Radboud University Nijmegen

Faculty of Science

The Critical Value of the Contact Process

Bachelor Thesis in Mathematics

Author:
Casper Algera

Supervisor:
dr. ir. Henk Don

Second reader:
dr. TBD TBD

June 10, 2025

Contents

Introduction

In 1974, a mathematician called Ted Harris introduced the contact process on a lattice. This process can be understood as a model for the spread of an infection. We consider points of the lattice, which can either be healthy or infected. Any infected site spreads the infection to its direct neighbours at a certain infection rate. Additionally, infected points randomly recover at a constant rate. One realisation of this process can be found in Figure 1. In this thesis we will study this process. We will look at varying rates of infection and ask ourselves the question, is there a rate for which the infection has a positive chance of surviving?

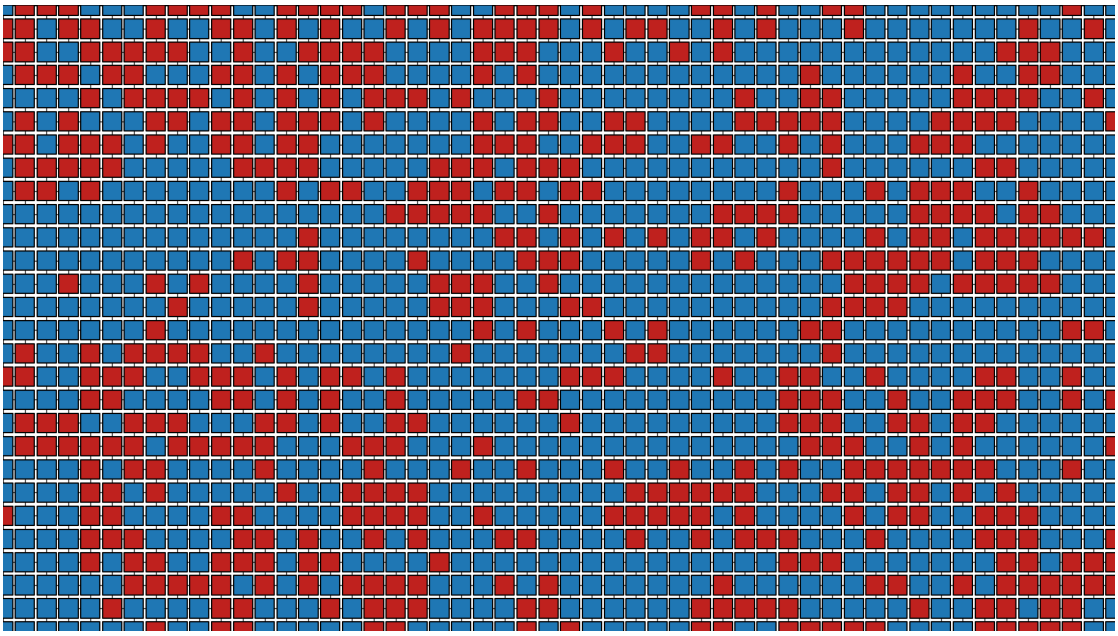


Figure 1: A realisation of the contact process on the square lattice. Red nodes are infected, blue ones are healthy.

There will turn out to be a specific rate at which this probability becomes strictly positive, starting from a single infection. This specific rate, called the critical value, will be the primary object of study throughout this thesis. We will determine bounds on this value on the lattices $\mathbb{Z}^d$.

To do so we will first study some necessary prerequisites in Section 1. Next we introduce the contact process in Section 2. These first two sections give all the information you need to read the next three sections, which are largely independent of each other. In the third Section 3 we will do a short study of the contact process on finite graphs. The fourth section 4 provides bounds on the critical value of the contact process. Finally, the fifth Section 5 provides numerical estimates of this critical value.

I want to thank... finish this

1 Prerequisites

In this first chapter we will be introducing the prerequisites needed to read this thesis. As a baseline the reader should be familiar with undergraduate probability theory and real analysis. A small portion of this thesis benefits from knowledge about measure theory, but this is by no means necessary.

Throughout this thesis, we will be making connections to a variety of stochastic processes. To keep the text around my own results compact, I have collected all definitions and notation for these processes in this first section. To start, we need to establish what a stochastic process is. Any process that evolves randomly over time, is considered a stochastic process.

Definition 1.1. A stochastic process is a collection of random variables $\{X_t\}_{t\in A}$. The process is called discrete-time if A is a finite or countable set and continuous-time if A is uncountable.

In the remainder of this chapter we will introduce various stochastic processes that we will use to study the contact process, as well as some results from probability theory.

1.2 Results from probability theory

We will sporadically use results from probability theory that might require a refresher. All these results have been collected in this subsection. First we recall what a memoryless distribution is.

Definition 1.3. We say that a random variable X is memoryless if

$$\mathbb{P}[X > t + a \mid X > a] = \mathbb{P}[X > t].$$

Example 1.4. Suppose that $X \sim Exp(\lambda)$, then X is memoryless, since

$$\mathbb{P}[X > t + a \mid X > a] = \int_a^{t+a} \frac{\lambda e^{-\lambda x}}{e^{-\lambda a}}\, dx = \int_a^{t+a} \lambda e^{-\lambda(x-a)} dx = \int_0^t \lambda e^{-\lambda x} dx = \mathbb{P}[X > t].$$

One useful result is that every (continuous-time) memoryless distribution is exponentially distributed. As outlined in [6, p. 68], this is a direct consequence of the fact that the only real-valued function that satisfies $f(s + t) = f(s)f(t)$ is the exponential function.

Theorem 1.5. *If a continuous-time random variable X is memoryless then there exists a $\lambda \in \mathbb{R}$ such that $X \sim Exp(\lambda)$.*

We need one more result of exponential random variables. By the nature of the contact process, it is often useful to compare two exponentially distributed random variables. This motivates the following lemma.

Lemma 1.6. *If $X_1 \sim Exp(\lambda_1)$ and $X_2 \sim Exp(\lambda_2)$ then $\mathbb{P}[X_1 \geq X_2] = \dfrac{\lambda_1}{\lambda_1 + \lambda_2}$.*

One way to prove this entails integrating the joint distribution, which is the product of the individual distributions by independence, over all values where $X_1 \geq X_2$. For the complete proof the reader can refer to [9, p. 292]. Another elementary tool from probability theory that will prove useful is the union bound.

Lemma 1.7. *(Union bound) For events $A_1, \ldots, A_n$,*

$$\mathbb{P}\left[\bigcup_{i=1}^{n} A_i\right] \leq \sum_{i=1}^{n} \mathbb{P}[A_i].$$

Finally, we will define the probability generating function. We will need this later in this chapter to prove an important result.

Definition 1.8. The probability generating function $G_X(s)$ of an integer valued random variable X is

$$G_X(s) = \mathbb{E}[s^X].$$

By the choice of this function $G_X(s)$ we can find the probabilities that $X = i$ by differentiating the probability generating function

$$\mathbb{P}[X = n] = \frac{G_X^{(n)}(0)}{n!}.$$

The probability generating function has many more useful properties. One can read more about these in [4, p. 148].

