete their processing of the timestamp before notifying its own parent. The master clock is easily implemented as a finite state machine, as illustrated on figure 2.4.

Specific clocks are not strictly necessary from a synchronization point-of-view, because slave clocks could communicate directly with subscribers, towers and operators. Their existence comes from the fact that that an early version of the simulator developed at Real Impact Analytics wasn't event-driven and all the subscribers exchanged messages with the slave clock at each timestamp. This massive processing of messages caused a pause at the beginning and at the end of each tick, with a single active thread. Indeed, only the master clock was active, while it processed all the `Done` messages. To remedy this, our first attempt was to add intermediary

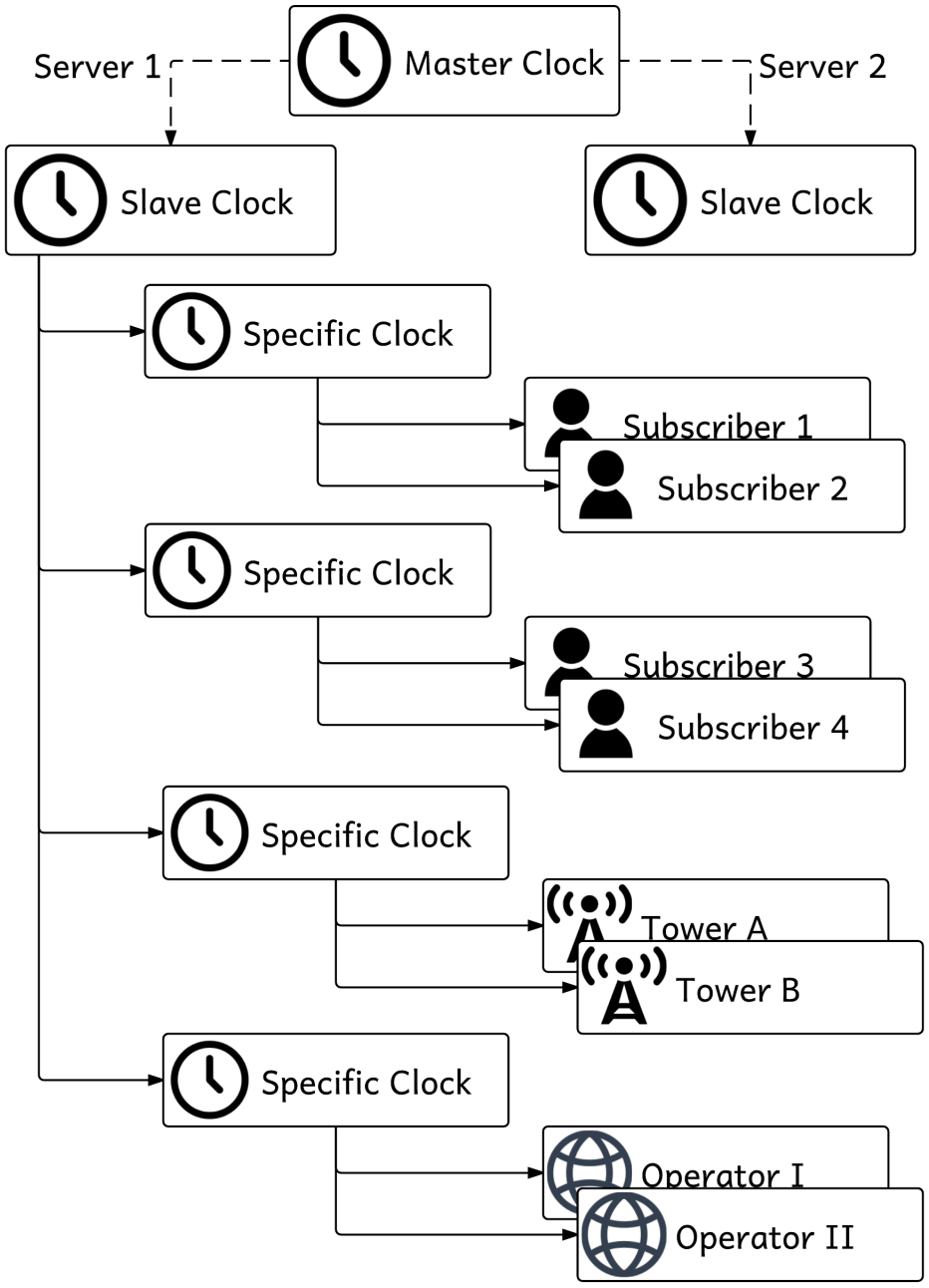


FIGURE 2.3: Global architecture of the discrete event simulation. There can be several servers, and the master clock can either reside on its own cluster node, or inside one of the other servers. The subscribers are evenly attributed to servers and specific clocks. It is also possible to run the simulation locally, on a single server.

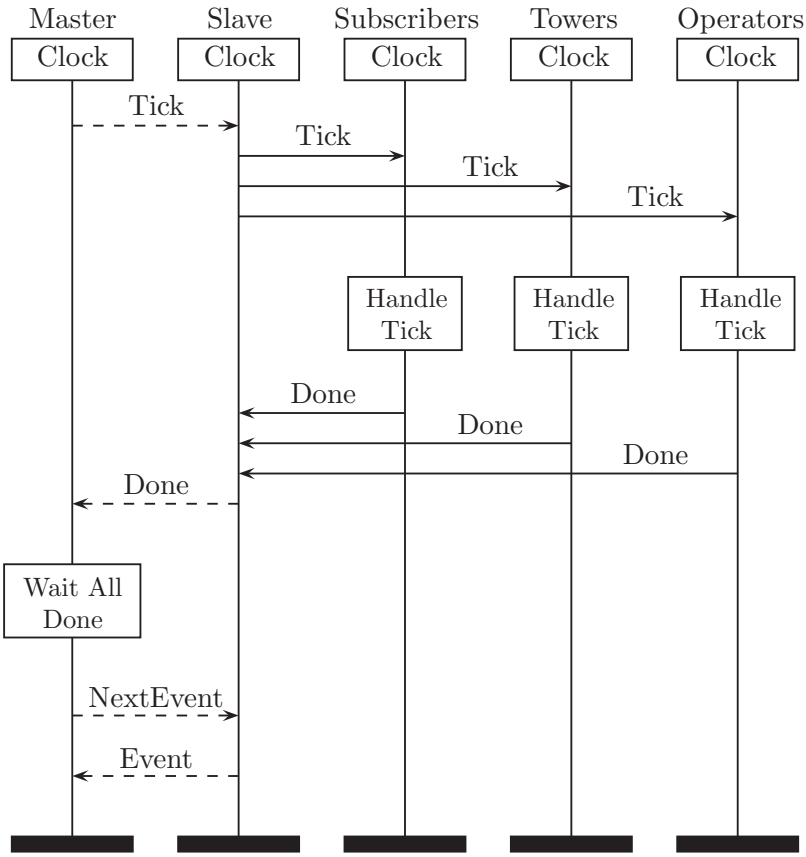


FIGURE 2.4: Communication protocol between clocks for one time step. The figure only shows interactions between the master clock and a single slave clock. In practice, there is one slave clock for each server. Time flows downward and the time it takes for a message to reach its destination is ignored (horizontal messages). Messages that may travel across the network are represented with a dotted arrow.
We ensure their delivery by acknowledging and repeating them if need be.

clocks to distribute the load across multiple actors, and thus multiple threads. In the current version of the simulator, specific clocks were kept because they worked fine and are occasionally useful in case of peaks of activity at given timestamps. Their presence doesn't create much overhead as they merely relay messages on the same node (no object creation and no network latency involved) and store events on a priority queue.

2.4.5 Communication Protocol

In this section, we discuss how active actors communicate to start calls, and how failed calls are handled.

Figure 2.5 shows the various messages exchanged by actors during a successful call. First, a a specific clock transmits a `Tick` message from the slave clock to an actor that is scheduled to be active during the current timestamp. When an actor receives a `Tick`, it checks that he is supposed to act at this timestamp and then creates a new `Call` message. By default, the callee

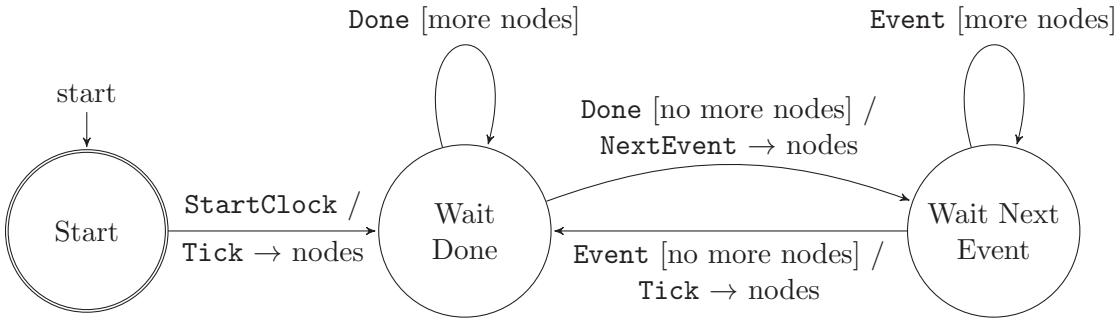


FIGURE 2.4: Finite State Machine implementation of the master clock. Input messages and actions are separated by a slash, and conditions are expressed inside brackets.

is chosen randomly among the caller's friends. The `Call` message contains many information, among which its duration and the callee's id.

The second step of the protocol is the forwarding of the call to the caller's tower. The steps that involve towers add some overhead to the simulation but, as we've mentioned before, they are needed for the simulation to stay close to the way calls are dealt with in the real-world. For example, this protocol allows the simulation of tower failures, or the rejection of calls depending on the current load (number of subscribers serviced by the tower). The scenario of a rejection of a call is represented separately on figure 2.6.

Once the tower receives the `Call`, it sends a query to the callee to ask information such as the callee's operator (this information may be needed, *e.g.* if billing needs to be simulated) or the tower that currently services the callee. This interaction doesn't really occur in the real-world, where a centralized entity exists, that knows this information for all subscribers. We decided not to model this entity in the simulation because it would have to maintain a huge data structure containing information about each subscriber, and subscribers would need to notify the entity whenever part of the information changes.

The tower then relays a `ConnectCall` to the callee's tower, which may in turn reject the call (*e.g.* to simulate a network failure), or transmit it to the callee. If the callee is available at the time of the call, he will accept the call by scheduling his next event at the end of the call. Since we're using a fully event-driven protocol, it is possible that both the callee and its specific clock are inactive during a given timestamp. To schedule its next event, the callee therefore needs to send an event message to its clock. The callee must also wait for an acknowledgment to come back before sending an `OkStatus` message back to its tower. The `OkCallStatus` will be relayed by the towers to the caller. Both towers also log the CDR of the call because they can belong to different operators: a CDR can be considered valid if both towers have logged it. Depending on the settings of the simulator, the CDRs are output in a file directly or through Kafka. Apache Kafka is used as a distributed log and insures that all CDRs are reliably written to disk, while maintaining good IO performances.

As soon as the caller receives the call status, he schedules his next event to the end of the call. The caller and the callee will thus both receive a `Tick` at the end of their call which will allow them to schedule their next event, as explained in the section 2.4.6. Let us assume that the callee was in an inactive period upon receiving the call, and that this period was scheduled to last until time t_{old}. If the call lasts until $t_c < t_1$, then the callee has an opportunity to reschedule

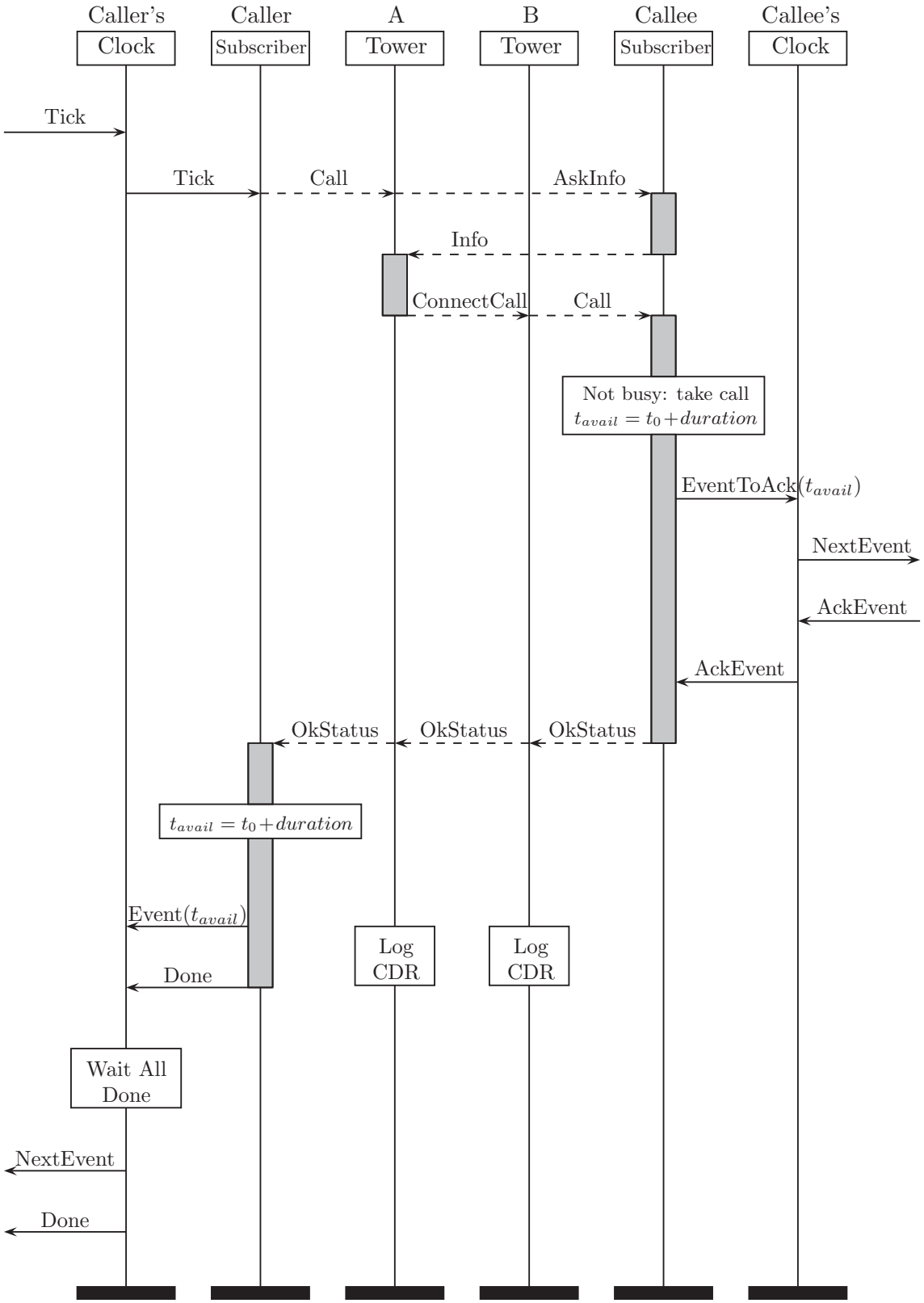


FIGURE 2.5: Protocol of a successful call from `Caller` to `Callee`. Possible synchronization (`Tick`/`Done`) messages between the callee and his supervising clock are not represented to avoid cluttering the diagram.

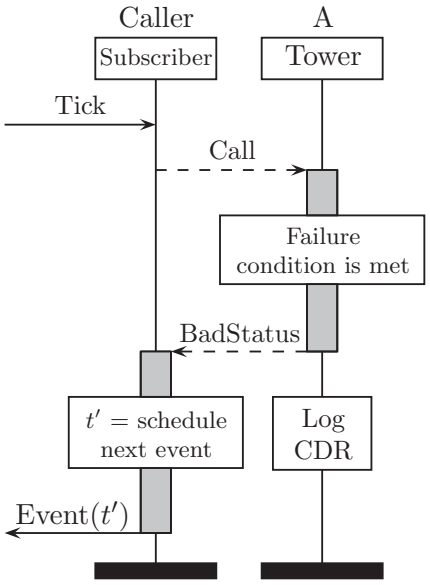


FIGURE 2.6: Rejection of a call by the caller's tower. The failure condition may *e.g.* depend on the number of subscribers the tower services. The call can also be rejected by the callee's tower (not shown here).

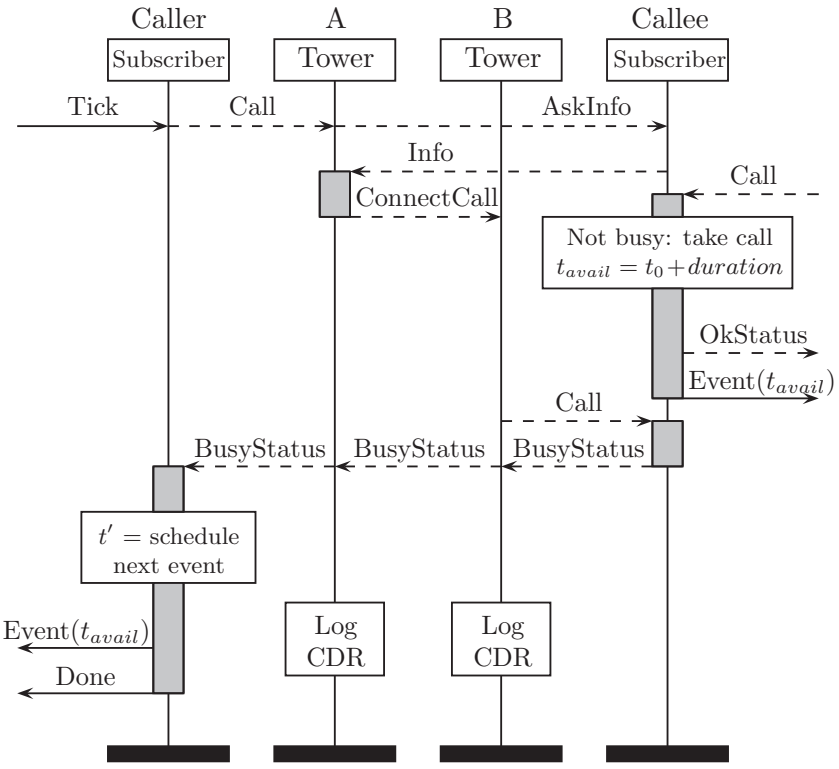


FIGURE 2.7: Another `Call` reaches the callee before the one from the caller. A busy call status is received by the caller. The CDR is logged anyway as it may carry interesting information.

his next event to time $t_c < t_{new} < t_{old}$. This causality between the call received from the caller and the early rescheduling of the next event of the callee can have consequences on the shape of the distribution of inter-time. These consequences will be explored more in depth in chapter 3.