1.9 The Galton-Watson branching process

One of the stochastic processes we will consider in this thesis is the Galton-Watson branching process.

Definition 1.10. The Galton-Watson process is a stochastic process $\{X_n\}_{n\in\mathbb{N}}$ with $X_n \colon \mathbb{N} \mapsto \mathbb{N}$ i.i.d. random variables. We maintain a list of nodes, starting with v_0. For each node in the list of nodes, the random variable X_n determines the number of children of the nth node, these children get added to the end of the list of nodes.

The nodes as generated by the Galton-Watson process described above generate a tree by drawing edges between the children of each node. This tree is called a Galton-Watson tree.

Example 1.11. An example of a Galton-Watson process with the discrete uniform offspring distribution $X_n \sim \mathit{Unif}\{1, 2, 3\}$ i.i.d. is illustrated in Figure 2. We start with v_0, and generate children according to X_0. This add two children, v_1 and v_2. We go down the list to the next node, v_1, which generates children its according to X_1. This continues either indefinitely, or until all nodes in the list have generated children.

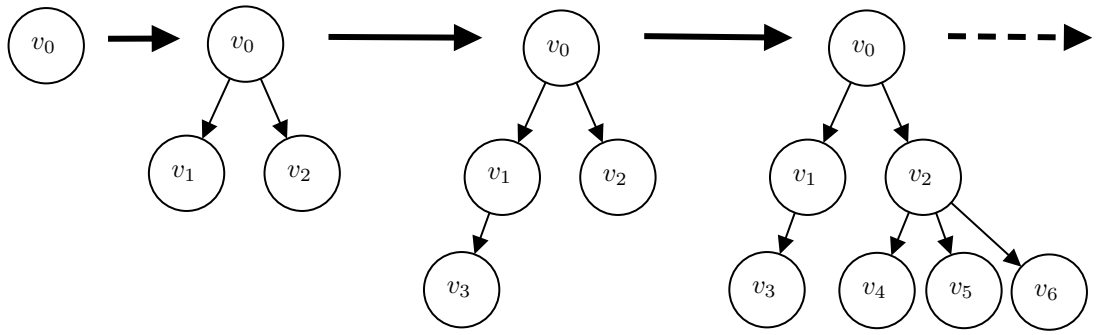


Figure 2: Example of a Galton-Watson process

Note that in this example, the process is bound to continue indefinitely, since we always generate at least one node. In this case we say that the process survives. Similarly, if we take X_n to be zero with probability 1, the process will always be finite, or, in other words, dies out. It turns out that there is a very general result about the survival of this process. To get to this results, we first define the distance between two edges.

Definition 1.12. If $G = (V_G, E_G)$ is a graph, and $e_1, e_2 \in E_G$ then the distance between e_1 and e_2 is the number of edges of the shortest path starting with e_1 and ending in e_2, minus 1. Similarly, the distance between $v_1, v_2 \in V_G$ is the number of nodes of the shortest path starting in v_1 and ending in v_2, minus 1. If there is no path between two edges or nodes, we say the distance is infinite.

When talking about Galton-Watson tree it makes sense to think in terms of generations. We will define a generation as the class of nodes that is a fixed distance away from the first node v_0.

Definition 1.13. The nth generation of the Galton-Watson tree, denoted by Z_n, is the set of nodes that are distance n away from initial node v_0.

Next, we want to know whether or not the process goes extinct, that is, whether or not there exists an $N \in Z_{\geq 0}$ such that $Z_N = 0$. This is neatly summarised in the following theorem.

Theorem 1.14. *For a branching process with i.i.d. offspring distribution X, denote the extinction probability by η. Then, $\eta = 1$ when $\mathbb{E}[X] \leq 1$ and $\mathbb{P}[X = 1] < 1$, while $\eta < 1$ if $\mathbb{E}[X] > 1$ or $\mathbb{P}[X = 1] = 1$. The extinction probability is the smallest solution in $[0, 1]$ of $\eta = G_X(\nu)$.*

We will follow the proof in [11, p. 86]. The number of children in a generation is some sum of copies of X. The probability distribution of such a sum is given by the convolution of copies of the probability distribution of X. It turns out that these convolutions are much easier to work with through their probability generating functions.

Proof. We start by showing that $\eta = G_X(\eta)$. We consider $\eta_n = \mathbb{P}[Z_n = 0]$, this is increasing in n and bounded by η so that $\eta_n \uparrow \eta$. Consider $G_n(s) = \mathbb{E}[s^{Z_n}]$, the generating function of the nodes in the nth generation. Then, $G_n(0) = \eta_n$.

We condition on the first generation, so that for $n \geq 2$,

$$G_n(s) = \mathbb{E}[s^{Z_n}] = \sum_{i=0}^{\infty} \mathbb{P}[X = i]\, \mathbb{E}[s^{Z_n} \mid Z_1 = i] = \sum_{i=0}^{\infty} \mathbb{P}[X = i] G_{n-1}(s)^i$$

Where in the last step we used the fact that all the i individuals after the first generation produce offspring for another $n - 1$ generations in an independent and identical way, so that

$$\mathbb{E}[s^{Z_n} \mid Z_1 = i] = \mathbb{E}\left[s^{\sum_{k=1}^{i} Z_{n-1}}\right] = G_{n-1}(s)^i.$$

We now write $G_X(s)$ for the generating function of X, which gives that

$$G_n(s) = G_X(G_{n-1}(s)).$$

Substituting $s = 0$, we obtain $\eta_n = G_X(\eta_{n-1})$. As $n \to \infty$ we get $\eta_n \uparrow \eta$ so that, by the continuity of $s \mapsto G_X(s)$,

$$\eta = G_X(\eta).$$

Next we show that η is the smallest solution of $s = G_X(s)$. Suppose that $\psi \in [0, 1]$ satisfies $\psi = G_X(\psi)$. We prove that $\eta_n \leq \psi$ for all n by induction. Clearly the base case, $\eta_0 = 0 \leq \psi$ holds. Next we assume that $\eta_n \leq \psi$. Since $G_X(s)$ in increasing on $[0, 1]$ this implies that

$$\eta_{n+1} = G_X(\eta_n) = G_X(\psi) = \psi.$$

Which completes the inductive step. Since $\eta_n \uparrow \eta$ we can conclude that $\eta \leq \psi$, meaning that η is the smallest solution of $s = G_X(s)$. To complete the proof we will now distinguish three cases.

Case 1: If $\mathbb{P}[X = 1] = 1$, then $\mathbb{P}[Z_n = 1]$ for all n, so its clear that $\eta = 0 < 1$. One can also note that $G_X(s) = s$ for all $s \in [0, 1]$ in this case.

Case 2: If $\mathbb{P}[X \leq 1] = 1$ and $p = \mathbb{P}[X = 0] > 0$, then $\mathbb{P}[Z_n = 0] = 1 - (1 - p)^n \to 1$, so that $\eta = 1$.