The processing of the current timestamp is terminated by the caller with a `Done` message to his supervisor. The supervisor clock will then wait for all its active subscribers to do the same, before computing the next event to occur over all of them, and sending it to the slave clock.

We mentioned that the status message from the callee had to be sent after the callee's clock had confirmed the reception of the new event. The reason for this ordering is simple: the actor system doesn't guarantee that messages from a same sender to different destinators arrive in the same order they were sent. If the callee doesn't wait, the status message can come back to the caller before the callee's clock has scheduled a new event, and the master clock may be able to send the next `Tick` before all actors are done for the current one. An important aspect is that the callee does not block during its waiting period, so he is still able to handle other messages.

We should note that it is also possible for the callee to receive a `Tick` at any point during the exchange of messages. This can cause him to schedule another call before the first one reaches him. It is also possible that another caller starts a call with the callee before the first call reaches him, as illustrated on figure 2.7. In both scenarios, the second call to reach the callee is rejected, and the corresponding caller schedules a new event.

2.4.6 Event Scheduling

Event-driven simulation relies heavily on distribution sampling. After each of its events, an actor must inform its supervising clock of the next time it wishes to be woken up. In our case, anytime a subscriber is done passing a call, it sends a message to its clock containing its next planned call. It makes sense for an individual to call at different rate, depending on what time it is: fewer calls are generally placed during the night than during the day! In order to have a flexible system, we provide two ways of customizing the call scheduling.

Sampling Inter-event Time Distributions

A first possibility is to specify the inter-event time directly. If the current time is t_0, and we get an inter-event time of Δ by sampling the distribution, we can simply schedule the next event to occur at time $t = t_0 + \Delta$. This approach allows to use any known distribution, and *Nature* has built-in support for a few well-known ones, such as the Power-Law distribution mentioned in chapter 1. As a reminder, this distribution can be written as follows, for some $\alpha > 1$ and an appropriate normalization factor C:

$$P(\Delta) = \begin{cases} C\Delta^{-\alpha} & \text{if } \Delta > \Delta_{min} \\ 0 & \text{otherwise} \end{cases}$$

One classical technique to sample any discrete distribution $f(\Delta)$ is called the inverse transform method. The method starts by choosing a number p uniformly at random in $[0, 1]$. If the

cumulative distribution $F(\Delta)$ of $f(\Delta)$ can be inversed, the sampling is performed using

$$\Delta = F^{-1}(p)$$

.

For some distribution, it can be hard or impossible to express $F(\Delta)$ or $F^{-1}(\Delta)$ in a closed-form. If that is the case, one possibility is to use $f(\Delta)$ to explicitly build an array of the values of $F(\Delta)$. To limit the size of the array, we will generally only include values of Δ belonging to an interval $[a, b]$ such that

$$\sum_{\Delta=a}^{b} f(\Delta) \geq 1 - \epsilon \quad \iff \quad F(b) - F(a) \geq 1 - \epsilon,$$

for some small $\epsilon > 0$. Once the array has been computed, a sample can be drawn by performing a binary search for p inside it, which runs in logarithmic time with respect to the size of the array.

Activity Patterns

It is not always intuitive to use inter-event time distributions to specify subscribers behaviours. To allow fine-grained customization, the second approach lets users of the simulator provide arbitrary activity patterns. The system is then able to use the activity patterns to compute the next event directly.

An activity pattern is a discrete function that associates each timestamp to the probability of executing an action (*e.g.* a call) at this timestamp. This is a generalization of a Bernoulli process, where the probability of acting is not constant but instead depends on the timestamp t. We should point out that an activity pattern is *not* a probability distribution.

To formalize the concept, let's define E_t as the binary random variable that is equal to one when an event occurs timestamp t, and equal to 0 otherwise. We define the activity pattern $e(t)$ as the probability of an event occurring at time t, that is $e(t) \triangleq P(E_t = 1)$.

For a given actor of the event-driven implementation, we need to be able to schedule the next event that will occur, knowing the current timestamp t_0. To achieve this, we have to model the probability of the next event occurring at time $t > t_0$.

$$\begin{aligned} n_{t_0}(t) &\triangleq P(\text{next event occurs at } t \,|\, \text{current time } = t_0) \\ &= P(E_{t_0+1} = 0, \ldots, E_{t-1} = 0, E_t = 1) \\ &= P(E_t = 1)\, P(E_{t_0+1} = 0) \ldots P(E_{t-1} = 0) \quad \text{(independence of events)} \\ &= e(t) \prod_{\tau=t_0+1}^{t-1} (1 - e(\tau)) \end{aligned} \tag{2.1}$$

One validation of equation 2.1 is that, when the activity pattern is a constant $e(t) = p$, the

inter-event time follows a geometric distribution. Unsurprisingly, the expression is not really dependent on the value of t_0, and can be expressed in terms of $\Delta t = t - t_0$, to get the inter-event time distribution:

$$n_{t_0}(t) = p\,(1-p)^{t-t_0-1} \rightarrow n(\Delta t) = p\,(1-p)^{\Delta t-1} \sim Ge(p)$$

An example of a realistic daily activity pattern for a subscriber is given on figure 2.8a. Because $e(t)$ is non-constant, $n_{t_0}(t)$ varies with t_0. Equation 2.1 was used to compute the distribution of the next event for $t_0 = 11$ a.m. and $t_0 = 6$ p.m., which are displayed on figures 2.8b and 2.8c. We can note the small dip induced by the "lunch break" from 12 p.m. to 2 p.m. on figure 2.8b and the influence of the night is clearly visible on figure 2.8c.

We should note that the timescale and the initial timestamp are both simulation specific. In this example, we applied sampling to call distributions, so we interpret activity patterns as call patterns, and t_0 as midnight. We could also have defined a weekly pattern, with less call activity during the weekend, and set t_0 as 7 a.m. The granularity of the simulation was also arbitrarily set to one minute. This means that the computed distributions are actually discrete, but they were represented as continuous for clarity.

It would have been hard to describe these event distributions directly without using activity patterns. Using them makes the customization of behaviours very intuitive. For even more flexible scheduling, it is also possible for subscribers to dynamically switch between different activity patterns in reaction to certain events.

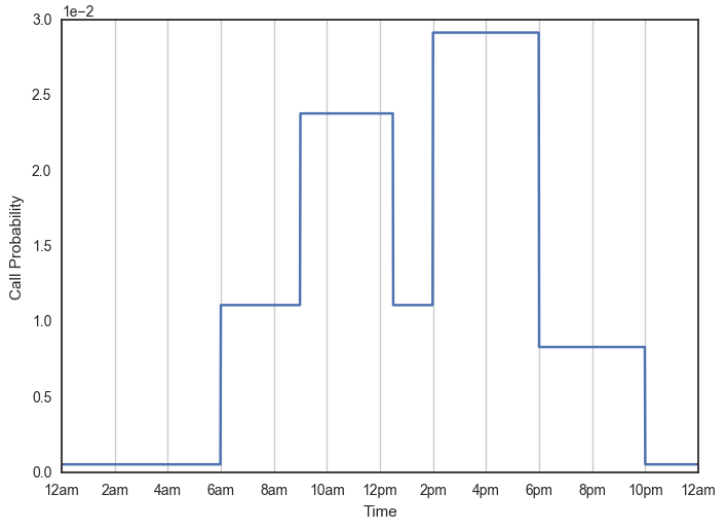
Dealing With Dependencies

To obtain the final analytical expression for $n_{t_0}(t)$ in equation 2.1, we assumed the independence of the events. Depending on the user behaviours, we are trying to model, this assumption may not always be verified. For example, taking the burstiness into account can make events dependant on each other: placing a call can be the cause of the next call. A model where a call to an actor can trigger another call in reaction, *e.g.* if the actor relays an information that he just received, would also violate the independence assumption.

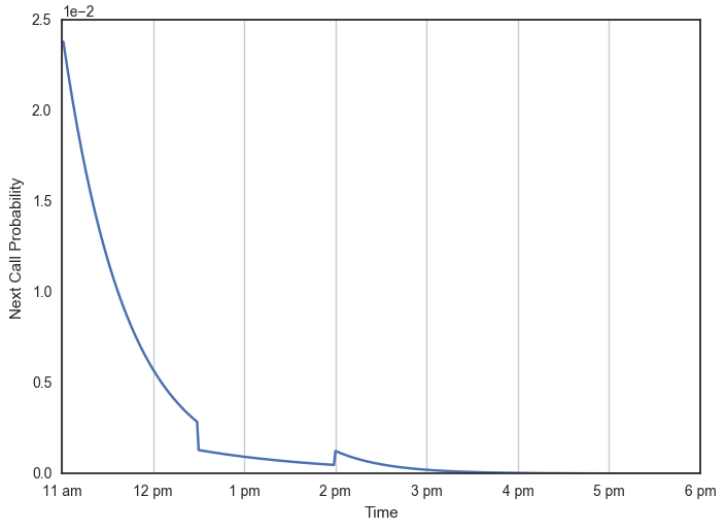
Fortunately, there are ways around this problem. One possibility is to dynamically adapt the call distributions during the simulation. We can create burstiness by increasing the activity pattern of an actor whenever it passes a call. This would increase to a higher probability of passing a call in the next few timestamps, which would lead to the presence of "bursts" of calls [Mal+08; Vaz+06]. Causal chains of calls can be dealt with in the same way, by increasing the activity of an actor after he receives a call.

Starting in Regime

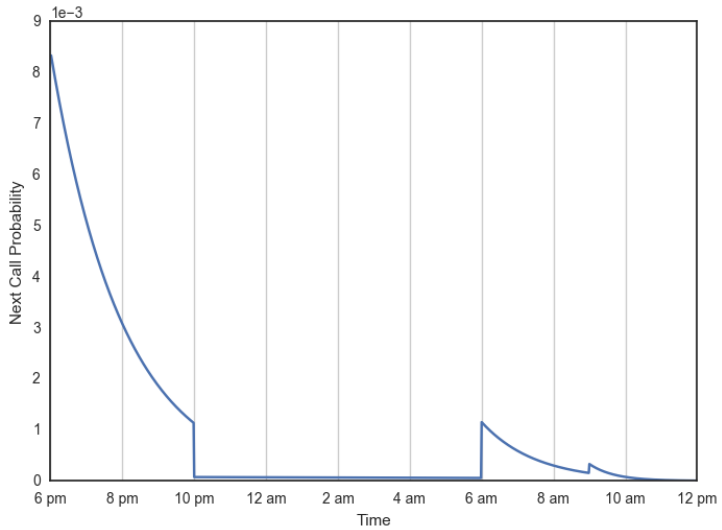
One challenge when designing a simulation system is to start it in regime. We can't simply contact all subscribers at time t_0 and sample the inter-event time distribution, because this could lead to biased results. Indeed, if the inter-event time distribution has memory, like is the case with bursty activity, we cannot assume that the last event of all subscribers occured at t_0 when scheduling their first event of the simulation.



(A) Activity Pattern $e(t)$



(B) Distribution of the next event with $t_0 = 11$ a.m.



(C) Distribution of the next event with $t_0 = 6$ p.m.

FIGURE 2.8: Example of an activity pattern for a subscriber's day, with two next event distributions $n_{11\text{a.m.}}(t)$ and $n_{6\text{p.m.}}(t)$.

A solution to this problem is to schedule the first event of each actor by sampling the residual inter-event time distribution instead of the simple inter-event time distribution. The residual inter-event time distribution is expressed as

$$\Psi(\Delta) = \frac{1}{\langle\Delta\rangle} \sum_{\tau=\Delta}^{\infty} f(\tau)$$

, where $f(\tau)$ is the inter-event time distribution and $\langle\Delta\rangle$ is the average inter-event time [LTD13a; Vaz+07]. The residual time corresponds to the time before the next event, knowing that the last event occured at a time chosen uniformly at random.

Scheduling the first interaction of each subscriber by sampling $\Psi(\Delta)$ solves the problem because we're no longer using an arbitrary starting time t_0 as the time at which the last event preceeding the beginning of the simulation occured.

For memoryless distributions, we can note that sampling from $\Psi(\Delta)$ is equivalent to directly sampling from $f(\tau)$. For example, using Poissonian activity leads to a geometric inter-event time distribution $f(\tau) = p(1-p)^{\tau-1}$, and we observe that $\Psi(\Delta)$ is equivalent to $f(\tau)$:

$$\Psi(\Delta) = \frac{1}{\langle\Delta\rangle} \sum_{\tau=\Delta}^{\infty} p(1-p)^{\tau-1} = \frac{(1-p)^{\Delta-1} - (1-p)^{\infty}}{\langle\Delta\rangle} = \frac{(1-p)^{\Delta-1}}{\langle\Delta\rangle} = p(1-p)^{\Delta-1} = f(\Delta).$$

The Distribution Actor

As illustrated on figure 2.8, the next event distribution $n_{t_0}(t)$ depends on t_0 in the general case. This forces us to compute a new distribution $n_{t_0}(t)$ everytime we're at a timestamp t_0 and want to schedule the next event to occur for an agent. If n agents are active at timestamp t_0 and must schedule their next event, we thus need to compute n new distributions per timestamp, n arrays of the values of $F(\Delta)$ if it is not possible to express $F^{-1}(\Delta)$ in a closed form. Those arrays can be big, for example, in the case of a discrete power law with a threshold of $\epsilon = 10^{-5}$, there is approximately 34000 elements.

Fortunately, many agents often share the same call behaviour, and we can compute at most one distribution for each timestamp and each kind of behaviour. To make this possible, we created a new actor that we call the "distribution actor". To avoid useless network communication, there is exactly one distribution actor on each node of the cluster. As figure 2.9 shows, an active subscriber may contact the distribution actor of its server to receive the distribution to sample. Computing and sampling distributions are CPU intensive and the distribution actor generally runs on a dedicated thread to avoid becoming a bottleneck of the simulation.

Implemented Call Distributions

We implemented a few predefined behaviours of subscribers in the form of call distributions. Burstiness is supported by using a discrete power-law distribution of the inter-event time, as discussed in the section 1.5. We support uniform activity patterns using a geometric distribution of inter-event times. We also implemented a behaviour that takes circadian cycles into account: the activity is constant through the day, and much lower during the night. A last activity pattern generalizes the circadian cycles to a one-week period, with a scaled-down activity during the

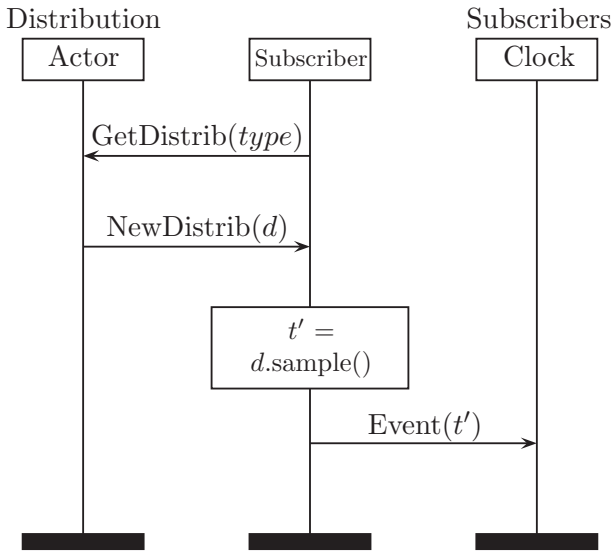


FIGURE 2.9: Protocol between the Distribution Actor and an active subscriber. If the distribution of the correct type was already computed for another actor at the current timestamp t_0, it is directly returned from a cache. The actual sampling is performed by the subscriber to help distribute the load between threads and to allow a subscriber to get as many samples with a single message exchange.

week-end. The presence of these different implementations is just a proof-of-concept, and any custom, experiment-specific, behaviour can be specified by users of the simulator.

2.5 Benchmarks

In this section, we evaluate the performances of the simulator by benchmarking key metrics. We start by looking at the maximal number of subscribers that can be handled. This number is constrained by the amount of RAM available. We then evaluate the speed of the simulation by measuring the average number of ticks simulated in a second and the average throughput in terms of CDRs. We finally look at the efficiency of resources usage, such as the average load of the CPU.

The performances are analyzed in a local setup, with the simulator running on a single computer. The analysis of performances on a single machine is interesting because it provides a way of computing an upper bound on the performances of a cluster. Because a distributed system is slowed down by the latency of network communication, we expect a cluster of n nodes to always have performances inferior to n times the performances of a single node. The single node experiments reported in this section were conducted using a mid-2015 MacBook Pro sporting a 2.2 Ghz i7 processor (4 cores), and 16 gigabytes of RAM. The results were averaged over 5 runs.

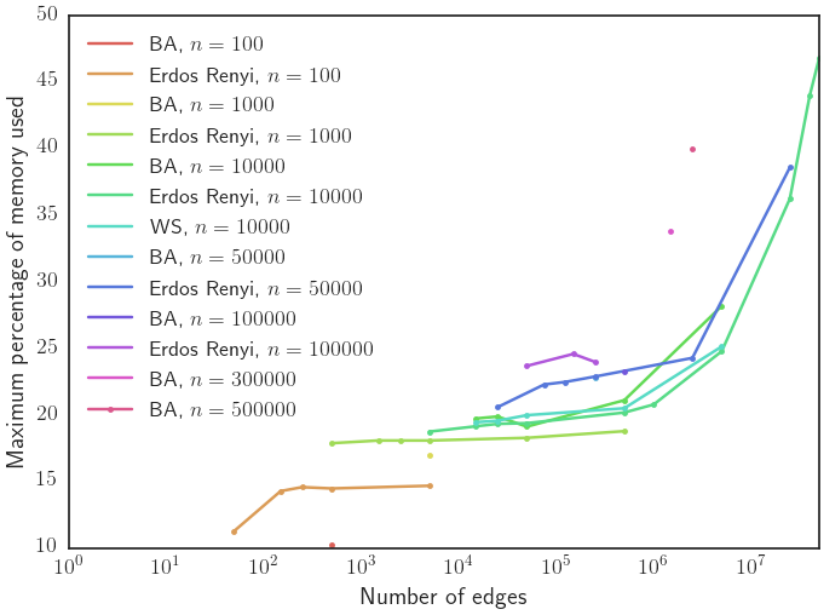


FIGURE 2.10: Evolution of the maximum percentage of memory used as a function of the number of edges, for networks of different kinds and sizes. Increasing the number of subscribers or using different network models has little impact on memory pressure, which is mostly governed by the number of edges.

2.5.1 Size of the Network & Memory Pressure

As the networks we simulate get larger in terms of number of subscribers, the number of relationships also naturally increases. Indeed, if the average degree $\langle k \rangle$ is preserved, growing the network by Δn individuals will on average result in the creation of $\Delta n \langle k \rangle$ new edges. The number of towers and operators may also grow, but should stay negligible compared to the quantity of subscribers for most applications.

This large quantity of edges means that friendships can quickly become the bottleneck in terms of memory consumption. This is confirmed by results reported on figure 2.10. The state of a subscriber is roughly contained in 104 bytes, while each friendship consumes 24 bytes of storage. A network of 100000 subscribers, each of which maintains on average $\langle k \rangle = 100$ social relationships, will result in the consumption of 10 and 240 megabytes of memory for subscribers and relationships, respectively. These sizes are estimated using the *jmap* tool which takes a snapshot of the memory usage of the *JVM*. They are only a rough estimate because *jmap* doesn't allow to compute the size of the objects pointed to by references located inside the subscribers and edges classes.

2.5.2 Performances of God

The performances of *God* are largely governed by the number of edges. This is illustrated on figures 2.11 and 2.12.

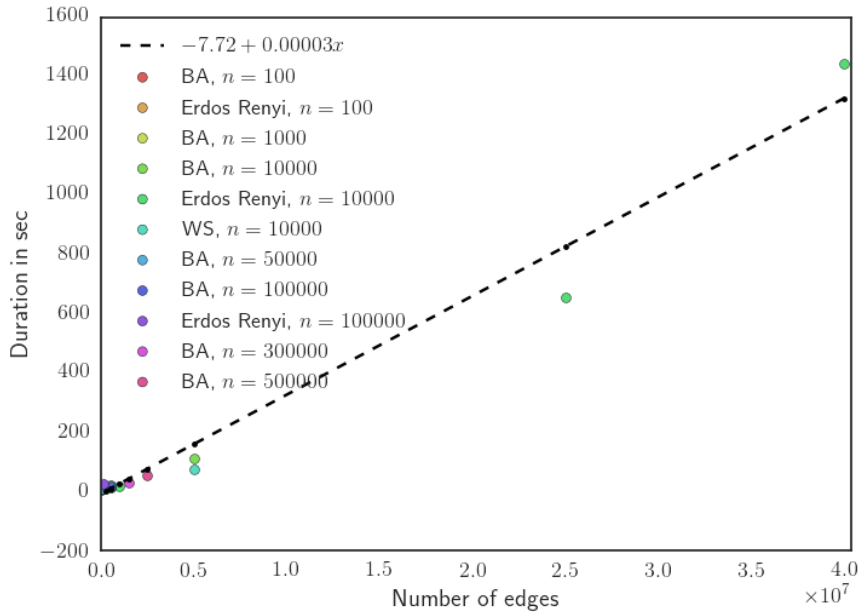


FIGURE 2.11: Evolution of *God*'s runtime as a function of the number of edges to generate, for different types and sizes of graph. The number of subscribers doesn't have nearly as high an impact as the number of edges.

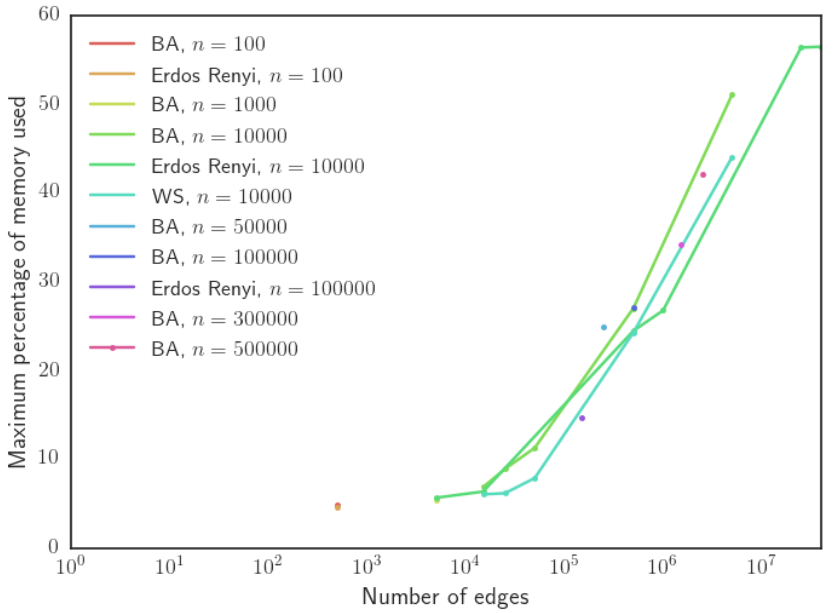


FIGURE 2.12: Evolution of the maximum percentage of memory used as a function of the number of edges for different types of graph and for different sizes.

2.5.3 Performances of Nature

To evaluate the performances of *Nature*, we define three key metrics:

- n_{active}, the average number of active actors for a tick

- n_{ticks}, the average number of ticks simulated in one second
- n_{cdrs}, the average number of CDRs that are output in one second (throughput)

If only few actors are active during a tick (n_{active} is low), we expect *Nature* to spend most of its computation time performing synchronization tasks. Although n_{ticks} should be large, the throughput will stay low. It is clear that if there aren't many active agents per tick, *Nature* will spend most of its time doing the synchronization. Although the number of ticks per second is going to be huge, the throughput n_{cdrs} is going to be small. On the other hand, if n_{active} increases, *Nature* will spend a larger fraction of time handling all the calls. The speed at which ticks are processed will logically decrease, but the throughput will increase. These intuitions are confirmed empirically, as can be seen on figures 2.13 and 2.14. Figure 2.15 confirms that the time it takes to simulate 2000 ticks grows linearly with the number of CDRs produced.

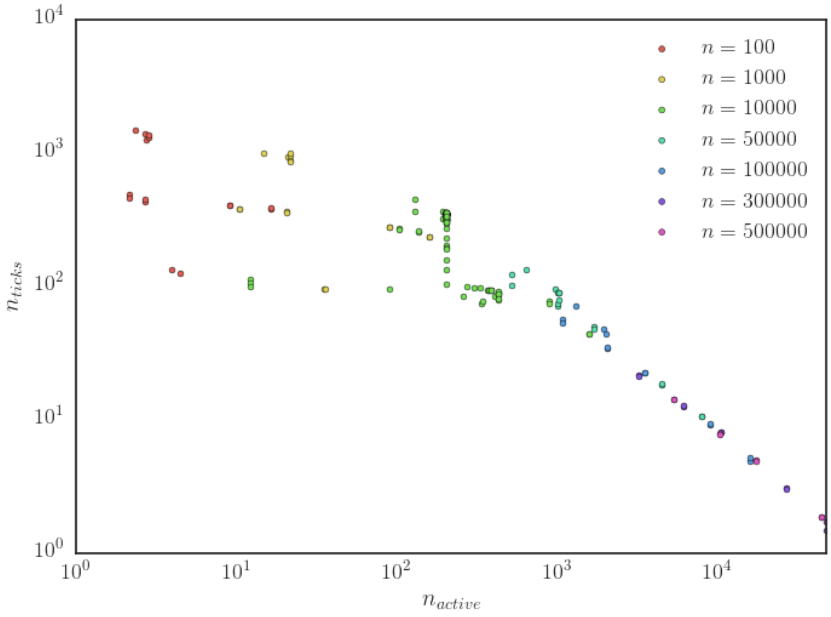


FIGURE 2.13: Evolution of the number of ticks handled per second as a function of the number of active agents for different types and sizes of graphs. As the number of active actors n_{active} increases, the simulation slows down.

The number of active agents depends on the size of the network and of the activity distribution. The greater n_{active}, the more CPU intensive *Nature* gets. This is illustrated on figure 2.16 which shows that *Nature* spends proportionally less time performing synchronization. The CPU load seems unaffected by the topology of the networks, but depends only on the number of active actors.

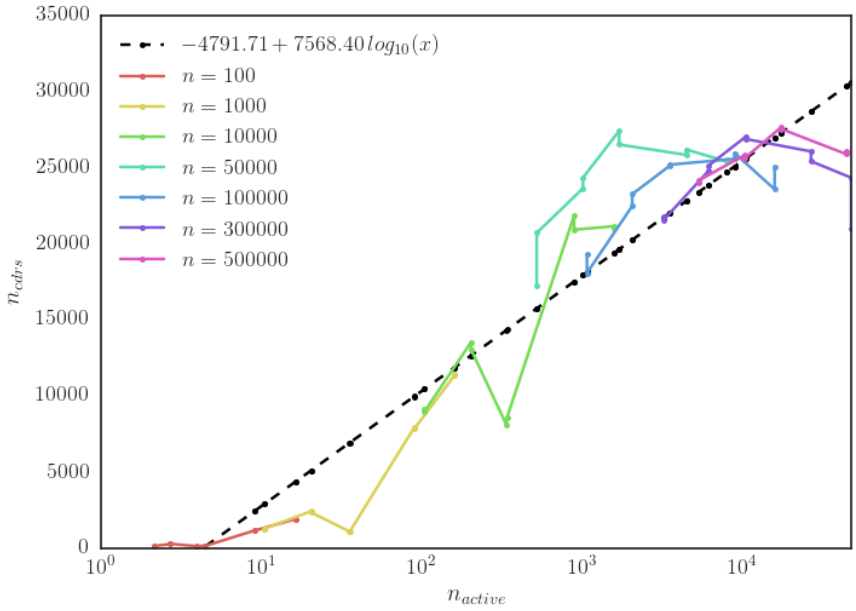


FIGURE 2.14: Evolution of the average number of CDRs generated per second as a function of the number of active agents for different sizes of graph BA graph. As the number of active actors increases, so does the throughput.

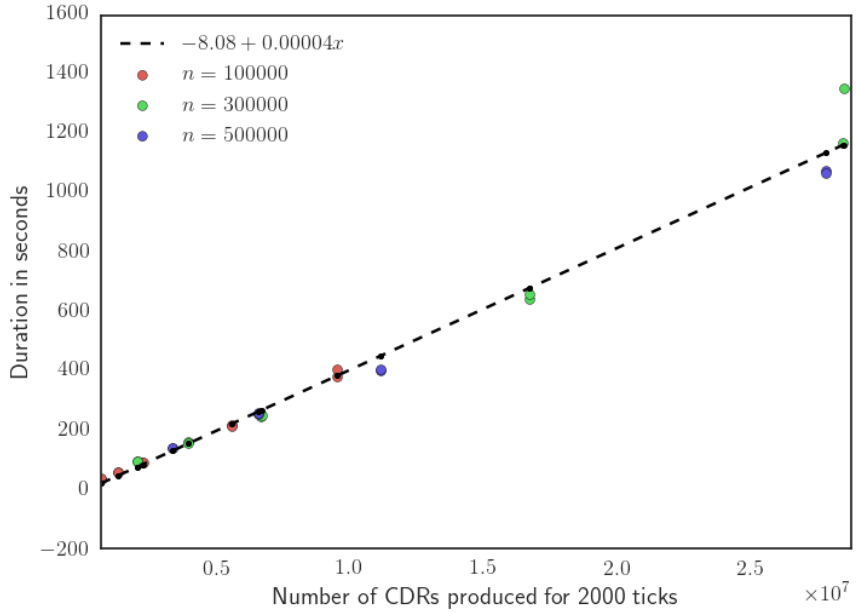


FIGURE 2.15: Evolution of the duration of the simulation of 2000 ticks as a function of the number of CDRs produced, for different types and sizes of graph. As the number of active actors n_{active} increases, the simulation speed slows down and the throughput increases.

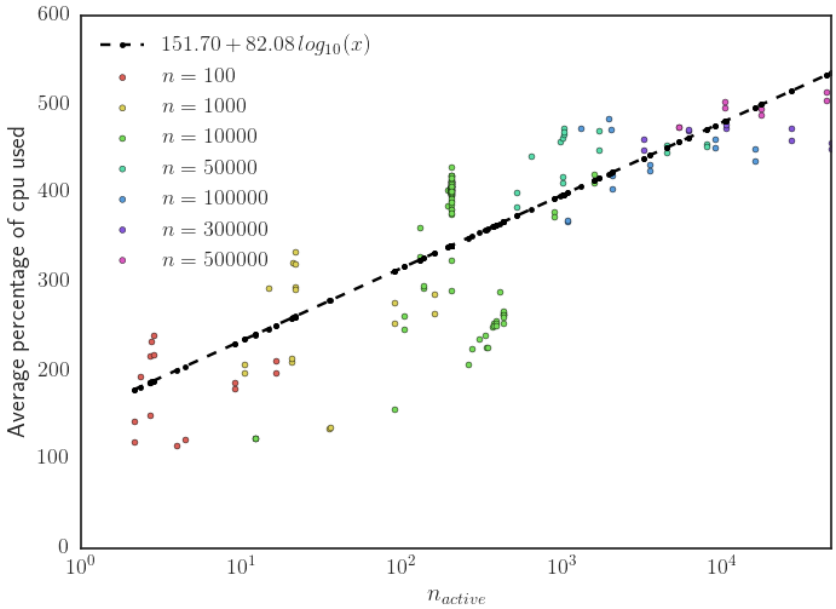


FIGURE 2.16: Evolution of the percentage of the CPU used as a function of the number of active agents for graphs of different kinds and sizes. As larger numbers of actors are active on average, *Nature* spends proportionally less time handling time synchronization which translates to a higher CPU load.

2.6 Conclusion & Future Work

The design choices explained in this chapter mainly aimed at the creation of an extensible simulator, capable of handling different assumptions about both the network topology and the activity patterns of individuals. The extensibility is provided by the high-level abstraction of actors which eases the modeling of human behaviours. Our benchmarks highlight the fact that memory consumption is linked to the number of edges of the network, while the raw performances largely depend on the number of active individuals.

Another goal of the design was to leverage the parallelism and increased resources made available on a distributed system. Although our benchmarks show that multiple cores are exploited when running the simulator on a single processor, this does not prove that using several machines would multiply the performances. To verify this, strict benchmarks should be conducted in a distributed setting, involving many different machines. Such benchmarks would involve monitoring memory and CPU usages on each machine, to verify that the computational load is correctly balanced. Another metric of interest is the network usage. Since network communication is an order of magnitude slower than memory reads, the exchange of many messages could become the bottleneck of the simulation.

In a distributed setting, suscribers are currently placed randomly on nodes of the cluster. One possible improvement would be to take the network topology into account when mapping subscribers to physical machines. As a first approximation, we could imagine using a community detection algorithm to find communities of similar sizes, before mapping the subscribers of each community to the same cluster node. This optimization would likely help reduce network load, because most exchanged messages would never leave the community of the sender, and would therefore stay on a single machine. Although this idea seems reasonable, it may not be efficient for social graphs that have small clustering coefficients, such as ER or BA networks.

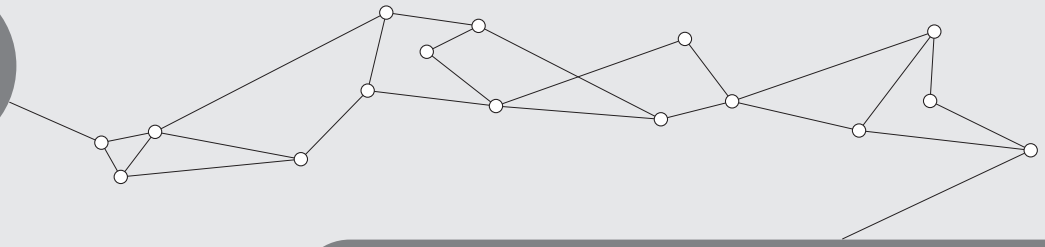
The performances could also likely be improved by optimizing the memory footprint of the representation of suscribers. Any change should be carefully considered, because removing information from the subscribers could deteriorate the extensibility and genericity of the simulator. In a similar fashion, the simulator could be specialized by removing the tower and operator actors, which would reduce the number of message exchanges needed to place a call. However, these actors are necessary in the production of meaningful CDRs that could be used by *Real Impact Analytics*.

Besides performances optimization, another area where work remains to be done is fault-tolerance. To ease the process of implementating a truly fault-tolerant simulator, we made sure to follow Akka's guidelines and to define a clear actor hierarchy.

Finally, it would be interesting to leverage the flexibility of the simulator to implement mobility patterns. Among many other applications, mobility aspects could be important for epidemic spreading. Spatial distances between individuals should be taken into account to model a realistic geographical propagation of an infection.

None of these suggestions should be implemented without first benchmarking the simulator in a truly large-scale setting, to objectively measure where improvements should be made. The main focus of this thesis was the development of a working, flexible, simulator. Our implementation was succesfully tested on a small-scale distributed environement of 4 machines, kindly provided by RIA. The current design turned out to be efficient enough for our purposes, since it allowed us to conduct experiments on reasonably large networks. The results of these epidemic experiments are presented in chapter 3.

3



Application to Epidemic Spreading Simulation

Many theoretical concepts of epidemic spreading analysis were introduced in chapter 1. In chapter 2, we discussed the design of an event-driven simulator able to generate a temporal network. The simulator allows users to specify different network models, and different behaviours for individuals through the customization of social activity patterns. In this chapter, we will put the flexibility of this tool to good use by using it to conduct experiments about epidemic spreading. Various network models and types of activity will be considered.

3.1 Simulation Procedure

To avoid targeting our simulator to the specific task of epidemic spreading simulation and keep it generic, we have decided to output CDRs and run an external epidemic simulation that uses these CDRs. This means that a complete spreading experiment includes tree distinct steps:

1. Run *God* to generate a social network.
2. Run *Nature* to simulate temporal interactions between individuals of the generated graph.
3. Run the epidemic simulation which interprets each CDR as a temporal contact. The simulation starts from a given set of randomly selected infected individuals. Interactions of these individuals with susceptible ones will propagate the disease. Our script was written in Python and supports the 3 standard epidemic models: SI, SIS and SIR.

Besides preserving the flexibility of *Nature*, running epidemic simulations externally also has the advantage that it is faster. Indeed, we try different epidemic parameters on the same set of CDRs without having to multiply runs of *God* and *Nature*. One drawback of our approach is that it prevents us from simulating epidemics-related behaviours inside *Nature*. For example, if the epidemic was run directly inside *Nature*, we could alter the behaviours of sick people so that they tend to communicate less with their neighbours. We can also imagine implementing quarantine procedures directly inside the simulator. Using an external Python script, we can attempt to reproduce this by discarding a fraction of the CDRs involving infected individuals, but this wouldn't take correlations between calls into account.

3.1.1 Possible Parameters

Each of the 3 steps we mentioned comes with its own set of parameters. As we have discussed in chapter 2, *God* can handle the 3 standard network models that are Erdős-Rényi (ER), Watts-Strogatz (WS) and Barabási-Albert (BA). In addition to specifying the number of nodes n, the

3 models have specific parameters that were already detailed in section 1.3. We briefly remind them here:

ER: p, the probability that any given edge exists.

WS: $\langle k \rangle$, the degree of nodes in the completely regular lattice and p, the probability of rewiring any given edge.