Case 3: If $\mathbb{P}[X \leq 1] < 1$, then we consider the second derivative of the generating function of X,

$$\mathbb{E}[X(X-1)s^{X-2}] \geq 0$$

This means that $G_X(s)$ is strictly increasing and strictly convex for $s > 0$. Consequently, there are at most two solutions of $s = G_X(s)$ in $[0, 1]$, one of which is necessarily $s = 1$, by definition of $G_X(s)$. Since $G_X(s)$ is strictly convex and $G_X(0) > 0$, we can find the number of solutions by looking at $G'_X(1)$. If $G'_X(1) < 1$ we have not yet passed through the line $f(s) = s$, and thus there is only one solution in $[0, 1]$ implying $\eta = 1$. Else if $G'_X(1) > 1$ we must have passed through the line $f(s) = s$, and thus there are two solutions in $[0, 1]$, showing that $\eta < 1$, which completes the proof.

□

The study of branching processes has many more interesting questions, but these are of not of relevance to this thesis. If you want to read more, you can refer to [11].

1.15 Percolation processes

Percolation models are stochastic processes which were devised to model the flow of a liquid through a porous medium. We will look at bond percolation, on a graph G. We write V_G and E_G for the sets of vertices and edges of the graph respectively. We will then decide whether or not the edges in this graph are "open".

Definition 1.16. The (ordinary) percolation process on a graph $G = (V_G, E_G)$ is a collection of Bernoulli random variables with parameter p given by $(X_e)_{e \in E_G}$.

The interpretation of Definition 1.16 is that we get a new graph with the same nodes as G and all the edges where $X_e = 1$. After we have done this, we ask if there exists an certain path through which water can flow. In infinite graphs, we look for an infinitely long path, that is, for each distance d, we can find two vertices in the path that are distance d' apart so that $d \leq d' < \infty$. We call this event "percolation". If we do this percolation process on an infinite graph G with an origin $\underline{0}$, we can define a so called critical probability.

Definition 1.17. The critical probability $p_c(G)$ for percolation on an infinite graph G with origin $\underline{0}$ is given by

$$p_c(G) = \inf\{\ p \mid \mathbb{P}_p\left[\underline{0} \leftrightsquigarrow \infty\right] > 0\}.$$

This probability is exactly the threshold for which an infinite path starting at the origin exists. It turns out that this probability is the same as the threshold for any infinite path to exist. We will not prove this here, but the proof is very similar to a later result, Lemma 2.16, on the critical value of the contact process.

One of the processes that we will be interested in oriented bond percolation on $\mathbb{Z}^2_{\geq 0}$. This is the same percolation process as in Definition 1.16, but with positively oriented edges. That is, horizontal edges go from (i, j) tot $(i + 1, j)$ and vertical edges go from (i, j) tot $(i, j + 1)$. A part of the oriented percolation process on $\mathbb{Z}^2$ is illustrated in Figure 3. A path to the edge of the picture has been marked in green. Percolation implies that there is a strictly positive probability that there exists a green path like this which is infinitely long.

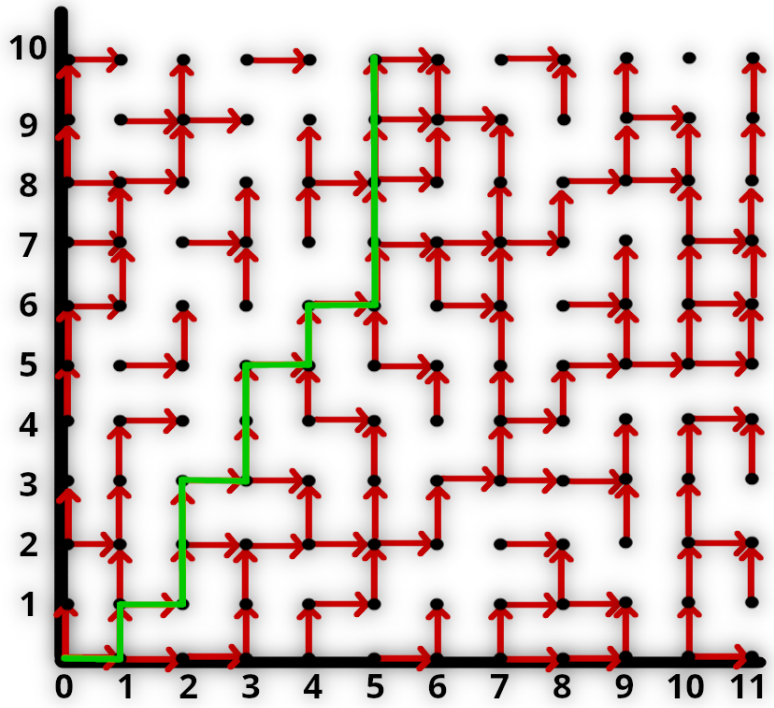


Figure 3: Path in oriented percolation

Not all percolation processes decide that edges are open with Bernoulli random variables. In this thesis we will require a special type of percolation. In particular, one where the event that an edge is open depends on the status of its neighbouring edges. This is called a n-independent percolation process.

Definition 1.18. An n-independent percolation process is a percolation process $(X_e)_{e \in E_G}$ such that the probability that X_{e_1} and X_{e_2} are independent if the distance between e_1 and e_2 is greater than n.

The study of percolation is one that is rich of interesting problems. However, for the purposes of this thesis, this short introduction suffices. If one wants to learn more about percolation, a good source is Geoffrey Grimmett's book [5].

2 Introduction to the contact process

In this section we introduce the subject of study in this thesis, the contact process. We start by discussing Markov processes in general. Next, we define the contact process and cover some important definitions and initial results. Finally, we will define couplings and discuss Harris' visual representation.

2.1 Markov processes

When modelling problems with stochastic processes we often deal with situations where the time you have to wait until the next event is memoryless. That is, regardless of how long you have been waiting, the probability of the event occurring within some fixed timespan remains constant. We can think of examples like the time until the next earthquake or until the failure of a machine.

This motivates the introduction of a special class of processes called Markov processes. In loose terms, a process is Markovian if the next state of the process does not depend on its history.

Definition 2.2. We say that a continuous-time stochastic process $\{X_t\}_{t\geq 0}$ with a countable state space S has the Markov property if

$$\mathbb{P}[X_t = x \mid X_r,\ 0 \leq r \leq s] = \mathbb{P}[X_t = x \mid X_s]$$

for $x \in S$ and $t \geq s \geq 0$.

In this thesis we will primarily consider Markov processes on graphs. Each node will have a state, and transitions to another state based on local rules. Another property we will come across is called time-homogeneity. In essence, time-homogeneity means that the transition probabilities do not depend on time.

Definition 2.3. We say that a continuous-time stochastic process $\{X_t\}_{t\geq 0}$ with state space S is time-homogenous if

$$\mathbb{P}[X_t = x \mid X_s = y] = \mathbb{P}[X_{t-s} = x \mid X_0 = y]$$

for $x, y \in S$ and $t \geq s \geq 0$..

One example of a Markov process is a continuous time Markov chain. This is a process with a discrete state space, where we describe the transitions from one state to the next. To ensure the Markov property, the transitions must be exponentially distributed due to Theorem 1.5. Hence, it suffices to give the parameter for each of the exponential distributions, which we call the transition rates.