BA: These networks are built by growing an initial graph through the addition of new nodes. Each new node gets attached to m existing nodes, taking preferential attachment into account.

The main parameter of *Nature* in our experiments is the activity distribution used by each individual. We consider here that all individuals share the same distribution. As explained in section 2.4.6, *Nature* supports arbitrary activity distributions, but our experiments will consider only uniform and power law distributions of the inter-event time, *i.e.* Poissonian and Bursty activities. The value of the average number of contacts of an individual at each timestamp is given by $\langle k \rangle \langle c \rangle$ and is implicitly defined by the chosen network model and activity distribution.

Finally, the epidemic spreading simulation that we run using CDRs generated by *Nature* shares some parameters with classical epidemic models:

- β, the probability that a sick individual infects a sane one upon contact.
- μ, the probability that an infected individual recovers during a given timestamp. The duration of the infectious period of an individual follows a geometric distribution of parameter μ. This parameter is only applicable to SIS and SIR simulations.
- i_0, the proportion of the population that is initially infected.

Because each one of *God*, *Nature* and the epidemic simulator carries some randomness, all our experiments must be repeated. We decided to repeat the *Nature* and *God* steps 3 times, and to perform 5 runs of the epidemic simulation. This means that for each set of parameters, we create 3 different social networks using *God*, then generate 3 sets of CDRs for each one of those social networks (9 sets of CDRs in total), before launching 5 runs of the epidemic simulation on each set (45 runs in total).

Since we use CDRs as a proxy for generic social contacts, the duration of calls may influence our results. To prevent this, we force *Nature* to generate calls that last for exactly one timestamp.

3.1.2 Normalization

In order to be able to meaningfully compare spreading results over different sets of parameters, these results need to be normalized. One possibility is to use the same transmibility factor R_0 for all the models that we compare. As discussed in section 1.2.1, $R_0 = \beta \langle c \rangle \langle k \rangle$ for the *SI* model, and $R_0 = \frac{\beta \langle c \rangle \langle k \rangle}{\mu}$ for the *SIS* and *SIR* models. As a reminder, R_0 can be interpreted as the expected number of infections caused by a single infected individual during his infectious period, assuming he is placed inside a fully susceptible population.

We will generally use the classical compartmental models as the baseline for our comparisons. As we have seen in section 1.2.1, these models are easily solved analytically or numerically. We distinguish empirical parameters used in simulations from the analytical parameters of the

classical models by subscripting them with a "e". Since the value of $\langle k_e \rangle$ varies from model to model, the normalization is applied by choosing the empirical infection probability of β_e as follows:

$$\begin{aligned} R_0 &= R_0^e \\ \frac{\beta \langle c \rangle \langle k \rangle}{\mu} &= \frac{\beta_e \langle c_e \rangle \langle k_e \rangle}{\mu_e} \\ \beta_e &= \frac{\beta \langle c \rangle \langle k \rangle}{\langle c_e \rangle \langle k_e \rangle}, \end{aligned}$$

where we chose to set $\mu = \mu_e$. Using this value for β_e in all empirical models enables us to compare them to the same baseline of classical models. Figure 3.1 shows an experiment comparing *SI* spreading on ER networks using different rates of Poissonian activity. As we would expect, the normalization causes all curves to become superimposed.

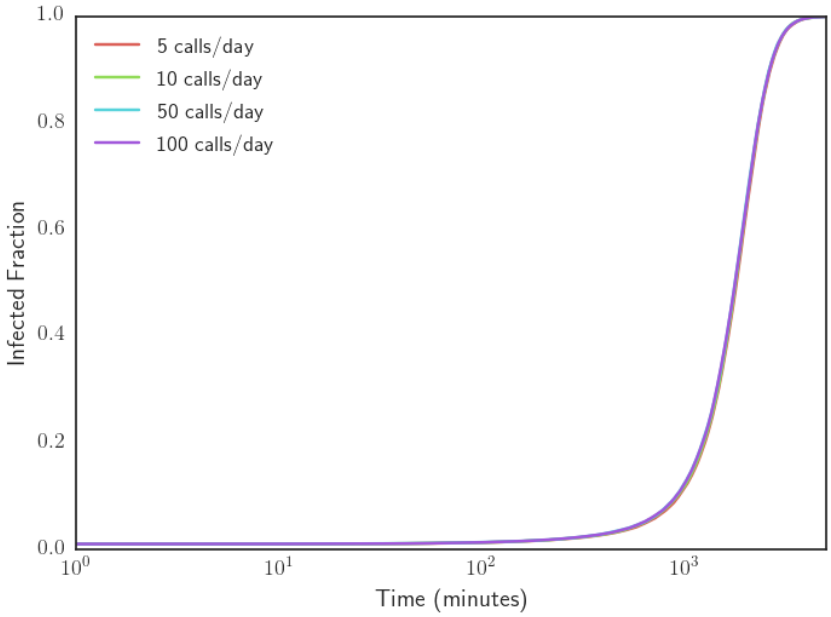


FIGURE 3.1: SI propagation over an ER network of size $n = 50000$. The probability of existence of an edge is $p = 0.0004$, which yields an average degree $\langle k_e \rangle = pn = 20$. The activity is Poissonian, and the number of calls per day varies between 5 and 100. All models are normalized such that $R_0^e = R_0$.

3.2 ER *vs.* Classical Compartmental Models

In this experiment, we will use our simulator to analyze the impact of using an ER network instead of assuming perfect mixing between all compartments like it is done in classical models. Perfect mixing is reproduced by using a complete network with uniform weights. This network can be generated using the ER model with $p = 1$, and equal w_{ij} for all edges. Under the Poissonian activity assumption, epidemic spreading simulations on such a network are equivalent to the classical compartmental models.

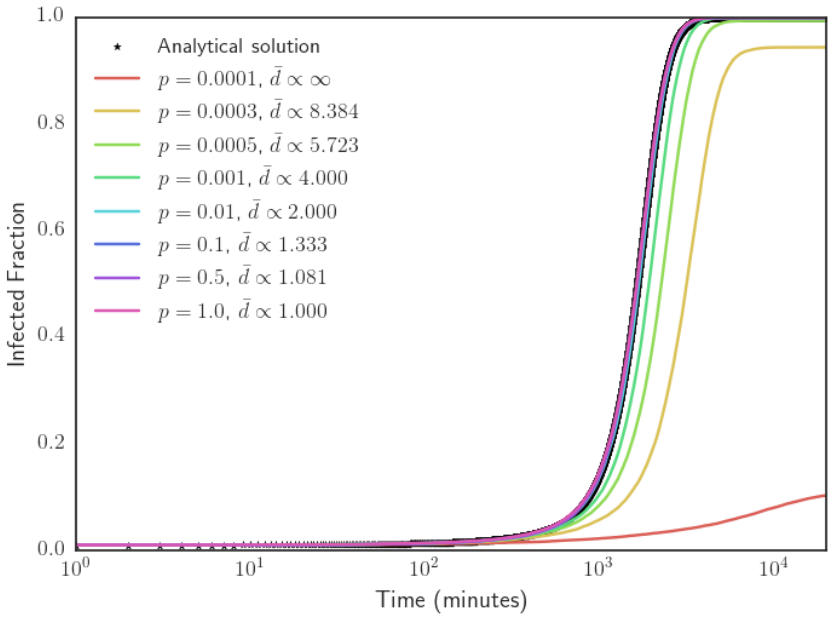


FIGURE 3.2: SI propagation over an ER network of size $n = 10000$ with a Poissonian activity of 10 calls per day. The value of β is fixed to 0.2, and the average degree varies with p. The order of magnitude of the diameter is estimated using $d \propto \frac{ln(n)}{ln(np)}$.

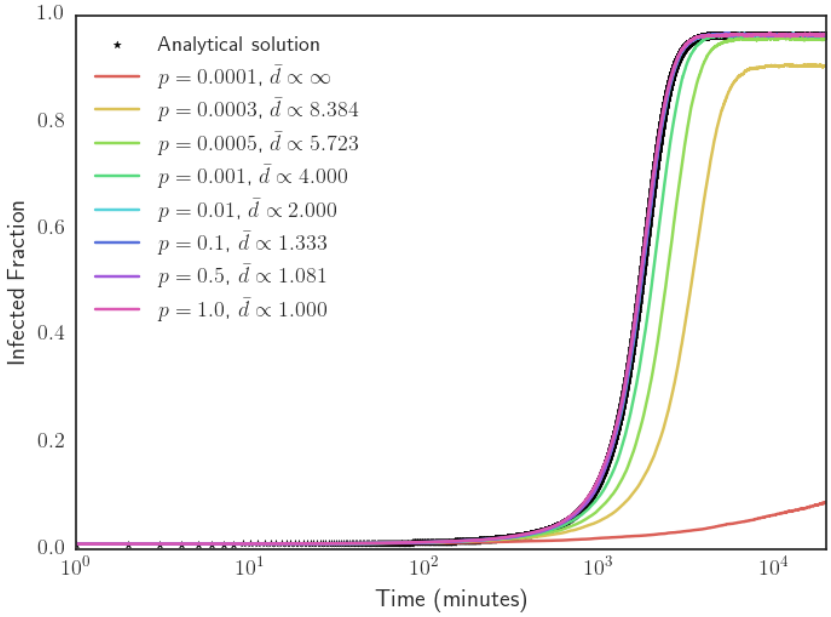


FIGURE 3.3: SIS propagation over an ER network of size $n = 10000$ with a Poissonian activity of 10 calls per day. The value of β is fixed to 0.2, and the average degree varies with p. The order of magnitude of the diameter is estimated using $d \propto \frac{ln(n)}{ln(np)}$.

As p gets smaller, the expected diameter grows, and some parts of the network become isolated. Intuitively, this makes it harder for a disease to spread on the graph. If p gets too small, it becomes likely that the network will not be connected anymore, resulting in an infinite

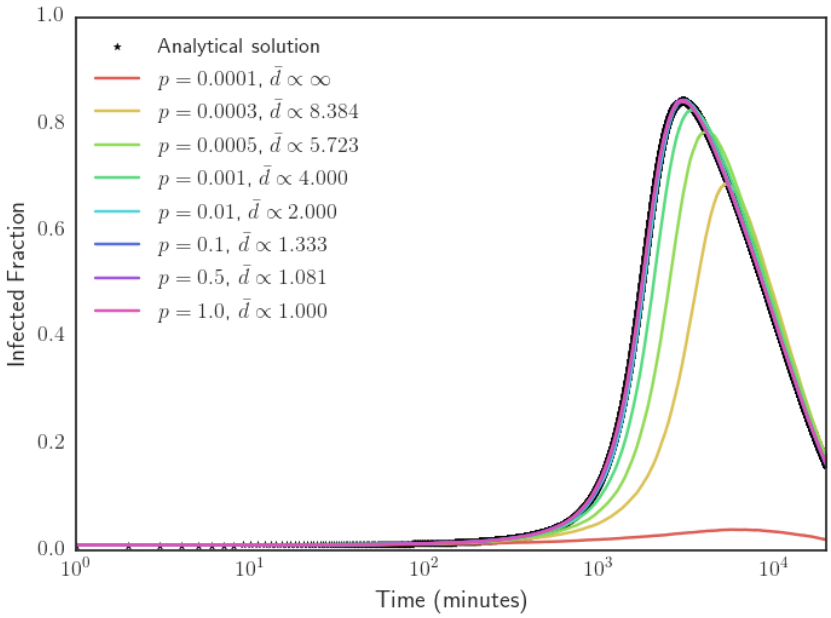


FIGURE 3.4: SIR propagation over an ER network of size $n = 10000$ with a Poissonian activity of 10 calls per day. The value of β is fixed to 0.2, and the average degree varies with p. The order of magnitude of the diameter is estimated using $\bar{d} \sim \frac{ln(n)}{ln(np)}$.

diameter. In this case, some subgraphs are isolated, and the infection can only reach the whole population if there is at least one initial infected in each subgraph.

In classical models, the average number of contacts at each timestamp is computed as $\langle c \rangle \langle k \rangle$. Using the results of our simulations, we can only estimate the empirical values by

$$\langle c_e \rangle \langle k_e \rangle = \frac{2}{\langle \Delta_e \rangle},$$

where $\langle \Delta_e \rangle$ denotes the average empirical inter-event time. The factor of 2 is needed because each event involves two individuals. In order to be able to compare the different models, we normalize all results using the procedure described in section 3.1.2. In our simulation, an individual can only make one call per timestamp, so this is just an approximation. If we consider Poissonian activity, the average number of contacts at a given timestamp is given by $2p_{call}$.

Figures 3.2, 3.3 and 3.4 respectively present spreading results for *SI*, *SIS* and *SIR* infections. As the diameter of the ER network grows, the spreading slows down as expected. Furthermore, infinite diameters mean that the infection may never reach some remote parts of the network, which causes the fraction of infected to always stay lower than 1. For all 3 infection types, the $p = 1$ case (complete network) matches the result provided by the solution to the corresponding classical model.

3.3 SIS Epidemic Thresholds on Scale-Free Networks

According to theorical SIS models, using a Poissonian activity assumption on scale-free networks should lead to vanishing epidemic thresholds for R_0, as the network size tends to infinity. Intuitively, the presence of hubs in the network makes the epidemic very hard to eradicate. Indeed, a single infected hub may propagate the disease to many of his neighbours.

It is important to be able to determine whether a SF network is large enough for the epidemic threshold to be radically smaller than on an ER network of the same size. For example, do we observe a significant difference between the two network types when modeling a city of 500000 inhabitants?

As mentioned in section 1.3.3, the mean field approach allows to express the SIS epidemic threshold under Poissonian activity as

$$\lambda_c = \frac{\langle k \rangle}{\langle k^2 \rangle}.$$

The value of λ_c actually defines a threshold on the value of the spreading rate $\lambda = \frac{\beta}{\mu}$. We can of course express this as a threshold on the value of the transmissibility R_0:

$$R_0 = \frac{\beta \langle k \rangle}{\mu} = \lambda \langle k \rangle \quad \longrightarrow \quad R_0^c = \lambda_c \langle k \rangle = \frac{\langle k \rangle^2}{\langle k^2 \rangle}$$

For ER networks that are sufficiently sparse, the threshold λ_c is simply derived as

$$\lambda_c^{ER} = \frac{\langle k \rangle}{\langle k^2 \rangle} = \frac{\langle k \rangle}{\langle k \rangle (\langle k \rangle + 1)} = \frac{1}{\langle k \rangle + 1}.$$

This yields $R_0^c = \lambda_c \langle k \rangle = \frac{\langle k \rangle}{\langle k \rangle + 1} \approx 1$, for large values of $\langle k \rangle$.

The computation is slightly more complicated for BA networks whose degree distribution is a power law Ck^{-3}, where C is a normalization constant. The second moment of the distribution is given by

$$\langle k^2 \rangle = \sum_{k=k_{min}}^{n-1} k^2 \, C \, k^{-3}.$$

We then get the threshold

$$\lambda_c = \frac{\langle k \rangle}{\langle k^2 \rangle} = \frac{\langle k \rangle}{\sum_{k=k_{min}}^{n-1} C \, k^{-1}}$$

Figure 3.5 compares the theoretical thresholds λ_c obtained for ER and BA networks of varying sizes and different average degrees $\langle k \rangle$. While the difference between the two kinds of network is noticeable, the conclusion that λ_c vanishes for large networks seems to only apply to very large networks. Indeed, even for networks of 1 million nodes, the difference never exceeds an order of magnitude. It is interresting to notice that multiplying the ER thresholds λ_c by $\langle k \rangle$ always yields $R_0 = \lambda_c \langle k \rangle \approx 1$, as expected it is close to the threshold of the classical compartmentsl models $R_0 = 1$.

On figure 3.6, we confront theoretical predictions of the value of the threshold with empirical

results from the simulation. An SIS epidemic is run with constant parameters and Poissonian activity, and the infected fraction of the population is observed for different network sizes. For small network sizes, the disease is not resilient enough to be able to spread. As the network grows, the theoretical threshold is lowered and the disease manages to propagate. These empirical results are consistent with the theorical expectations presented on figure 3.5.

Figure 3.7 and 3.8 allow us to compare the resilience of a SIS epidemic over ER and BA networks, respectively. The two networks are of the same size, and the same epidemic parameters are used in both cases. Since the activity is Poissonian and the average degree $\langle k \rangle = 200$ is already quite high compared to the size of the network ($n = 10000$), we expect the ER network to have a behaviour close to the predictions of classical compartmental models. This is indeed the case, as the curves are very close. The average diameter of these ER networks was measured to be of 1.7. This explains why the ER spreading is slightly slower than the theoretical classical spreading model, which can be seen as having a diameter of 1. Figure 3.8 confirms that the power law degree distribution of BA networks tends to help the spreading. The spreading is both faster than what is predicted by the classical model and observed on the ER networks, but also much more resilient.

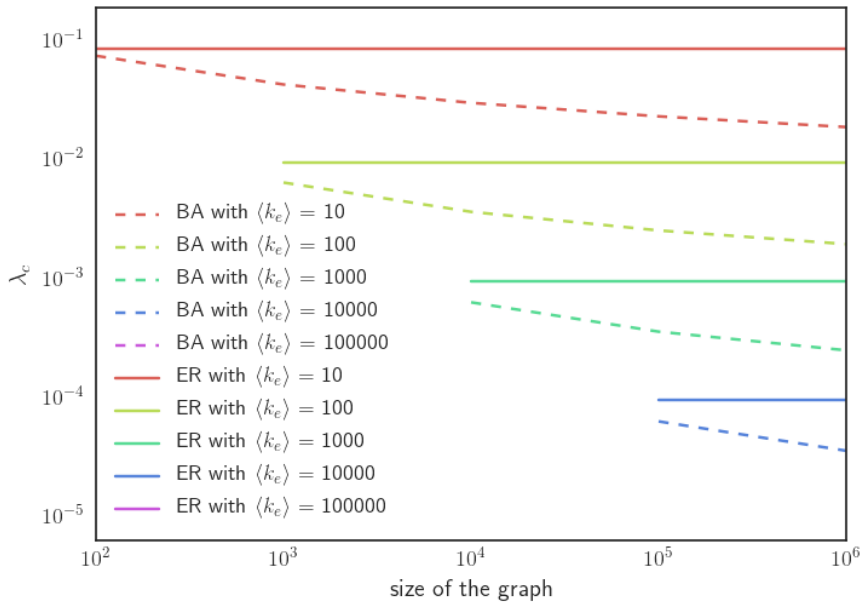


FIGURE 3.5: Comparison of the theoretical threshold λ_c for BA and ER networks as a function of the number of nodes and of the average degree $\langle k \rangle$. For networks containing less than a million nodes, the threshold obtained for BA networks is always less than an order of magnitude smaller than the threshold obtained for ER networks.