Definition 2.4. A continuous-time Markov chain is a continuous-time Markov process $(X_t)_{t\geq 0}$ with $X_t \in V$. It has a countable state space S with transitions rates $(s_{i,j})_{i,j\in S}$ corresponding to the rate at which X_t is state v_i transitions to v_j.

The prime example of a Markov process is the Poisson process. This process is discussed in great detail in many texts on undergraduate probability, such as [6, p. 65] and [9, p. 292]. For completeness, one description of the Poisson process is included below.

Example 2.5. We look at a series of events where the times between occurrences are distributed by $T_i \sim Exp(\lambda)$. Let $N_t = \max\left[n : \sum_{i=0}^{n} T_i \leq t\right]$, this is what we call the Poisson process. We

claim that this is a Markov process, which is also time-homogenous. First recall from Example 1.5 that the transition times are memoryless. Using this, it is evident that this is a time-homogenous Markov process. The only thing that impacts the future probabilities is how many events there have been so far. It does not matter what time the previous event occurred at. The Poisson process can be thought of as a continuous-time Markov chain with state space $\mathbb{N}$ and transition rates $n \to n+1$ equal to λ.

The fact that the Poisson process has the Markov property and is homogenous in time will be used many times in this thesis.

2.6 The contact process

We start with the definition of the contact process. As explained in the introduction of this thesis, this is a model for the spread of an infection. We will keep track of a set of infected nodes, ξ_t, and specify the rates at which healthy nodes become infected and vice versa.

Definition 2.7. The contact process is a continuous-time Markov Process on the subsets of nodes of an undirected graph $G = (V, E)$ with a parameter λ, where the transitions are determined by

$$\begin{aligned} &\text{for } x \in \xi_t, \quad \xi_t \mapsto \xi_t \setminus \{x\} \text{ with rate } 1, \\ &\text{for } x \notin \xi_t, \quad \xi_t \mapsto \xi_t \cup \{x\} \text{ with rate } \lambda |\{y \in \xi_t \colon \{x, y\} \in E\}|. \end{aligned} \tag{2.1}$$

To fully define this process, we need to specify an initial occupancy. That is, we need to specify which nodes are infected at $t = 0$. To specify a contact process with initial occupancy $A \subset V$ we use $\{\xi_t^A\}_{t \geq 0}$.

Throughout this thesis we will be studying the contact process on the square lattice on $\mathbb{Z}^d$. This is a graph with nodes in $\mathbb{Z}^d$, and edges between any pair of points that has distance exactly 1 (as defined in Definition 1.12). We refer to this graph simply by $\mathbb{Z}^d$. We will concern ourselves with the so-called critical infection rate $\lambda_c(\mathbb{Z}^d)$. This is the value of λ at which there occurs a fundamental change in the limiting behaviour of the stochastic process, analogous to critical probabilities in the study of random graphs. For the contact process this means it is the infinum over all the rates for which the set of infected nodes is eventually empty.

Definition 2.8. The critical infection rate on $\mathbb{Z}^d$, is defined by

$$\lambda_c(\mathbb{Z}^d) = \inf \left\{ \lambda : \lim_{t \to \infty} \xi_t^{\{0\}} = \varnothing \right\}. \tag{2.2}$$

Often we will just write λ_c if its clear which graph it concerns. Later we will proof in Lemma 2.16 that instead of initial occupancy $\{0\}$, we can take any finite selection of nodes. The critical value can be defined for a much broader class of graphs, however, to do this rigorously is beyond the scope of this thesis, so we stick with this simpler alternative.

When talking about the contact process we distinguish the so called subcritical and supercritical regions. These are the values of λ such $\lambda < \lambda_c(G)$ and $\lambda > \lambda_c(G)$ respectively. Qualitatively, when looking a finite portion of the process, we expect the share of infected sites to quickly tend towards 0 in the subcritical case, whereas in the supercritical case we expect the infection to converge to a certain non-zero (probabilistic) equilibrium. This illustrated below in Figure 4, which illustrates the expected behaviour of a portion of the graph $\mathbb{Z}$ with 100 nodes where all nodes are infected at the start.

Let us now consider finite graphs, in this case the infection always dies out, so the critical value is trivially 0. Hence, it makes sense to look at another metric, namely the extinction time, which is defined as follows.

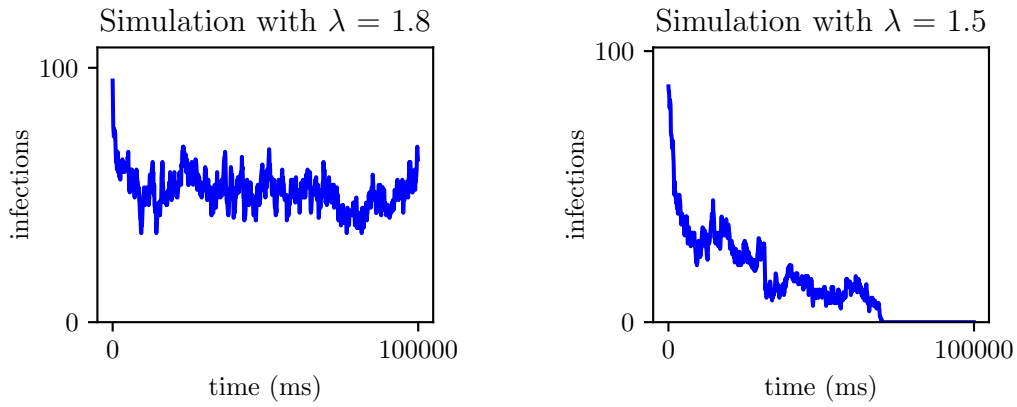


Figure 4: The contact process in the sub- and supercritical case

Definition 2.9. The extinction time of the contact process of a graph G with initial occupancy $A \subset G$ is given by

$$\tau_G^A = \inf\{t : \xi_t^A = \varnothing\}. \tag{2.3}$$

In Section 3 we will make it plausible that a similar notion of sub- and supercritical exists for finite graphs. For the contact process we also define what is called the survival function. This function gives the probability of survival, given the parameter of the contact process.

Definition 2.10. The survival function $\psi : \mathbb{R}_{\geq 0} \mapsto [0, 1]$ maps a parameter λ of the contact process $\xi_t^{\{0\}}$ on $\mathbb{Z}^d$ to $\mathbb{P}\left[\lim_{t\to\infty} \xi_t^{\{0\}} = \varnothing\right]$.

This survival function is often studied in literature regarding the contact process. In the final chapter we will develop methods for estimating the critical value, which also allow us to estimate this function. In Figure 5 the approximate shape of ψ is visualised.