3.4 Time Correlation

In section 2.4.5, we mentioned the possibility for an individual to reschedule his next event upon receiving a call. In *Nature*, this occurs with a constant probability r defined as a parameter of the simulator. This kind of rescheduling can introduce a correlation between different interactions. Indeed, if the time of the next event is recomputed by an individual each time he receives a call, it may in some cases tend to reduce the presence of long inactivity periods.

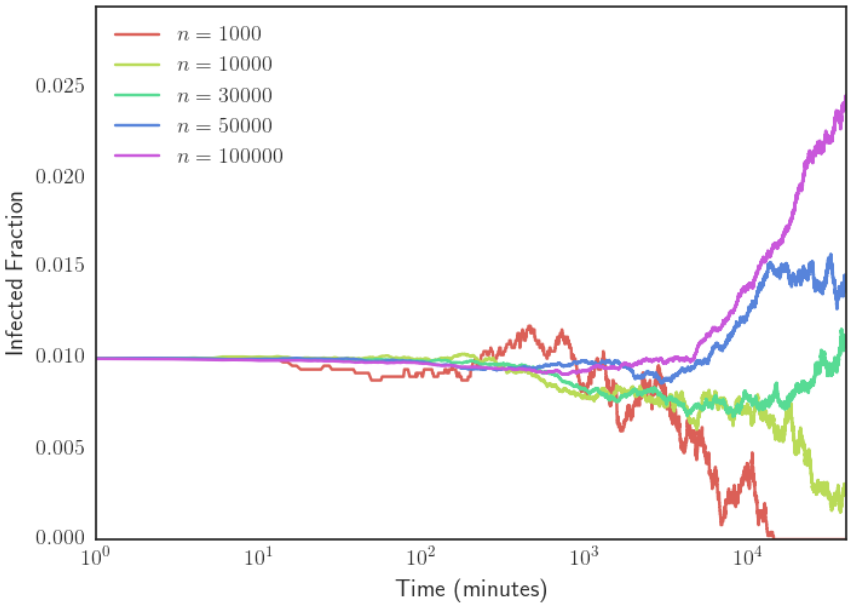


FIGURE 3.6: Comparison of SIS spreading over BA networks with constant average degree $\langle k \rangle = 200$ and epidemic parameters $\beta = 0.125$, $i_0 = 0.01$ and $\mu = 0.002$, but varying sizes. The activity is Poissonian with 10 calls per day. As the network grows, the spreading gets more resilient and the epidemic threshold is lowered, as predicted by theoretical results obtained through mean field theory.

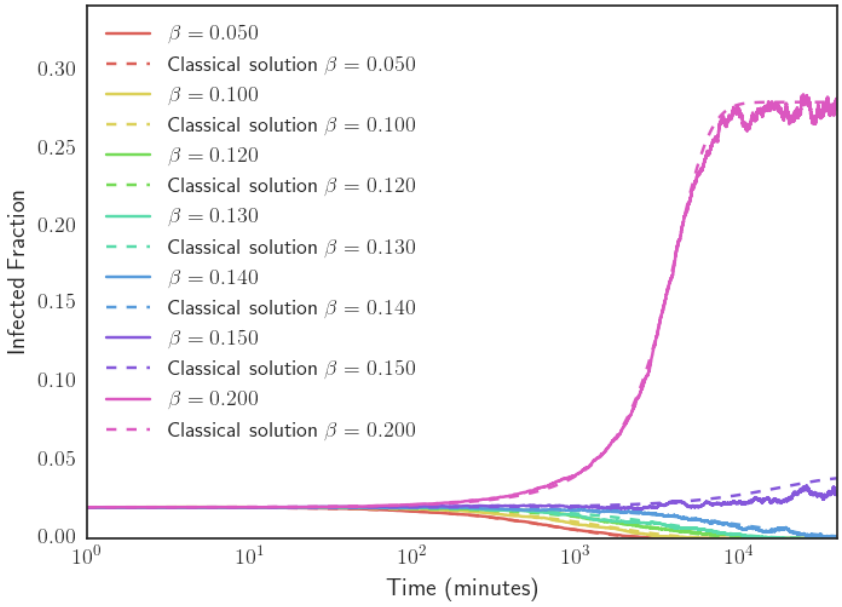


FIGURE 3.7: SIS propagation over an ER network of size $n = 10000$, with $\langle k \rangle = 200$, $p = 0.01$, $\mu = 0.002$, $i_0 = 0.002$ and varying values of β. The activity is Poissonian with a rate of 10 calls per day. The plots clearly show that is a threshold between $\beta = 0.15$ and $\beta = 0.14$ which corresponds to the theoretical threshold, $\beta_c = \frac{\lambda_c * \mu}{\langle c \rangle} = \frac{\mu \langle \Delta \rangle \langle k \rangle}{<k>+1} = 0.0144$. This critical value for β is close to the theoretical value provided by classical models ($\beta_c = 0.0145$).

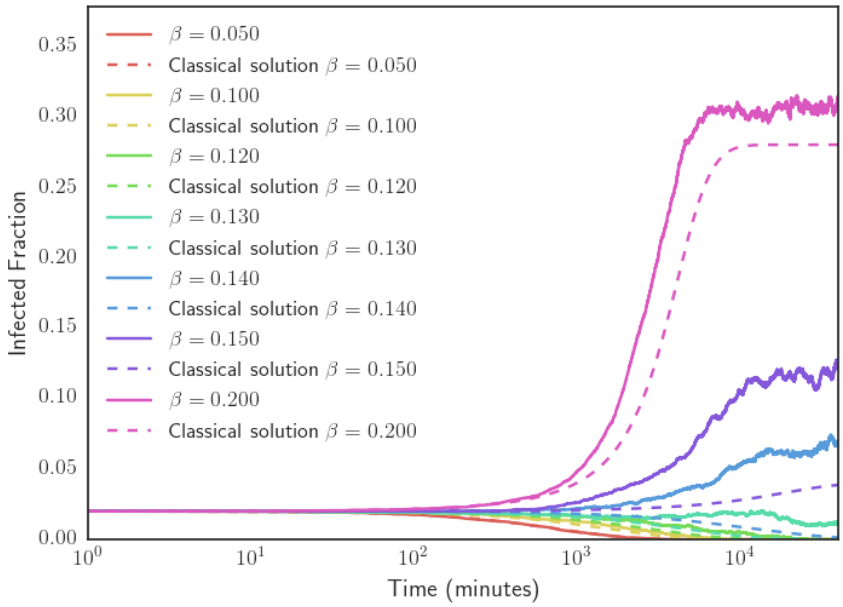


FIGURE 3.8: SIS propagation over an BA network of size $n = 10000$, with $\langle k \rangle = 200$, $p = 0.01$, $\mu = 0.002$, $i_0 = 0.002$ and varying values of β. The activity is Poissonian with a rate of 10 calls per day. We can clearly see that there is a threshold between $\beta = 0.13$ and $\beta = 0.12$ which correspond to the theoretical threshold, $\beta = \frac{\lambda_c * \mu}{\langle c \rangle} = \frac{\mu \langle \Delta \rangle \langle k \rangle^2}{< k^2 >} = 0.0125$. The critical β is smaller than on a comparable ER network since at $\beta = 0.013$, the spreading dies on the ER network but survives on this BA network.

3.4.1 Impact of Correlations using Poissonian Activity

This kind of correlations doesn't have an impact on spreading when the activity is Poissonian. Indeed, the rate of activity p stays constant and the probability of placing a call at a given time t depends only on the value of t. The inter-event time follows a geometric distribution and we can schedule the next event by drawing a sample Δt from it and adding it to the current time t_0. Equivalently, we can perform a binomial trial at every timestamp from t_0 until we get a success. Let's assume we scheduled a waiting period at time t_0 and it is supposed to last until time t_1. If a neighbour interrupts the waiting period and the next event is rescheduled, it will have the exact same probability of being rescheduled at time t_1 as when we first scheduled it during timestamp t_0.

Whenever the process is memoryless, *i.e.* whenever we can express the probability of placing a call at a given time t as a function of t only, reschedulings are expected to have no effect. To verify this, we compared experimental inter-event time distributions observed when setting $r = 0$ (no correlation) and $r = 1$ (systematic rescheduling). The results are illustrated on figure 3.9. We only discussed Poissonian activity, but this result holds for any activity pattern that we defined in section 2.4.6, since these activity patterns are defined as probabilities of placing a call at a given timestamp, irrespective of previous calls that were placed.

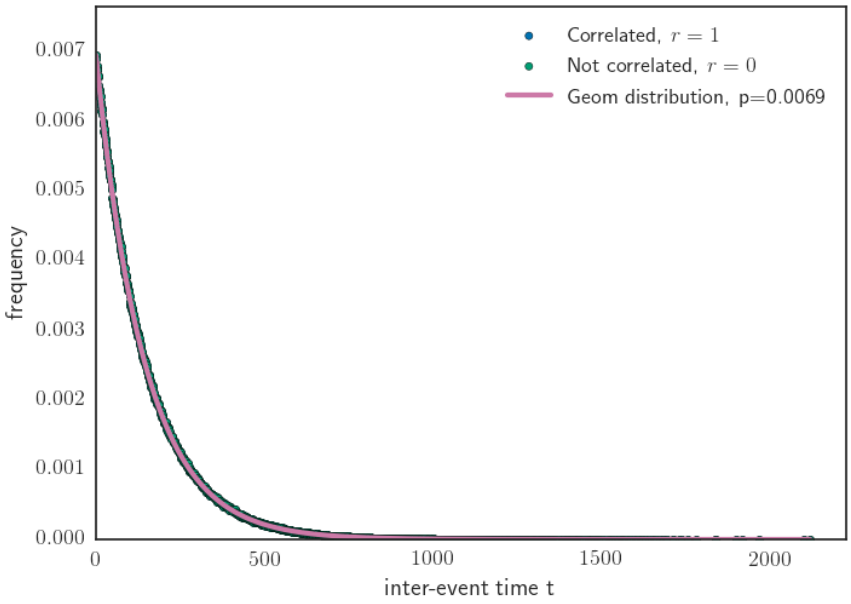


FIGURE 3.9: Comparison of a Poissonian activity with time correlation and Bursty actitivity with time correlation Distribution of inter-event times obtained on a ER network ($\langle k \rangle = 200$) with Poissonian activity with time correlation. The presence of correlations provoked by the rescheduling of the next event upon the reception of a call has no noticeable impact on the distribution. The simulation was run for 40000 timestamps, with a constant activity corresponding to 10 calls per day ($p = \frac{10}{24 \cdot 10} = 0.00694$). The two curves converge to the theorical geometrical distribution that uses the same value of p.

3.4.2 Impact of Correlations using Burstiness

The effect of correlations between calls is noticeable when dealing with processes that have some memory of past calls. To show this, we will now experiment with burstiness. A bursty behaviour is characterized by an inter-event time that follows a power law distribution. The long tail of a power law means that a lot of events are only separated by short periods of time, while longer pauses are also possible. The longer a pause, the more likely it is to be interrupted by a call from a neighbour, if correlations are enabled ($r > 0$).

Using burstiness, the activity cannot be expressed as a function of the current timestamp, because it also depends on previous calls. Again, let's assume an individual is in the middle of a pause: after ending a call at t_0, he scheduled his next event at time $t_1 > t_0$ by sampling t_1 from its next event distribution. This next event distribution depends on the time t_0 of the last interaction. If a rescheduling occurs in the middle of the pause, the probability of scheduling the next event at t_1 will be different than it was when we first scheduled it during timestamp t_0. This is was not the case when dealing with memoryless processes. Events that were scheduled at time t_1 are likely to be rescheduled at an earlier time $t_1' < t_1$, which corresponds to cutting the tail of the inter-event time distribution. This result holds for both ER and BA networks, as illustrated on figure 3.10.

It is interesting to note that the empirical inter-event time distribution using correlations

shows graphical similarities with a double pareto distribution. This distribution regularly appears in the litterature and can be generated by stopping a lognormal process after an exponentially distributed random time [Mit04; Dec15].

To analyze correlations more formally, let us introduce 3 random variables:

- $T_{call} \sim c(t)$: The time before making the next call.
- $T^S_{call} \sim s(t)$: The time before making the next call, sampled from the inter-event time distribution.
- $T_{recv} \sim r(t)$: The time before receiving the next call.

We can express $P(T_{call} = \tau)$ as follows:

$$\begin{aligned} P(T_{call} = \tau) &= P(T^S_{call} = \tau,\, T_{recv} > \tau) + \sum_{t=1}^{\tau-1} P(T^S_{call} > t,\, T_{recv} = t,\, T_{call} = \tau - t) \\ &= P(T^S_{call} = \tau)\, P(T_{recv} > \tau) + \sum_{t=1}^{\tau-1} P(T^S_{call} > t)\, P(T_{recv} = t)\, P(T_{call} = \tau - t) \end{aligned}$$

This allows us to express the inter-event time distribution $c(\tau)$ based on the distribution of the sampled inter-event time $s(\tau)$ and on $r(t)$, the distribution of the time remaining before the next call is received:

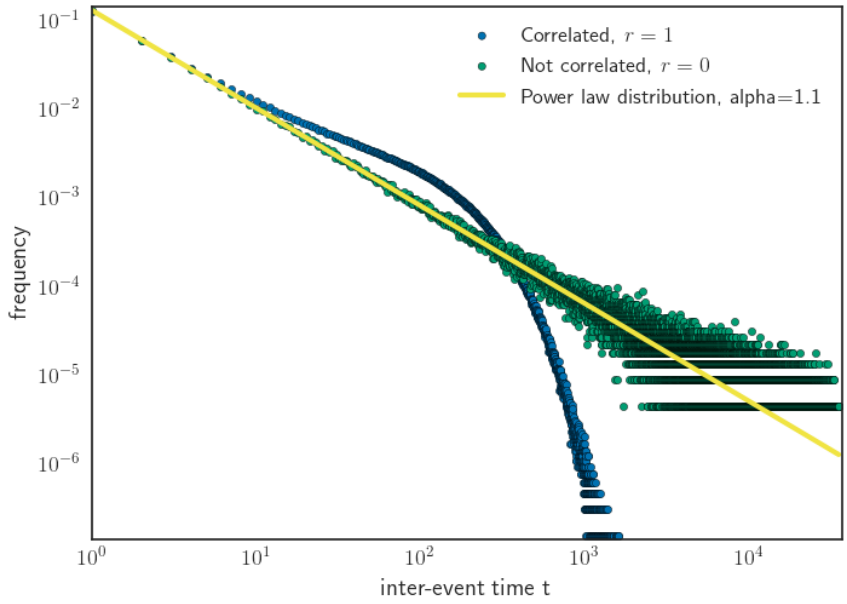
$$c(\tau) = s(\tau) \left(1 - \sum_{t=1}^{\tau} r(t)\right) + \sum_{t=1}^{\tau-1} \left(1 - \sum_{i=1}^{t} s(i)\right) r(t)\, c(\tau - t) \tag{3.1}$$

We still need to simplify this recurrence equation because $r(t)$ represents the probability that a neighbour contacts the individual at time t and not before. This probability of course depends on the degree of the individual, but also on the degree and dynamics of its neighbours, because neighbours call their friends at random. A possible approach would be to use a mean field approximation, either based on degrees, or on invidividuals.

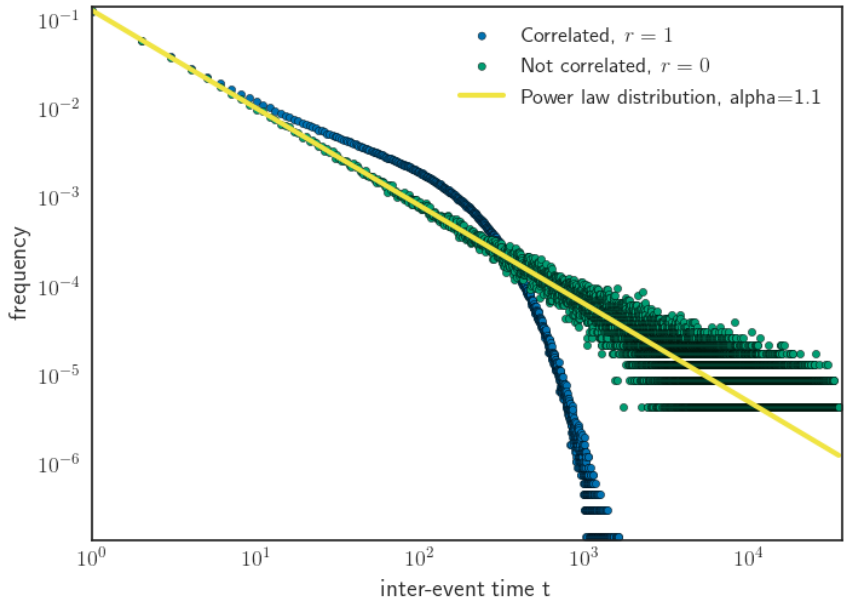
On figure 3.11, we compare the inter-event time distributions obtained using systematic rescheduling ($r = 1$) for both Poissonian and bursty activities. The theoretical distribution described by equation 3.1 is also part of the figure and is very close to the empirical result. Computing this distribution was done with the help of a simplifying assumption. We assume that $r(t)$ is independent of the neighbours and memoryless. At a given instant, there is a probability $p = \langle \Delta_e \rangle^{-1}$ that an individual receives a call, independently of the current time. The value of $\langle \Delta_e \rangle$ is estimated from the data, and $r(t)$ is thus a geometric distribution:

$$r(t) = p\,(1-p)^{t-1}.$$

The figure also shows that the curves for Poissonian and bursty activities exhibit similar tails, which may lead to similar spreading dynamics. We will observe this is the next section.



(A) BA network with $m = 100$ and $\langle k \rangle = 200$.



(B) ER network with $p = 0.02$ and $\langle k \rangle = 200$.

FIGURE 3.10: Comparison of the distributions of bursty activity inter-event time with and without time correlation. The simulation is done on networks of size $n = 10000$, for 40000 timestamps. The bursty activity is realized by sampling the inter-event time from a power law $Ct^{-\alpha}$ with $\alpha = 1.1$ and a cut-off $\epsilon = 10^{-5}$ before normalization using C (see section 3.1.2 for details about normalizationf).