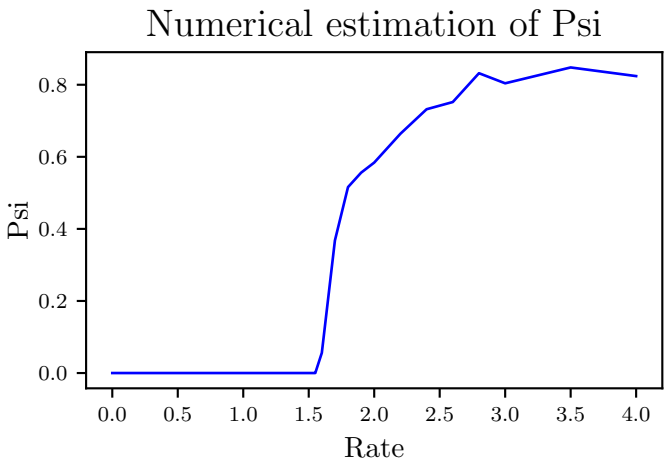


Figure 5: Numerical estimation of psi

2.11 Couplings

In this thesis we will greatly rely on the following two definitions. In order to study the contact process, it is often useful to make a comparison with a simpler process. We will often do so using a technique called coupling.

Definition 2.12. A coupling between two random variables $X_1 : \Omega_1 \mapsto \mathbb{R}$ and $X_2 : \Omega_2 \mapsto \mathbb{R}$ is a new probability space Ω' with two random variables $X_1' : \Omega' \mapsto \mathbb{R}$ and $X_2' : \Omega' \mapsto \mathbb{R}$ such that X_1' and X_2' are identically distributed to X_1 and X_2 respectively.

Most comparisons we will want to make are where one process dominates the other, that is, if an event happen in one process with a certain probability p, it also happens in the other with at least probability p. This motivates the following definition.

Definition 2.13. A random variable $X : \Omega \mapsto \mathbb{R}$ is said to (stochastically) dominate $Y : \Omega \mapsto \mathbb{R}$ if

$$\mathbb{P}[X \geq x] \geq \mathbb{P}[Y \geq x], \text{ for all } x \in \mathbb{R}.$$

In this case, we write $X \geq_{st} Y$. If we have that

$$\mathbb{P}[X \geq x] > \mathbb{P}[Y \geq x], \text{ for all } x \in \mathbb{R},$$

we say that X strictly (stochastically) dominates Y, and we write $X >_{st} Y$

This in itself is not yet sufficient, as we will often we dealing with random variables that do not necessarily lie in the same probability space. The following result, shows that it is sufficient to find a suitable coupling.

Lemma 2.14. *A random variable $X : \Omega_1 \mapsto \mathbb{R}$ is said to stochastically dominate $Y : \Omega_2 \mapsto \mathbb{R}$ if and only if there exists a coupling (X', Y') on Ω' such that $\mathbb{P}[X' \geq Y'] = 1$.*

For the proof the reader can refer to [11, p. 64]. It might not immediately obvious why couplings are even worth studying, since we are just making copies of the random variables we already had. The trick will be to make these copies dependent, so that knowledge about one random variable tells us something about the other. Let us look at a simple example.

Example 2.15. Consider $X_1 \sim B(n, p)$ and $X_2 \sim B(m, p')$ with $n \geq m$ and $p \geq p'$. We will show that $X_1 \geq_{st} X_2$ through a suitable coupling. We will take samples $x_0, \ldots, x_{n-1}$ from $U(0, 1)$ and define

$$(X_1', X_2') = \left(\sum_{i=0}^{n} \mathbb{1}_{[0,p]}(x_i), \sum_{i=0}^{m} \mathbb{1}_{[0,p']}(x_i) \right).$$

Then we have that X_1 and X_2 are identically distributed to X_1' and X_2' since we are counting the number of successful Bernoulli experiments, and $\mathbb{P}[X_1' \geq X_2'] = 1$ since any success in X_2 is also a success in X_1. This shows $X_1 \geq_{st} X_2$ with Lemma 2.14.

Example 2.15 illustrates the power of couplings. Proving this inequality directly would be very cumbersome and computational. One very important result on the contact process is due to Harris. There exists a coupling between the contact process and a carefully selected set of Poisson processes. For each node in the graph we assign a rate 1 Poisson process, and a rate λ Poisson process for each outgoing edge. The first process determines the time when a site becomes healthy. The second processes determine when the process infects the corresponding neighbour, if they're infected themselves. For the contact process on $\mathbb{Z}$, an illustration can be found in Figure 6. In Figure 6 the blue crosses represent recoveries, and red arrows corresponds

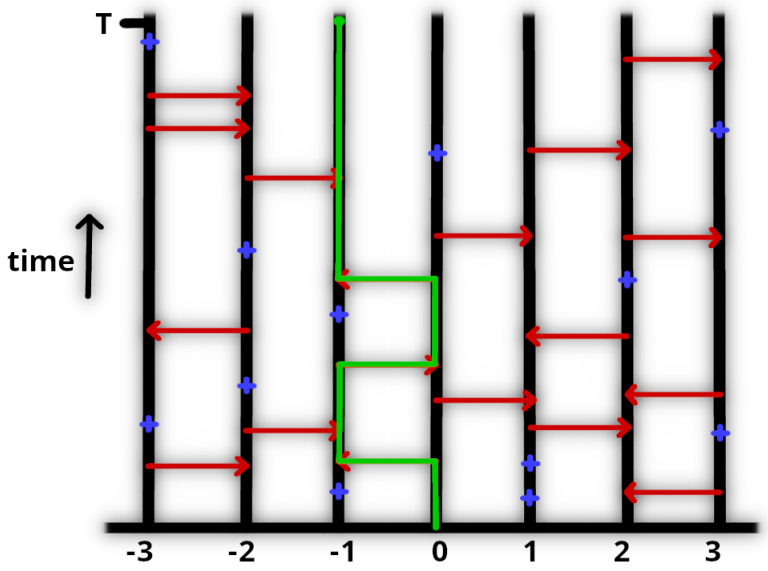


Figure 6: Harris' visual representation

to infections. In the figure we can see that the node -1 is infected at time T if node 0 is infected at time 0, since we draw a path from 0 at time 0 to -1 at time T without any recoveries. We will mimic notation from percolation, writing $v_1 \rightsquigarrow \infty$ if there is an infinitely long infection path in time starting in v_1 at time 0. By the coupling with Poisson processes it should be clear from Example 2.5 that the contact process is homogenous in time and has the Markov property.

Let us now return to the question after Definition 2.8. We can define the critical value as $\inf\left\{\lambda : \lim_{t\to\infty} \xi_t^A = \varnothing\right\}$ for any finite initial configuration A, by the following lemma.

Lemma 2.16. *Considering a contact process $(\xi_t)_{t\geq 0}$ on a graph G. Then, the following are equivalent for any finite $A \subset \mathbb{Z}^d$,*

1. $\lambda > \lambda_c$,
2. $\psi(\lambda) > 0$,
3. $\lambda > \inf\left\{\lambda : \lim_{t\to\infty} \xi_t^A = \varnothing\right\}$.

Proof. It follows directly from the definition of the critical value that (1) and (2) are equivalent. The implication (2) $\implies$ (3) is trivial since $\{0\}$ is a possible choice of A. For the final implication (3) $\implies$ (2), let us assume by contraposition that $\psi(\lambda) = 0$. Let Ξ be one realisation of Harris' visual representation. Then

$$\begin{aligned}\mathbb{P}\left[\lim_{t\to\infty} \xi_t^A = \varnothing\right] &= \mathbb{P}\left[A \not\rightsquigarrow \infty \text{ in } \Xi\right] \\ &\leq \sum_{i\in A} \mathbb{P}\left[i \not\rightsquigarrow \infty \text{ in } \Xi\right] \\ &= \sum_{i\in A} \mathbb{P}\left[\lim_{t\to\infty} \xi_t^{\{i\}} \neq \varnothing\right] = 0\end{aligned}$$

using the union bound (Lemma 1.7). This shows that $\lambda \leq \inf\left\{\lambda : \lim_{t\to\infty} \xi_t^A = \varnothing\right\}$, completing the proof. □

One might wonder if we can prove any strict (stochastic) inequalities on the extinction time of more general classes of graphs. So to finish this we will first show a (non-strict) inequality for graph that are included in one another by defining a coupling between these two graphs.

Theorem 2.17. *If $G_1 \subset G_2$ are graphs, $A \subset G_2$ an initial occupancy and $A' = A \cap V_{G_1}$.*

- *If G_1 and G_2 are finite, then $\tau_{G_1}^{A'} \leq_{st} \tau_{G_2}^{A}$*
- *If G_1 and G_2 are infinite, then $\lambda_c(G_1) \leq \lambda_c(G_2)$.*

Proof. We prove this statement by defining a suitable coupling. Let $(\xi_t^A)_{t\geq 0}$ be a contact process on G_1 and $(\zeta_t^A)_{t\geq 0}$ a contact process on G_2. At time $t = 0$ we will have that $\xi_0^{A'} \subset \zeta_0^A$. We couple in a way that preserves this inclusion. For sets A and B we will write

$$r(A, B) := \lambda|\{y \in A\colon \{x, y\} \in B\}|.$$

This is the rate of infection from infected points in A through edges in B. We will now couple in the, so that the contact processes coincide where G_1 and G_2 coincide.

For $x \in \zeta_t$,	$(\xi_t, \zeta_t) \mapsto (\xi_t \setminus \{x\}, \zeta_t \setminus \{x\})$ *with rate* 1,
for $x \in \xi_t \setminus \zeta_t$,	$(\xi_t, \zeta_t) \mapsto (\xi_t \setminus \{x\}, \zeta_t)$ *with rate* 1,
for $x \in \xi_t$ and $x \notin \zeta_t$,	$(\xi_t, \zeta_t) \mapsto (\xi_t, \zeta_t \cup \{x\})$ *with rate* $r(\zeta_t, E_{G_1})$
for $x \notin \xi_t \cap V_{G_1} \cap V_{G_2}$,	$(\xi_t, \zeta_t) \mapsto (\xi_t \cup \{x\}, \zeta_t \cup \{x\})$ *with rate* $r(\xi_t \setminus \zeta_t, E_{G_1})$
for $x \notin \xi_t \cap (V_{G_2} \setminus V_{G_1})$,	$(\xi_t, \zeta_t) \mapsto (\xi_t \cup \{x\}, \zeta_t)$ *with rate* $r(\xi_t \setminus \zeta_t, E_{G_2} \setminus E_{G_1})$.

This has been carefully constructed to exactly give us our two contact processes, whilst ensuring $\xi_t^{A'} \subset \zeta_t^A$ for all $t \geq 0$.

As a consequence of this coupling, it holds that $\mathbb{P}[|\xi_t| \geq |\zeta_t|] = 1$. This means that if the infection survives in G_1, we then immediately know that it must also survive in G_2, allowing us to apply Lemma 2.14 to conclude that the indicator of the event of survival in G_2 dominates that of G_1. Consequently, we get that $\tau_{G_1}^A \leq_{st} \tau_{G_2}^A$ for all non-empty initial occupancies A in the finite case and hence that $\lambda_c(G_2) \leq \lambda_c(G_1)$ in the infinite case. □

3 The contact process on finite graphs

In this first section we will look at the contact process on finite graphs. In particular, we will derive a way to find a crude upper and lower bound on the expected extinction time of the contact process on n-cyclic graphs. Additionally, we will take a look at general class of finite graphs, and prove strict inequalities on extinction time, for graphs that are strictly included in a another graph.

3.1 Expected extinction time on cyclic graphs

In this section we will derive a crude upper and lower bound on the expected extinction time of the contact process on n-cyclic graphs. We will do so by considering the probabilities that a certain number of nodes are infected at a set time. These probabilities are very hard to determine, but we are able to find an upper and lower bound. Let us first derive a few specific results on exponentially distributed random variables.

Lemma 3.2. *If X_1 and X_2 are independently exponentially distributed with parameters λ_1 and λ_2 respectively, then* $\min(X_1, X_2) \sim Exp(\lambda_1 + \lambda_2)$.

Proof. Since X_1 and X_2 are independent,

$$\mathbb{P}[\min(X_1, X_2) > a] = \mathbb{P}[X_1 > a, X_2 > a] = \mathbb{P}[X_1 > a]\,\mathbb{P}[X_2 > a] = e^{-\lambda_1 a - \lambda_2 a}.$$

Hence $\mathbb{P}[\min(X_1, X_2) \leq a] = 1 - e^{-\lambda_1 a - \lambda_2 a}$, and since this is precisely the cumulative distribution function of $Exp(X_1 + X_2)$, it must hold that $\min(X_1, X_2) \sim Exp(\lambda_1 + \lambda_2)$. □

This now allows us to prove the following lemma, which will be useful for deriving the main result of this section.

Lemma 3.3. *Given that $X_1 \sim Exp(\lambda_1)$ and $X_2 \sim Exp(\lambda_2)$ independently*

$$\mathbb{E}[X_1 \mid X_1 < X_2] = \tfrac{1}{\lambda_1 + \lambda_2}.$$

Proof. We start by noting that the X_1 is equal to $\min(X_1, X_2)$ under the condition that $X_1 < X_2$, hence

$$\begin{aligned}\mathbb{E}[X_1 \mid X_1 < X_2] &= \mathbb{E}[\min(X_1, X_2)] \\ &= \frac{1}{\lambda_1 + \lambda_2}\end{aligned}$$

Using that $\min(X_1, X_2) \sim Exp(\lambda_1 + \lambda_2)$ as proven in Lemma 3.2. □

Now that we have this tool we can formulate and prove the main results of this section. We will determine the lower and upper bound by comparing with a suitable continuous time Markov chain. Let us first determine a lower bound.