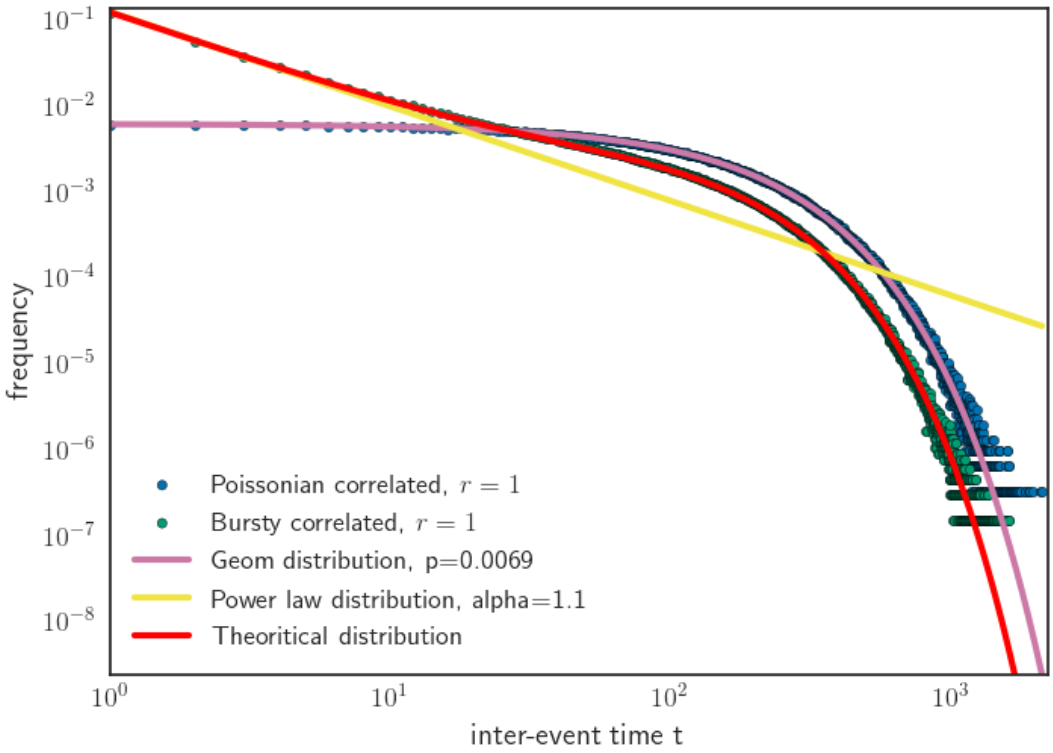


FIGURE 3.11: Comparison of the correlated inter-event time distributions using Poissonian activity (10 calls/day) and bursty activity. Both simulations were run for 40000 timestamps on a BA network of size $n = 10000$ ($m = 100$, $\langle k \rangle = 200$). The theoretical distribution is the distribution defined by equation 3.1, using an approximation for $r(t)$.

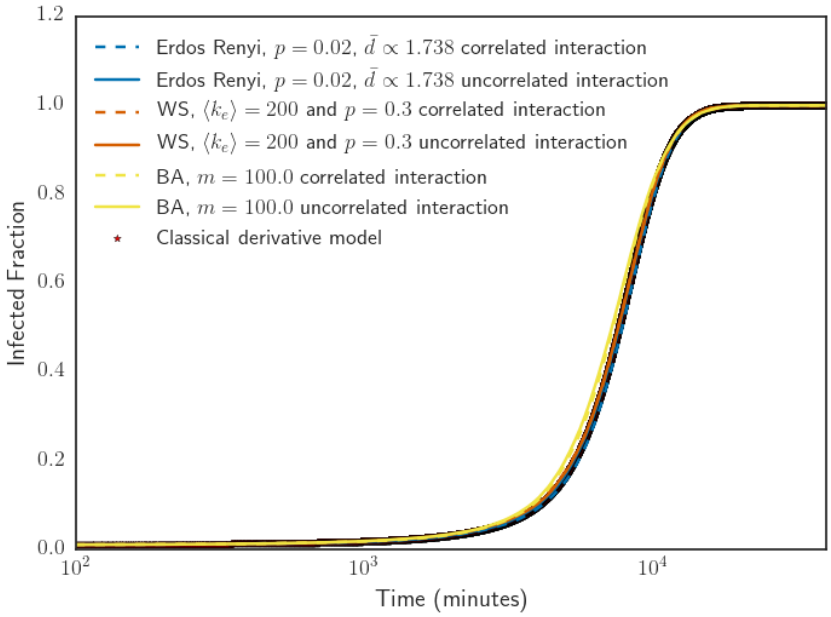
3.5 Time Correlations on Different Network Models

We observed that the time correlations impact the activity distribution that aren't memoryless, such as the Power Law. On the other hand, it has no impact on memoryless distribution such as the Poissonian activity. The results presented on the figures 3.12a and 3.13a confirm that the time correlation has no impact on the Poissonian activity. These results highlight that the spreading on BA network is eased by the topology.

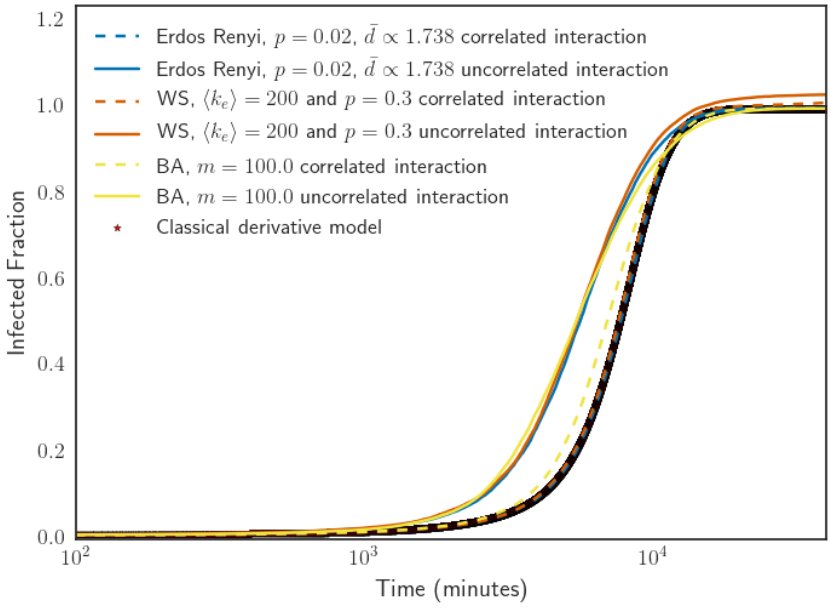
When there are time correlations, the tail of the Power Law activity is similar to the tail of the Poissonian activity. Nevertheless, as underlined on the figure 3.11, short inter-event times are less probable while in contrast average inter-event times are more probable.

Due to similarities between the poissonian and power law with time correlations, we can expect that the spreading won't be too different. In fact, as we can observe on the figures 3.12 and 3.13b, the bursty activity with the time correlation is generally close to the analytical results. However, the spreading is easier and faster on the BA networks as predicted in the section 1.3.5 and as illustrated on the figures 3.12a and 3.13a.

Furthermore, the spreading is faster in the early stage but slower in the late stage and the epidemic sizes are smaller when the activity is bursty and uncorrelated, confirming the results of Vazquez et al. [Vaz+07].

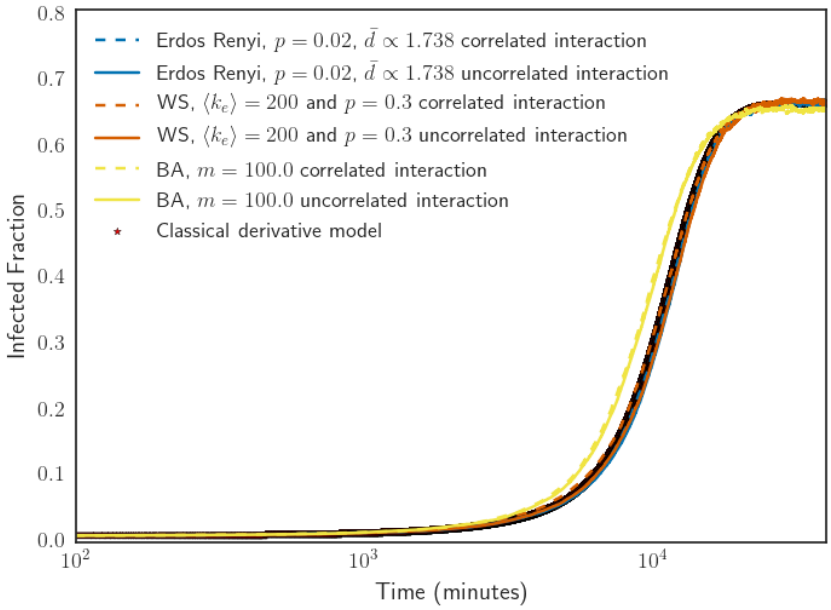


(A) Poissonian activity (10 calls/day).

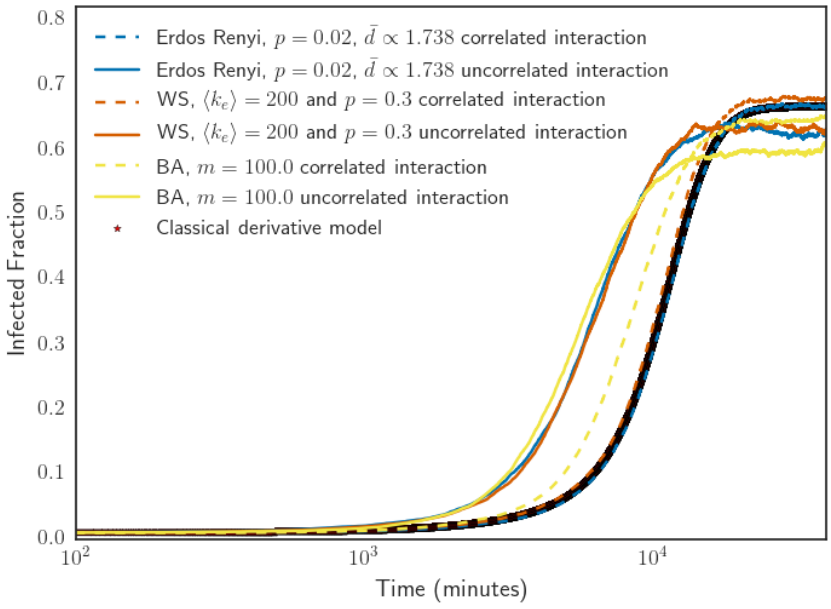


(B) Bursty activity

FIGURE 3.12: SI propagation over ER, WS and BA networks of size $n = 10000$. The average degree is $\langle k \rangle = 200$. All models correspond to the same R_0 value and the normalization is performed by varying the value of β.

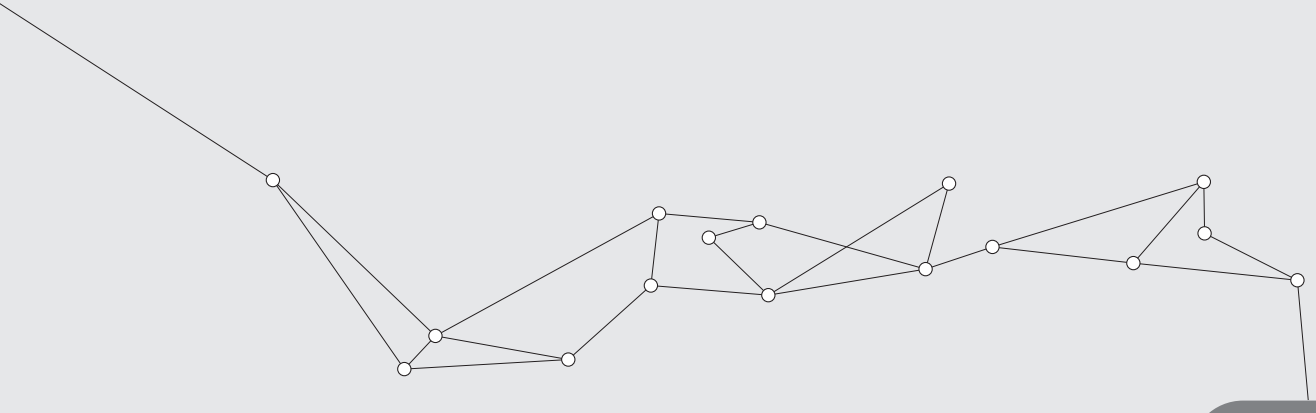


(A) Poissonian activity (10 calls/day).



(B) Bursty activity

FIGURE 3.13: SIS propagation over an ER, WS, BA network of size 10000 with average degree $< k >= 200$. The model use the same R_0, normalized via the β.



Conclusion

We begun this thesis with a journey presenting different epidemics models in historical order of appearance. Starting from the classical compartmental models, we progressively reduced the strength of some assumptions and started studying more realistic models. For example, letting go of the perfect mixing assumption allowed us to present more advanced network models. We also discussed the bursty nature of human communication, and the impact it has on epidemic spreading.

One of the important results presented in the first part of our work is the existence of an epidemic threshold below which epidemics are likely to die, or to only reach a small fraction of the total population. This result is of prime importance in the design of vaccination strategies which aim at increasing the threshold to ease the eradication of a disease.

Network epidemiology is a booming field with many important problems and opportunities to solve them. The review of 34 publications from recent years allowed us to identify some of the trends of the field. As years go by, studies tend toward more and more complex models for which no analytical solution is available. To circumvent this issue, our approach was to propose a flexible distributed simulator for generic social contacts.

In collaboration with *Real Impact Analytics*, we designed and implemented a Distributed Event Driven simulator capable of generating a dynamic temporal network in the form of Call Data Records. This simulator is a large program consisting of 7000 lines of Scala code, and relying on many standard libraries such as Spark and Akka. It is also well-tested, with more than 2300 lines of unit-testing code.

One of our main design goals was flexibility, since it is one of the major advantages of resorting to simulation. We believe our final version of the simulator to be both generic and extensible. It is able to generate datasets using any network topology. Implemented behaviours include the standard Poissonian and bursty activities, as well as a framework for custom activity profiles. Furthermore it could easily be extended to support mobility aspects and network dynamics.

Even though we had to trade performances for generality to some degree, the program is still reasonably fast, as illustrated by benchmarks conducted in chapter 2. We successfully tested our system on 4 machines provided by *Real Impact Analytics*, but the scalability to a more important cluster still has to be investigated. Fault-tolerance could also be improved.

In chapter 3, the flexibility of the simulator was put to good use when studying the impact of different assumptions on the spreading of epidemics. Our first experiments highlighted the impact of the diameter of networks and illustrated the differences between the perfect mixing assumption of classical models, and spreading properties of ER networks.

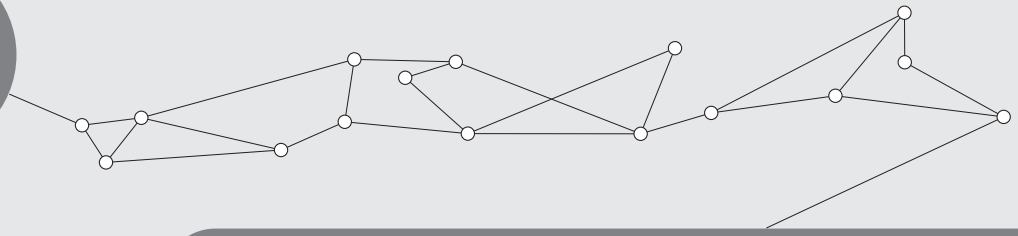
We then studied the epidemic threshold of SIS infections over ER and BA networks. As

predicted by theoretical models, ER networks exhibit thresholds similar to the classical models. In contrast, the threshold is lower on scale-free networks. This difference increases with the size of the network, but the difference in threshold between ER and BA networks is contained to less than an order of magnitude, for network sizes of at least up to 1 million individuals. In addition, we were able to provide empirical confirmations of the mean field approximation which predicts the threshold to decrease as the BA networks get larger.

Our final experiments studied the impact of time correlations on inter-event time distributions and on epidemic spreading. These correlations were shown to have no impact when considering memoryless activity patterns. However, they result in a visible cutoff of the tail of the inter-event time distribution when activity relies on memory. This is for example the case with burstiness. These experiments also underlined the impact of burstiness, which accelerates the early stages of the spreading while slowing it down near the end, resulting in reduced final size in the SIS case.

Potential future work includes challenging new kinds of assumptions, or analyzing the impact of vaccination, prevention and quarantine strategies. Many models have been suggested to explain the burstiness of human behaviors, and could be included in the simulator to analyze their impact on epidemic spreading.

A



Comparison of Different Epidemic Spreading Studies

To compare different models and approaches to the study of epidemic spreading, we propose a systematic review method in the spirit of the reviews presented by Serrano, Iglesias, and Garijo and Shamshirband et al. [SIG15; Sha+13]. We compare 34 articles about epidemics through the lens of 6 key questions:

Q1: Does the study use a static graph (S), a dynamic one (D), or compares them (S vs D)?

Q2: Does the study use a scale-free network?

Q3: Does it make the poissonian activity assumption (P) or the burstiness assumption (B)?

Q4: Was data-driven simulation used?

Q5: What is the epidemic model considered (SI/SIS/SIR/...)?

Q6: What are the main theoretical approaches used?

Table A.1 presents brief summaries of article results, and the answers to the 6 questions are presented in table A.2. Furthermore, the answers of questions 1 to 4 are summarized on the Venn diagram 1.9.

Table A.1: Summaries of the 34 articles.

Key	Citation	Title	Results
T1	[SS88]	The Spread and Persistence of Infectious-Diseases in Structured Populations	Determine the critical size of the subpopulation for which there is no threshold for the epidemic spreading in a subpopulation model.
T2	[All94]	Some discrete-time SI, SIR, and SIS epidemic models	Derive the solution to the non-linear discrete equation of SI, SIR and SIS. Show that under certain parameters, there is a period-doubling and chaos in the SIS case.
T3	[Hes96]	Disease in Metapopulation Models : Implications for Conservation	Model different network of subpopulation (Island, spider, necklace and loop)
T4	[WS98]	Collective dynamics of 'small-world' networks.	Propose "small world" network which exhibits the small world property and a high clustering. Show that the duration of the epidemics is correlated with the average path length.
T5	[Kee99]	The effects of local spatial structure on epidemiological invasions.	Model the correlation between individuals, using a pair-wise model. Their experiences highlight that when the degrees are small and that there is a high clustering, the spreading is smaller than what the classical mean-field model predicts and the thresholds get bigger and bigger.
T6	[Het00]	The Mathematics of Infectious Diseases	Derived the threshold for SIR epidemic and endemic model and for MSEIR and SEIR with or without age groups. Take also birth and death into account.
T7	[PSV01b]	Epidemic spreading in scale-free networks	Show that for scale-free network with an exponent $2 < \alpha \leq 3$, there is no threshold.
T8	[PSV01a]	Epidemic dynamics and endemic states in complex networks	Study the spreading on WS graphs and on BA graphs. Showing that for the spreading on WS graphs, there is a threshold while on scale free graph with $2 < \alpha \leq 3$, there is no threshold.