Theorem 3.4. *Let $\{\xi_t^A\}_{t \geq 0}$ denote the contact process on the n-cyclic graph $G = C_n$. Then a lower bound X_i of $\mathbb{E}\left[\tau_G^A \mid |A| = i\right]$ can be found for all $1 \leq i \leq n$ by solving the system*

$$\begin{cases} X_1 &= \frac{1}{(2\lambda+1)^2} + \frac{2\lambda}{2\lambda+1}\left(X_2 + \frac{1}{2\lambda+1}\right), \\ X_i &= \frac{i}{2\lambda+i}\left(X_{i-1} + \frac{1}{2\lambda+i}\right) + \frac{2\lambda}{2\lambda+i}\left(X_{i+1} + \frac{1}{2\lambda+i}\right),\ 2 \leq i \leq n-1, \\ X_n &= X_{n-1} + \frac{1}{n}. \end{cases}$$

Proof. We want to find a lower bound on the number of infections. We establish an order on the edges and on the nodes of C_n, denoted by $\geq_V$ and $\geq_E$ respectively. Observe the total rate of new infections across the whole graph C_n, is at least 2λ if there is an healthy site. We will couple a continuous Markov chain in Figure 7 to the contact process on C_n so that if the contact process has gone extinct, the chain has as well. Denote the Markov chain by M_t. For every $t \geq 0$, let x_t^m be the first m infected nodes in $\geq_V$ and E_t^{min} the two smallest edges in $\{y \in \xi_t \colon \{x, y\} \in E_{C_n}\}$ according to $\geq_E$ at time t. Additionally, for sets A and B we will write

$$r(A, B) := \lambda |\{y \in A \colon \{x, y\} \in B\}|.$$

This is the rate of infection from infected points in A through edges in B. The coupling is defined as follows.

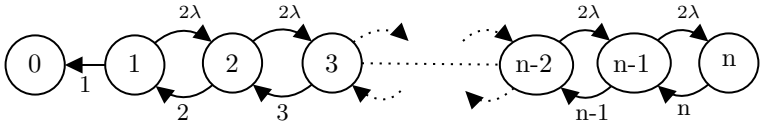


Figure 7: Lower bound Markov chain

$$\textit{For } x \in \xi_t \textit{ and } m = M_t, \begin{cases} (\xi_t, M_t) \mapsto (\xi_t \setminus \{x\}, m-1) \textit{ with rate } 1, \textit{if } \xi_t \in x_t^m, \\ (\xi_t, M_t) \mapsto (\xi_t \setminus \{x\}, m) \textit{ with rate } 1, \textit{ else.} \end{cases}$$

$$\textit{For } x \notin \xi_t \textit{ and } m = M_t, \begin{cases} (\xi_t, M_t) \mapsto (\xi_t \cup \{x\}, m+1) \textit{ with rate } r\left(\xi_t, E_t^{min}\right), \\ (\xi_t, M_t) \mapsto (\xi_t \cup \{x\}, m) \textit{ with rate } r\left(\xi_t, E_{C_n} \setminus E_t^{min}\right). \end{cases}$$

Any time step from i to $i-1$ in the Markov chain corresponds to a recovery in one of the first i infected nodes in the contact process. A step from i to $i+1$ in the Markov chain corresponds to an infection through one of the first two edges between a healthy and an infected site. Any other events in the contact process do not affect the Markov chain. By construction, the state of this Markov chain is certainly smaller or equal to the number of infections in the contact process, so the time until this Markov reaches 0 starting at $|A|$ is stochastically dominated by τ_G^A.

Now, we can derive a system of equations based on this Markov chain. Let $X \sim \exp(2\lambda)$ and $Y_i \sim \exp(i)$, then given that we are in state $2 \leq i \leq n-1$ we can find the expected time until extinction based on the neighbouring states. We let X_i be the time that this Markov chain goes extinct if we start in state i, then the expected value of X_i satisfies

$$\begin{cases} X_1 &= \mathbb{P}[Y_1 < X]\,\mathbb{E}[Y_1 \mid Y_1 < X] + \mathbb{P}[X < Y_1]\,(X_2 + \mathbb{E}[X \mid X < Y_1]), \\ X_i &= \mathbb{P}[Y_i < X]\,(X_{i-1} + \mathbb{E}[Y_i \mid Y_i < X]) + \\ & \quad \mathbb{P}[X < Y_i]\,(X_{i+1} + \mathbb{E}[X \mid X < Y_i]),\ 2 \leq i \leq n-1 \\ X_n &= X_{n-1} + \mathbb{E}[Y_n]. \end{cases}$$

By Lemma 1.6 we get $\mathbb{P}[Y_i < X] = \dfrac{i}{2\lambda + i}$ and $\mathbb{P}[X < Y_i] = \dfrac{2\lambda}{2\lambda + i}$ and by Lemma 3.3 we find $\mathbb{E}[X \mid X < Y_i] = \mathbb{E}[Y_i \mid Y_i < X] = \dfrac{1}{2\lambda + i}$ to get the desired result. □

Now, to find an upper bound we will analogously determine a Markov chain that gives an upper bound on the number of infections.

Theorem 3.5. *Let $\{\xi_t^A\}_{t\geq 0}$ denote the contact process be a contact process on the n-cyclic graph $G = C_n$. Then a lower bound X_i of $\mathbb{E}\left[\tau_G^A \mid |A| = i\right]$ can be found for all $1 \leq i \leq n$ by solving the system*

$$\begin{cases} X_1 &= \frac{1}{(2\lambda+1)^2} + \frac{2\lambda}{2\lambda+1}\left(X_2 + \frac{1}{2\lambda+1}\right), \\ X_i &= \frac{i}{2i\lambda+i}\left(X_{i-1} + \frac{1}{2i\lambda+i}\right) + \frac{2i\lambda}{2i\lambda+i}\left(X_{i+1} + \frac{1}{2i\lambda+i}\right),\ 2 \leq i \leq \left\lfloor \frac{n-1}{2} \right\rfloor, \\ X_i &= \frac{i}{2(n-i-1)\lambda+i}\left(X_{i-1} + \frac{1}{2(n-i-1)\lambda+i}\right) + \\ & \quad \frac{2(n-i-1)\lambda}{2(n-i-1)\lambda+i}\left(X_{i+1} + \frac{1}{2(n-i-1)\lambda+i}\right),\ \left\lfloor \frac{n-1}{2} \right\rfloor < i \leq n-1, \\ X_n &= X_{n-1} + \frac{1}{n}. \end{cases}$$

Proof. We no consider a Markov chain which follows the upper bounds on the infection rates. The infection rate is maximal when there are as many healthy sites bordering infectious ones as possible. While the number of infections is strictly smaller than $\frac{n}{2}$, this means we have an infection rate of at most 2λ times the number of infected sites. Once the number of infections is larger or equal to $\frac{n}{2}$ we get a rate of at most 2λ times the number of healthy sites. We will use this fact to couple to the continuous-time Markov chain in Figure 8. Using the same notation as in Theorem 3.4, we will couple as follows.