TABLE A.1: Summaries of the 34 articles.

Key	Citation	Title	Results
T9	[New02]	The spread of epidemic disease on networks.	Solve SIR cases in which probabilities of infection are non-uniform and correlated. Furthermore, the degree distribution is arbitrary. They showed that the statistical properties of a disease depend only on the transmissibility T, the a priori probability of transmission of the disease between two individuals and not the individual rates and time of infection. Finally, derived that for power law degree distribution with $2 < \alpha \leq 3$ derived that there is no threshold !
T10	[MPSV02]	Epidemic outbreaks in complex heterogeneous networks	Solve mean field approximation of the SIR for WS and scale-free network. Show that for scale-free network with $2 < \alpha \leq 3$ we have no threshold for which major outbreak is impossible.
T11	[PSV02]	Epidemic dynamics in finite size scale-free networks	Show that for finite size power law graph with and without the cut-off, even though there is a threshold, the thresholds decrease with the size of the graph and that the threshold is significantly smaller than the one derives for the homogeneous mixing.
T12	[VM03]	Resilience to damage of graphs with degree correlations.	Using bond percolation study the impact of the degree correlation in scale-free network on the SIR spreading showing that there is a threshold if the network is highly disassortative.
T13	[BPSV03]	Epidemic Spreading in Complex Networks with Degree Correlations	Using a mean field approximation show that on disassortative or assortative scale-free network, a threshold doesn't exist for SIS spreading.
T14	[Vaz06]	Polynomial growth in branching processes with diverging reproductive number	Study spreading on scale free dynamic network with an exponent $2 < \alpha \leq 3$. They showed that it changes the growth pattern, the fraction of new infected at each time step (expo followed by polynomial vs standard exponential growth).
T15	[Vaz+07]	Impact of non-Poissonian activity patterns on spreading processes	Show on email data that the Poisson approximation underestimate the decay of the spreading and propose a renewal process to take into account the non-Poissonian activity.
T16	[VM07]	Susceptible-infected-recovered epidemics in dynamic contact networks.	Incorporate the fact that we frequently break or make social relationship. Propose a model which interpolate smoothly between static network model and mass action. Furthermore, evaluate it on data from a high school.
T17	[Onn+07]	Structure and tie strengths in mobile communication networks.	Show the impact of communities using a weighted network extracted from email data . Once an individual is infected in a community, the disease spread quickly within it. While the spreading is slow between the communities because of the small weight between communities.
T18	[ZRG08]	Infection spreading in a population with evolving contacts	Show that if enough susceptible people break their links with infected people, the infection can die in the SIS case.
T19	[REE08]	Dynamic social networks and the implications for the spread of infectious disease	Based on a diary-survey of 49 participants over 14 non-consecutive days. Show that for short SIR epidemics on this small data set, the unweigthed static network assumption is enough.
T20	[CD09]	Contact processes on random graphs with power law degree distributions have critical value 0	Using a branching process and the localy tree like assumption, show that there is no threshold for power law degree distribution network even for $\alpha > 3$.
T21	[VM09]	Epidemic thresholds in dynamic contact networks.	Derive R0 and the epidemic thresholds for SIR on simple dynamic contact random network, showing that static network assumption can be inadequate.
T22	[VK09]	Representing the UK's cattle herd as static and dynamic networks.	Study the impact of different static and dynamic network assumption on the SIR spreading. The networks are derived from data about the UK's cattle herd. Show that the static networks fail to approximate the result of the dynamic networks. Furthermore, the difference between the unweighted static and weighted static networks is clear. Finally, weighted static networks are closed to the dynamic network when the transmission rate is low but differentiate when the transmission rate is long !
T23	[KN10]	Message passing approach for general epidemic models.	Using a time dependent message passing formulation, where the messages are the probability that the disease didn't pass to a neighbour by time t, derive exact result for SIR spreading with arbitrary inter-event time distribution such as power law on tree-like network and a upper bound for non tree like network. It is unclear if the approach is possible for SIS.

TABLE A.1: Summaries of the 34 articles.

Key	Citation	Title	Results
T24	[Kar+10]	Small But Slow World: How Network Topology and Burstiness Slow Down Spreading	Compare the impact of different correlation on SI spreading via a null model on real data. The assumptions are community structure, weight-topology correlation, bursty single-edge dynamics, event-event correlation between edges. Show that the spreading is the fastest when all these correlations are removed except the daily pattern. Furthermore, the burstiness slows down the spreading while the temporal correlation between adjacent edges have slight accelerating effect.
T25	[Ste+11]	Simulation of a SEIR infectious disease model on the dynamic contact network of conference attendees	Showed that a static network (homogeneous topology of contact) fails to reproduce the size of the epidemic compared to a temporal network. But a delta t aggregation of the temporal network (here daily) can be a good approximation depending on the duration of the spreading.
T26	[MML11]	Dynamic strength of social ties in information spreading	Show the influence of temporal patterns on the spreading based on mobile phone calls (20 million people). Burst slow down the propagation at large scales while conversation, trains of calls, favour local rapid cascades.
T27	[RLH11]	Simulated epidemics in an empirical spatiotemporal network of 50,185 sexual contacts	Study SI and SIR on empirical temporal network and show that temporal correlations accelerate outbreaks in the early phase while network topology slows down the epidemic.
T28	[Kiv+12]	Multiscale analysis of spreading in a large communication network	Show the importance of the burstiness, the event correlations and of the temporal pathways for the SI spreading using mobile data. Characterize their effects using a relay time formulation. The relay time is the time it takes for a newly infected node to spread the infection to another node via an event it participates in.
T29	[RG12]	Influence of network dynamics on the spread of sexually transmitted diseases	Provide a model generic enough to take into account the dynamics of the relationship and to be fitted on the data. While remaining possible to be studied analytically using mean field techniques. Analyse the steady state, derive that the threshold on static network is a lower bound for the threshold of dynamic networks.
T30	[LTD13b]	Burstiness and spreading on temporal networks	Show how temporal pattern in a complex network can change dramatically the properties of the spreading. Identify that the time ordering of the events is more important than the inter-event time distribution. The importance of an edge isn't in the number of activation but rather in how much it participates in the spreading.
T31	[MGK13]	Suppression of epidemic outbreaks with heavy-tailed contact dynamics	Show that heterogeneity of contacts can suppress the disease transmissibility and by doing so, reduce the size of the outbreak. This contrast with the fact that the heterogeneity of the network removes the thresholds. For inter-event time following a power law distribution with $\alpha = 2$, the disease doesn't take off on random network. But on scale-free network, and $\alpha \leq 2$, it depends on the recovery time, but the infection remains small. Show that the bigger the maximum inter-event time, the heavier the tail, the slower is the spreading.
T32	[RB13]	Bursts of Vertex Activation and Epidemics in Evolving Networks	Show the impact of heterogeneous contact patterns on SIR epidemic in comparison to the homogeneous case. The first causing earlier and larger epidemics.
T33	[KPV14]	Time varying networks and the weakness of strong ties.	Using mobile phone datasets, show that strong ties may slow down the spreading because the information is confined to individuals with recurring communication patterns.
T34	[DLR15]	Diffusion on networked systems is a question of time or structure.	Study the impact of network structure such as communities and temporal properties such as inter-event time. They identify the situations when the structural properties are important for the prediction on the spreading.

TABLE A.2: Answers to the 6 key questions

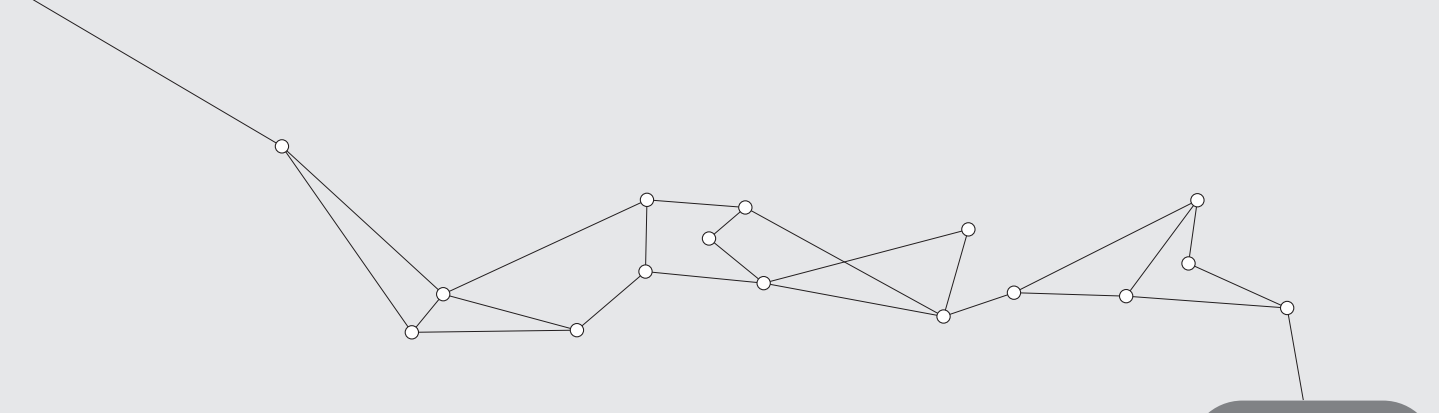
Key	Citation	Title	Q1	Q2	Q3	Q4	Model	Approach
T1	[SS88]	The Spread and Persistence of Infectious-Diseases in Structured Populations	S	✗	P	✗	SIR	Differential equations
T2	[All94]	Some discrete-time SI, SIR, and SIS epidemic models	S	✗	P	✗	discrete SIS, SIR, SI	Discrete derivative
T3	[Hes96]	Disease in Metapopulation Models : Implications for Conservation	S	✗	P	✗	SIR	Differential equations
T4	[WS98]	Collective dynamics of 'small-world' networks.	S	✗	P	✗	SIR	Simulation over a generated WS graph
T5	[Kee99]	The effects of local spatial structure on epidemiological invasions.	S	✗	P	✗	SIR	Use behaviour on pairs [A B]
T6	[Het00]	The Mathematics of Infectious Diseases	S	✗	P	✗	MSEIR, SEIR, SI, SIS	Differential equations
T7	[PSV01b]	Epidemic spreading in scale-free networks	S	✓	P	✗	SIS	Dynamical mean field
T8	[PSV01a]	Epidemic dynamics and endemic states in complex networks	S	✓	P	✗	SIS	Dynamical mean field
T9	[New02]	The spread of epidemic disease on networks.	S	✓	P	✗	SIR	Bond percolation using a generating function for the degree distribution.
T10	[MPSV02]	Epidemic outbreaks in complex heterogeneous networks	S	✓	P	✗	SIR	Dynamical mean field
T11	[PSV02]	Epidemic dynamics in finite size scale-free networks	S	✓	P	✗	SIS	Dynamical mean field
T12	[VM03]	Resilience to damage of graphs with degree correlations.	S	✓	P	✗	SIR	Bond percolation
T13	[BPSV03]	Epidemic Spreading in Complex Networks with Degree Correlations	S	✓	P	✗	SIS	Mean field
T14	[Vaz06]	Polynomial growth in branching processes with diverging reproductive number	D	✓	P	✓	SI	Generating function for the time, branching process (infection tree).
T15	[Vaz+07]	Impact of non-poissonian activity patterns on spreading processes	D	[1]	B	✓	SI	Renewal process
T16	[VM07]	Susceptible-infected-recovered epidemics in dynamic contact networks.	D	✓	B	✓	SIR	Neighbour exchange model and generating function.
T17	[Onn+07]	Structure and tie strengths in mobile communication networks.	W	[1]	P	✓	SI	Data driven
T18	[ZRG08]	Infection spreading in a population with evolving contacts	D	✗	P	✗	SIS	Mean field
T19	[REE08]	Dynamic social networks and the implications for the spread of infectious disease	S	[1]	P	✓	SIR	Data driven
T20	[CD09]	Contact processes on random graphs with power law degree distributions have critical value 0	S vs D	✓	P	✗	SIR, SIS	Contact processes
T21	[VM09]	Epidemic thresholds in dynamic contact networks.	S vs D	✓	P	✗	SIR	Neighbour exchange model and generating function.
T22	[VK09]	Representing the UK's cattle herd as static and dynamic networks.	S vs D	[1]	[1]	✓	SIR	Data driven
T23	[KN10]	Message passing approach for general epidemic models.	S	[2]	B	✗	SIR	Message passing

[1] From data
[2] locally tree like network

TABLE A.2: Answers to the 6 key questions

Key	Citation	Title	Q1	Q2	Q3	Q4	Model	Approach
T24	[Kar+10]	Small But Slow World: How Network Topology and Burstiness Slow Down Spreading	D	1	B	✓	SI	Null model
T25	[Ste+11]	Simulation of an SEIR infectious disease model on the dynamic contact network of conference attendees	S vs D	1	1	✓	SEIR	Data driven
T26	[MML11]	Dynamical strength of social ties in information spreading	D	1	B	✓	SIR	Null model
T27	[RLH11]	Simulated epidemics in an empirical spatiotemporal network of 50,185 sexual contacts	S vs D	1	1	✓	SI, SIR, SII(case of HIV-1)	Null model
T28	[Kiv+12]	Multiscale analysis of spreading in a large communication network	D	1	B [1]	✓	SI	Temporal network, null model
T29	[RG12]	Influence of network dynamics on the spread of sexually transmitted diseases	S vs D	✓	P	✓	SIS	Mean field
T30	[LTD13b]	Burstiness and spreading on temporal networks	D?	3	P	✗	SIR	Random walker and non-markovian random process
T31	[MGK13]	Suppression of epidemic outbreaks with heavy-tailed contact dynamics	S	✓	B	✗	SIR	renewal theory and branching factor
T32	[RB13]	Bursts of Vertex Activation and Epidemics in Evolving Networks	D	1	B [1]	✗	SI, SIR	Temporal network
T33	[KPV14]	Time varying networks and the weakness of strong ties.	D	✓[1]	P	✓	ISR (ignorant, spreader and stifler)	Egocentric network
T34	[DLR15]	Diffusion on networked systems is a question of time or structure.	D	✓	B	✓	Yes	random walk and linear multi-agent system

[3] locally tree like network



References

[AJB99] Réka Albert, Hawoong Jeong, and Albert Laszlo Barabasi. "Internet: Diameter of the World-Wide Web". In: *Nature* 401.6749 (1999), pp. 130–131. ISSN: 0028-0836. DOI: 10.1038/43601.

[All+90] Linda Allen et al. "A mathematical analysis and simulation of a localized measles epidemic". In: *Applied Mathematics and Computation* 39.1 (1990), pp. 61–77. ISSN: 00963003. DOI: 10.1016/0096-3003(90)90122-J.

[All94] Linda J S Allen. "Some discrete-time SI, SIR, and SIS epidemic models". In: *Mathematical Biosciences* 124.1 (1994), pp. 83–105. ISSN: 00255564. DOI: 10.1016/0025-5564(94)90025-6.

[AM91] R M Anderson and R M May. "Infectious Diseases of Humans: Dynamics and Control". In: *Oxford University Press* (1991).

[Ana] Real Impact Analytics. *Using telecom data for social good*. URL: https://realimpactanalytics.com/en/data-for-good-cases (visited on 05/01/2016).

[Arm+96] Joe Armstrong et al. *Concurrent Programming in Erlang*. 2nd Editio. Prentice-Hall, 1996.

[BA99] Albert-László Barabási and Reka Albert. "Emergence of scaling in random networks". In: *Science* 286.5439 (1999), p. 11. ISSN: 00368075. DOI: 10.1126/science.286.5439.509.

[Ban+10] Shweta Bansal et al. "The dynamic nature of contact networks in infectious disease epidemiology". In: *Journal of Biological Dynamics* 4.5 (2010), pp. 478–489. ISSN: 1751-3758. DOI: 10.1080/17513758.2010.503376.

[Bar05] Albert-Laszlo Barabási. "The origin of bursts and heavy tails in human dynamics". In: *Nature* 435.May (2005). DOI: 10.1038/nature03526.1..

[Bar12] Albert-László Barabási. "Chapter 5: The Barabási-Albert Model". In: *Network Science* 0 (2012), pp. 1–44.

[Bar15] Albert-László Barabási. "Network Science Chapter 10: Spreading Processes". In: *Book* (2015).

[Bar16] Albert-László Barabási. *Network Science*. Cambridge University Press, 2016. ISBN: 1107076269.

[Bar67] Robert Bartoszynski. "Branching processes and the theory of epidemics". In: *Berkeley Symposium on Mathematical Statistics and Probability*. 1967, pp. 259–269.

[Ber+13] Michele Berlingerio et al. "AllAboard: A system for exploring urban mobility and optimizing public transport using cellphone data". In: *Machine Learning and Knowledge Discovery in Databases*. Vol. 8190. 2013. Chap. 51, pp. 663–666. ISBN: 9783642409943. DOI: 10.1007/978-3-642-40994-3.

[Bor10] Dhruba Borthakur. "HDFS Architecture Guide". In: (2010), pp. 1–14.

[BPSV03] Marián Boguñá, Romualdo Pastor-Satorras, and Alessandro Vespignani. "Epidemic Spreading in Complex Networks with Degree Correlations". In: *Statistical Mechanics of Complex Networks* January (2003), pp. 127–147. ISSN: 0075-8450. DOI: 10.1007/978-3-540-44943-0_8.

[BR02] B Bollobás and Om Riordan. "Mathematical results on scale-free random graphs". In: *Handbook of Graphs and Networks: From the Genome to the Internet* (2002), pp. 1–38. DOI: 10.1002/3527602755.ch1.

[Cas+09] Claudio Castellano et al. "Statistical physics of social dynamics". In: *Reviews of modern physics* 81.2 (2009), pp. 591–646. ISSN: 0034-6861. DOI: 10.1103/RevModPhys.81.591.

[CBMT96] Bernadette Charron-Bost, Friedemann Mattern, and Gerard Tel. "Synchronous, asynchronous, and causally ordered communication". In: *Distributed Computing* 9.4 (1996), pp. 173–191. ISSN: 0178-2770. DOI: 10.1007/s004460050018.

[CD09] Shirshendu Chatterjee and Rick Durrett. "Contact processes on random graphs with power law degree distributions have critical value 0". In: *Annals of Probability* 37.6 (2009), pp. 2332–2356. ISSN: 00911798. DOI: 10.1214/09-AOP471.

[CH03] Reuven Cohen and Shlomo Havlin. "Scale-free networks are ultrasmall." In: *Physical review letters* 90.5 (2003), p. 058701. ISSN: 0031-9007. DOI: 10.1103/PhysRevLett.90.058701.

[CHBA03] Reuven Cohen, Shlomo Havlin, and Daniel Ben-Avraham. "Efficient immunization strategies for computer networks and populations." In: *Physical review letters* 91.24 (2003), p. 247901. ISSN: 0031-9007. DOI: 10.1103/PhysRevLett.91.247901.

[Dec15] Adeline Decuyper. "Big Data for Modeling Human Behavior: Applications Using Mobile Phone Data". PhD thesis. Université Catholique de Louvain, 2015.

[Dev+14] Pierre Deville et al. "Dynamic population mapping using mobile phone data". In: *Proceedings of the National Academy of Sciences* 111.45 (2014), pp. 15888–15893. ISSN: 0027-8424. DOI: 10.1073/pnas.1408439111.

[DLR15] Jean-Charles Delvenne, Renaud Lambiotte, and Luis E C Rocha. "Diffusion on networked systems is a question of time or structure." In: *Nature communications* 6 (2015), p. 7366. ISSN: 2041-1723. DOI: 10.1038/ncomms8366.

[EMS03] Jean-Pierre Eckmann, Elisha Moses, and Danilo Sergi. "Dialog in e-Mail Traffic". In: (2003), pp. 1–5.

[ER59] P Erdös and a Rényi. "On random graphs". In: *Publicationes Mathematicae* 6 (1959), pp. 290–297. ISSN: 00029947. DOI: 10.2307/1999405.

[Fel91] Scott L. Feld. "Why Your Friends Have More Friends Than You Do". In: *American Journal of Sociology* 96.6 (1991), p. 1464. ISSN: 0002-9602. DOI: 10.1086/229693.

[FT98] Alois Ferscha and S K Tripathi. "Parallel and distributed simulation of discrete event systems". In: *Event London* (1998), pp. 1–65.

[Fuj00] Richard M Fujimoto. *Parallel and Distributed Simulation Systems*. Wiley, 2000, p. 300. ISBN: 0471183830.

[Fuj99] Richard M Fujimoto. "Parallel and distributed simulation". In: *Proceedings of the 1999 Winter Simulation Conference* 1.6-7 (1999), pp. 122–131. ISSN: 13837621. DOI: 10.1145/324138.324176.

[Gan+16] Yerali Gandica et al. "On the origin of burstiness in human behavior: The wikipedia edits case". In: (2016), pp. 1–12.

[Gat10] Bill Gates. "The Next Epidemic - Lessons from Ebola". In: *The New England Journal of Medicine* 363.1 (2010), pp. 1–3. DOI: 10.1056/NEJMp1502918.

[HBS73] Carl Hewitt, Peter Bishop, and Richard Steiger. "A Universal Modular Actor Formalism for Artificial Intelligence". In: *IJCAI* (1973), pp. 235–245.

[Her+10] A. Hernando et al. "Unravelling the size distribution of social groups with information theory in complex networks". In: *European Physical Journal B* 76.1 (2010), pp. 87–97. ISSN: 14346028. DOI: 10.1140/epjb/e2010-00216-1.

[Hes96] George Hess. "Disease in Metapopulation Models : Implications for Conservation". In: *Ecology* 77.5 (1996), pp. 1617–1632.

[Het00] Herbert W Hethcote. "The Mathematics of Infectious Diseases". In: 42.4 (2000), pp. 599–653. ISSN: 0036-1445. DOI: 10.1137/S0036144500371907.

[HO07] Philipp Haller and Martin Odersky. "Actors That Unify Threads and Events". In: *Coordination 2007, Lncs 4467* (2007), pp. 171–190. ISSN: 03029743. DOI: 10.1007/978-3-540-72794-1_10.

[HS12] Petter Holme and Jari Saramäki. "Temporal networks". In: *Physics Reports* 519.3 (2012), pp. 97–125. ISSN: 03701573. DOI: 10.1016/j.physrep.2012.03.001.

[Kar+10] Márton Karsai et al. "Small But Slow World: How Network Topology and Burstiness Slow Down Spreading". In: (2010), pp. 1–4. DOI: 10.1103/PhysRevE.83.025102.

[KE02] Konstantin Klemm and Víctor M. Eguíluz. "Growing scale-free networks with small-world behavior". In: *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 65.5 (2002), pp. 1–4. ISSN: 15393755. DOI: 10.1103/PhysRevE.65.057102.

[Kee99] M J Keeling. "The effects of local spatial structure on epidemiological invasions." In: *Proceedings. Biological sciences / The Royal Society* 266.1421 (1999), pp. 859–67. ISSN: 0962-8452. DOI: 10.1098/rspb.1999.0716.

[Ken75] David G Kendall. "The Genealogy of Genealogy branching processes before (and after) 1873". In: *Bulletin of the London Mathematical Society* 7.April (1975), pp. 225–253. ISSN: 0024-6093. DOI: 10.1112/blms/7.3.225.

[Kiv+12] Mikko Kivelä et al. "Multiscale analysis of spreading in a large communication network". In: *Journal of Statistical Mechanics: Theory and Experiment* 2012.03 (2012), P03005. ISSN: 1742-5468. DOI: 10.1088/1742-5468/2012/03/P03005.

[KN10] Brian Karrer and M. E J Newman. "Message passing approach for general epidemic models". In: *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 82.1 (2010), pp. 1–9. ISSN: 15393755. DOI: 10.1103/PhysRevE.82.016101.

[KPV14] Márton Karsai, Nicola Perra, and Alessandro Vespignani. "Time varying networks and the weakness of strong ties." In: *Scientific reports* 4 (2014), p. 4001. ISSN: 2045-2322. DOI: 10.1038/srep04001.

[Kri12] Gautier Krings. "Extraction of information from large networks". PhD thesis. Université Catholique de Louvain, 2012.

[LTD13a] Renaud Lambiotte, Lionel Tabourier, and Jean-Charles Delvenne. "Burstiness and spreading on temporal networks". In: *The European Physical Journal B* 86.7 (2013), p. 320. ISSN: 1434-6028. DOI: 10.1140/epjb/e2013-40456-9.

[LTD13b] Renaud Lambiotte, Lionel Tabourier, and Jean-Charles Delvenne. "Burstiness and spreading on temporal networks". In: *The European Physical Journal B* 86.7 (2013), p. 320. ISSN: 1434-6028. DOI: 10.1140/epjb/e2013-40456-9.

[Lyn96] Nancy A. Lynch. *Distributed Algorithms*. Morgan Kaufmann, 1996. ISBN: 1558603484.

[Mal+08] R Dean Malmgren et al. "A Poissonian explanation for heavy tails in e-mail communication." In: *Proceedings of the National Academy of Sciences of the United States of America* 105.47 (2008), pp. 18153–18158. ISSN: 0027-8424. DOI: 10.1073/pnas.0800332105.

[Mas66] Abraham H. Maslow. *The Psychology of Science: A Reconnaissance*. First Edit. Joanna Cotler Books, 1966. ISBN: 0060341459.

[MGK13] Byungjoon Min, K.-I. Goh, and I.-M. Kim. "Suppression of epidemic outbreaks with heavy-tailed contact dynamics". In: *EPL (Europhysics Letters)* 103.5 (2013), p. 50002. ISSN: 0295-5075. DOI: 10.1209/0295-5075/103/50002.

[MH13] Naoki Masuda and Petter Holme. "Predicting and controlling infectious disease epidemics using temporal networks." In: *F1000prime reports* 5.March (2013), p. 6. ISSN: 2051-7599. DOI: 10.12703/P5-6.

[Mil67] Stanley Milgram. "The small world problem". In: *Psychology today* 1.1 (1967), pp. 61–67. ISSN: 01472011. DOI: 10.1007/BF02717530.

[Mis86] Jayadev Misra. "Distributed discrete-event simulation". In: *ACM Computing Surveys* 18.1 (1986), pp. 39–65. ISSN: 03600300. DOI: 10.1145/6462.6485.

[Mit04] Michael Mitzenmacher. "A Brief History of Generative Models for Power Law and Lognormal Distributions". In: *Internet Mathematics* 1.2 (2004), pp. 226–251. ISSN: 1542-7951. DOI: 10.1080/15427951.2004.10129088.

[MML11] Giovanna Miritello, Esteban Moro, and Rubén Lara. "Dynamical strength of social ties in information spreading". In: *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 83.4 (2011), pp. 3–6. ISSN: 15393755. DOI: 10.1103/PhysRevE.83.045102.

[Moo75] Gordon E. Moore. "Progress in digital integrated electronics". In: *1975 International Electron Devices Meeting* 21 (1975), pp. 11–13. ISSN: 1098-4232. DOI: 10.1109/N-SSC.2006.4804410.

[Mos+08] Joël Mossong et al. "Social contacts and mixing patterns relevant to the spread of infectious diseases". In: *PLoS Medicine* 5.3 (2008), pp. 0381–0391. ISSN: 15491277. DOI: 10.1371/journal.pmed.0050074.

[MPSV02] Y. Moreno, R. Pastor-Satorras, and A. Vespignani. "Epidemic outbreaks in complex heterogeneous networks". In: *European Physical Journal B* 26.4 (2002), pp. 521–529. ISSN: 14346028. DOI: 10.1007/s10051-002-8996-y.

[MS15] Esteban Moro and Jari Saramäki. "From seconds to months: multi-scale dynamics of mobile telephone calls". In: *The European Physical Journal B* 88 (2015), pp. 1–11. ISSN: 1434-6028. DOI: 10.1140/epjb/e2015-60106-6.

[New02] M. E J Newman. "The spread of epidemic disease on networks". In: *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 66.1 (2002). ISSN: 15393755. DOI: 10.1103/PhysRevE.66.016128.

[New03] Mark E J Newman. "The structure and function of complex networks". In: *Society for Industrial and Applied Mathemathics* 45.2 (2003), pp. 167–256.

[New05] M. E. J. Newman. "Power laws, Pareto distributions and Zipf's law". In: *Power laws, Pareto distributions and Zipf's law. Contemporary physics* 46.5 (2005), pp. 323–351. ISSN: 0010-7514. DOI: 10.1016/j.cities.2012.03.001.

[OB05] J Oliveira and Albert-László Barabási. "Darwin and Einstein correspondence patterns". In: *Nature* 437.October (2005), p. 1251. DOI: 10.0138/4371251a.

[OC12] Jukka Pekka Onnela and Nicholas A. Christakis. "Spreading paths in partially observed social networks". In: *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 85.3 (2012), pp. 1–12. ISSN: 15393755. DOI: 10.1103/PhysRevE.85.036106.

[Onn+07] J-P Onnela et al. "Structure and tie strengths in mobile communication networks." In: *Proceedings of the National Academy of Sciences of the United States of America* 104.18 (2007), pp. 7332–7336. ISSN: 0027-8424. DOI: 10.1073/pnas.0610245104.

[OSV08] Martin Odersky, Lex Spoon, and Bill Venners. *Programming in Scala*. 2008. ISBN: 0981531644.

[PS+15] Romualdo Pastor-Satorras et al. "Epidemic processes in complex networks". In: *Reviews of Modern Physics* 87.3 (2015), pp. 925–979. ISSN: 15390756. DOI: 10.1103/RevModPhys.87.925.

[PSV01a] Romualdo Pastor-Satorras and Alessandro Vespignani. "Epidemic dynamics and endemic states in complex networks". In: *Physical Review E* 63.6 (2001), p. 066117. ISSN: 1063-651X. DOI: 10.1103/PhysRevE.63.066117.

[PSV01b] Romualdo Pastor-Satorras and Alessandro Vespignani. "Epidemic spreading in scale-free networks". In: *Physical Review Letters* 86.14 (2001), pp. 3200–3203. ISSN: 00319007. DOI: 10.1103/PhysRevLett.86.3200.

[PSV02] Romualdo Pastor-Satorras and Alessandro Vespignani. "Epidemic dynamics in finite size scale-free networks". In: *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 65.3 (2002), pp. 1–4. ISSN: 15393755. DOI: 10.1103/PhysRevE.65.035108.

[RB13] Luis E C Rocha and Vincent D. Blondel. "Bursts of Vertex Activation and Epidemics in Evolving Networks". In: *PLoS Computational Biology* 9.3 (2013). ISSN: 1553734X. DOI: 10.1371/journal.pcbi.1002974.

[REE08] J M Read, K T D Eames, and W J Edmunds. "Dynamic social networks and the implications for the spread of infectious disease". In: *Journal of the Royal Society Interface* 5.26 (2008), pp. 1001–1007. ISSN: 1742-5689. DOI: 10.1098/rsif.2008.0013.

[RG12] Sebastian Risau-Gusman. "Influence of network dynamics on the spread of sexually transmitted diseases". In: *Journal of the Royal Society Interface* 9.71 (2012), pp. 1363–1372. ISSN: 1742-5689. DOI: 10.1098/rsif.2011.0445.

[RLH11] Luis E C Rocha, Fredrik Liljeros, and Petter Holme. "Simulated epidemics in an empirical spatiotemporal network of 50,185 sexual contacts". In: *PLoS Computational Biology* 7.3 (2011), pp. 1–9. ISSN: 1553734X. DOI: 10.1371/journal.pcbi.1001109.

[Sch+14] Ingo Scholtes et al. "Causality-driven slow-down and speed-up of diffusion in non-Markovian temporal networks." In: *Nature communications* 5 (2014), p. 5024. ISSN: 2041-1723. DOI: 10.1038/ncomms6024.

[Sha+13] Shahaboddin Shamshirband et al. "An appraisal and design of a multi-agent system based cooperative wireless intrusion detection computational intelligence technique". In: *Engineering Applications of Artificial Intelligence* 26.9 (2013), pp. 2105–2127. ISSN: 09521976. DOI: 10.1016/j.engappai.2013.04.010.

[SIG15] Emilio Serrano, Carlos A. Iglesias, and Mercedes Garijo. "A Survey of Twitter Rumor Spreading Simulations". In: *Program* (2015), pp. 113–122. ISSN: 0302-9743. DOI: 10.1007/978-3-642-40495-5.

[SS88] L Sattenspiel and C P Simon. "The Spread and Persistence of Infectious-Diseases in Structured Populations". In: *Math.Biosci.* 90 (1988), pp. 341–366. DOI: 10.1016/0025-5564(88)90074-0.

[Ste+11] Juliette Stehlé et al. "Simulation of an SEIR infectious disease model on the dynamic contact network of conference attendees". In: *BMC Medicine* 9 (2011), p. 87. ISSN: 1741-7015. DOI: 10.1186/1741-7015-9-87.

[Uga+11] Johan Ugander et al. "The Anatomy of the Facebook Social Graph". In: *Arxiv preprint arXiv* abs/1111.4 (2011), pp. 1–17.

[Vaz+06] Alexei Vazquez et al. "Modeling bursts and heavy tails in human dynamics". In: *Physical Review E - Statistical, Nonlinear, and Soft Matter Physics* 73.3 (2006), pp. 1–19. ISSN: 15393755. DOI: 10.1103/PhysRevE.73.036127.

[Vaz+07] Alexei Vazquez et al. "Impact of non-poissonian activity patterns on spreading processes". In: *Physical Review Letters* 98.15 (2007), pp. 1–4. ISSN: 00319007. DOI: 10.1103/PhysRevLett.98.158702.

[Vaz06] Alexei Vazquez. "Polynomial growth in branching processes with diverging reproductive number". In: *Physical Review Letters* 96.3 (2006), pp. 1–4. ISSN: 10797114. DOI: 10.1103/PhysRevLett.96.038702.

[Vaz07] Alexei Vazquez. "Impact of memory on human dynamics". In: *Physica A: Statistical Mechanics and its Applications* 373.i (2007), pp. 747–752. ISSN: 03784371. DOI: 10.1016/j.physa.2006.04.060.

[VH04] Peter Van Roy and Seif Haridi. *Concepts, Techniques, and Models of Computer Programming*. Ed. by MIT Press. MIT Press, 2004. ISBN: 0-262-22069-5.

[VK09] Matthew C Vernon and Matt J Keeling. "Representing the UK's cattle herd as static and dynamic networks." In: *Proceedings. Biological sciences / The Royal Society* 276.1656 (2009), pp. 469–76. ISSN: 0962-8452. DOI: 10.1098/rspb.2008.1009.

[VM03] Alexei Vázquez and Yamir Moreno. "Resilience to damage of graphs with degree correlations." In: *Physical review. E, Statistical, nonlinear, and soft matter physics* 67.2 (2003), p. 015101. ISSN: 1063-651X. DOI: 10.1103/PhysRevE.67.015101.

[VM07] Erik Volz and Lauren Ancel Meyers. "Susceptible-infected-recovered epidemics in dynamic contact networks." In: *Proceedings of the Royal Society of London B: Biological Sciences* 274.1628 (2007), pp. 2925–2933. ISSN: 0962-8452. DOI: 10.1098/rspb.2007.1159.

[VM09] Erik Volz and Lauren Ancel Meyers. "Epidemic thresholds in dynamic contact networks." In: *Journal of the Royal Society, Interface / the Royal Society* 6.32 (2009), pp. 233–241. ISSN: 1742-5689. DOI: 10.1098/rsif.2008.0218.

[Wal16] Mitchell Waldrop. "More than Moore". In: *Nature* 530 (2016), pp. 144–147. DOI: 10.1038/530144a.

[WS98] D J Watts and S H Strogatz. "Collective dynamics of 'small-world' networks." In: *Nature* 393.6684 (1998), pp. 440–2. ISSN: 0028-0836. DOI: 10.1038/30918.

[Yan+09] Yang Yang et al. "The Transmissibility and Control of Pandemic Influenza A (H1N1) Virus". In: 326.5953 (2009), pp. 729–733. DOI: 10.1126/science.1177373.The.

[Zah+10] Matei Zaharia et al. "Spark : Cluster Computing with Working Sets". In: *HotCloud'10 Proceedings of the 2nd USENIX conference on Hot topics in cloud computing* (2010), p. 10. ISSN: 03642348.

[Zim08] Armin Zimmermann. *Stochastic Discrete Event Systems: Modeling, Evaluation, Applications*. Springer, 2008. ISBN: 978-3-540-74172-5. DOI: 10.1007/978-3-540-74173-2.

[ZRG08] Damián H. Zanette and Sebastián Risau-Gusmán. "Infection spreading in a population with evolving contacts". In: *Journal of Biological Physics* 34.1-2 SPEC. ISS. (2008), pp. 135–148. ISSN: 00920606. DOI: 10.1007/s10867-008-9060-9.

[Lig16a] Lightbend Inc. *Fault-Tolerance for Akka*. 2016. URL: http://doc.akka.io/docs/akka/snapshot/scala/fault-tolerance.html (visited on 04/29/2016).

[Lig16b] Lightbend Inc. *What is Akka*. 2016. URL: http://doc.akka.io/docs/akka/2.4.4/intro/what-is-akka.html (visited on 04/27/2016).

Rue Archimède, 1 bte L6.11.01, 1348 Louvain-la-Neuve **www.uclouvain.be/epl**