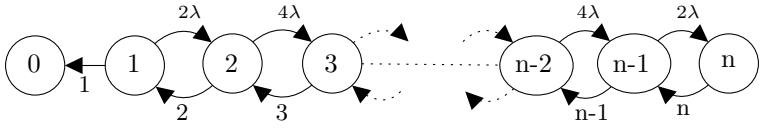


Figure 8: Upper bound Markov chain

For $x \in \xi_t$ and $m = M_t, (\xi_t, M_t) \mapsto (\xi_t \setminus \{x\}, m-1)$ with rate 1.

$$\textit{For } x \notin \xi_t \textit{ and } m = M_t, \begin{cases} (\xi_t, M_t) \mapsto (\xi_t \cup \{x\}, m+1) \textit{ with rate } r\left(\xi_t, E_{C_n}\right), \\ (\xi_t, M_t) \mapsto (\xi_t, m+1) \textit{ with rate } 2\lambda \min\left(n-m, m\right) - r\left(\xi_t, E_{C_n}\right). \end{cases}$$

Through this coupling we ensure that every recovery means we take a step from i to $i-1$ in the Markov chain, and for every infection we also take a step from i to $i+1$ in the chain. In addition to this we will also take steps from i to $i+1$ so that the total rate is always equal to the maximal rate we will ever have in the normal contact process.

By construction, the state of this Markov chain is certainly greater or equal to the number of infections in the contact process, so the time until this Markov reaches 0 starting at $|A|$ stochastically dominates τ_G^A. The rest of this proof of this theorem is analogous to that of Theorem 3.4, by again deriving a system of equations from the Markov chain.

□

For small values of n, you can write computer programs that are capable of solving these systems algebraically. We have done so to visualise the quality of these bounds, the scripts utilised can be found in Appendix A.1. If we specifically look at the extinction time given full occupancy, we get something that looks like Figure 9. We can see that as n increases, the difference between our upper and lower bound increases drastically. This pattern holds across all initial configurations, but gets slightly less drastic.

With the aid of the numeric solver an attempt has been made to derive and explicit solution for the systems in this chapter, but this was to no avail. We have seen in Figure 4, there is a

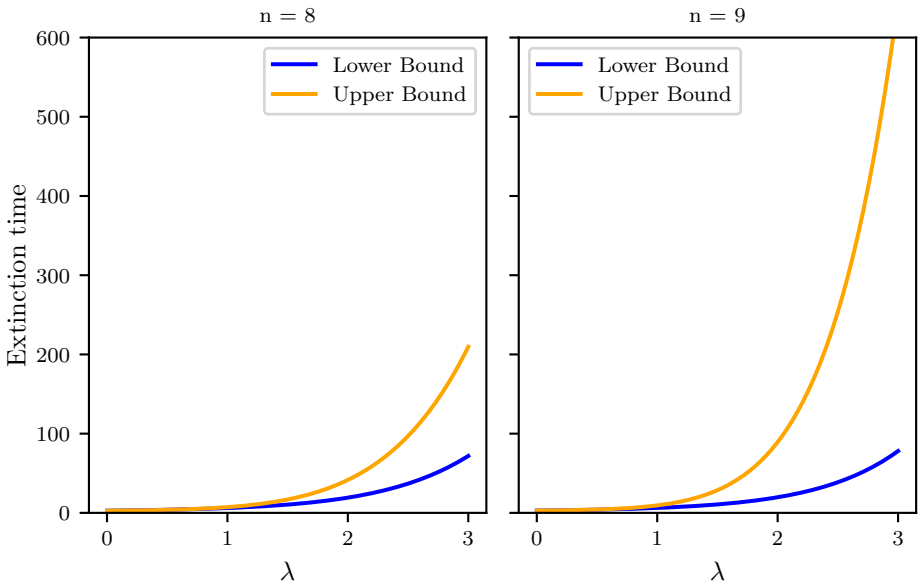


Figure 9: Upper and lower bound for n = 8 and n = 9

change in behaviour in infinite graphs around the critical value. Heuristically, one might think a similar change would occur in finite graphs. To investigate this, we will look at solutions of the system for fixed values of λ as n grows.

For most rates, the upper bound seemingly grows exponentially, whilst the lower bound appears to grow logarithmically. We can see this illustrated in Figure 10a where we plotted the bounds for rate 1. In this case, it is unclear how the expected extinction time behaves as n grows. However, if we now look at the graph with rate 0.25 in Figure 10b up until $n = 40$, we will see that both the upper and lower bound appear to grow logarithmically. Making the bound relatively tight for these smaller values of λ.

Additionally, if we take the rate greater than $\frac{n}{2}$, we see exponential growth until n in the lower bound. This is illustrated in Figure 10c, for rate 9 up until $n = 19$, to stay within the floating point range. This hints at the fact that a change in behaviour also occurs in these finite graphs. However, since the change does not occur in the lower bound for a fixed value of λ we cannot definitively say that it does in fact occur.

3.6 Strict inequality on extinction time

One might wonder if we can find any strict inequalities on extinction time. Given the inequality from Theorem 2.17, it suffices to show that the random variables $\tau^A_{G_1}$ and $\tau^A_{G_2}$ are different, for all A and $G_1 \subsetneq G_2$. To do so, we will identify a strictly positive probability, that G_1 goes extinct, while the infection is G_2 still non-extinct.

Theorem 3.7. *If $G_1 \subsetneq G_2$ are finite connected graphs, then $\tau^A_{G_1} <_{st} \tau^A_{G_2}$ for all non-empty initial occupancies.*

Proof. Since we have a finite graph, the total rate of infections and recoveries is always smaller or equal to $r = 2\lambda|E_{G_2}| + |V_{G_2}|$ where $|E_{G_2}|$ and $|V_{G_2}|$ denote the number of edges and vertices

of G_2 respectively. Throughout this proof, we utilise the same coupling as used in Theorem 2.17. Additionally, because of this same theorem, it suffices to prove $\tau_{G_1} \neq \tau_{G_2}$.

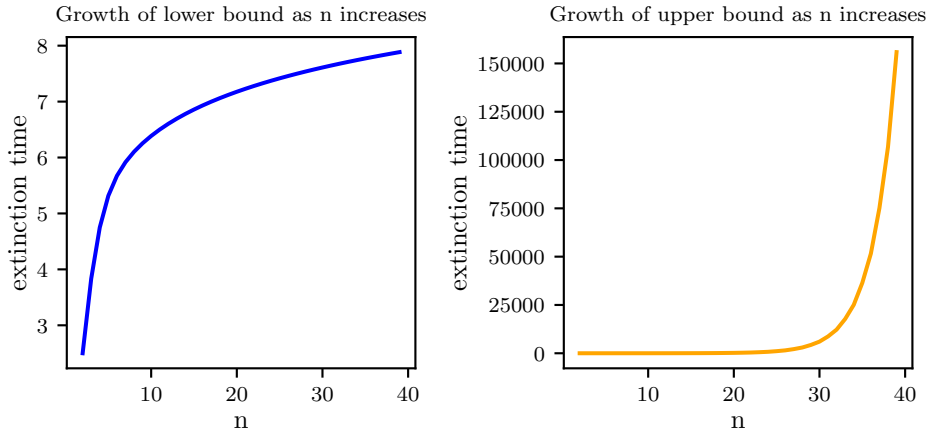
We first show that, without loss of generality, we can assume that the initial occupancy is full occupancy. We do this by showing that there is a positive probability to reach full occupancy, after which $\tau_{G_1} \neq \tau_{G_2}$ implies $\tau^A_{G_1} \neq \tau^A_{G_2}$ by the Markov property of the contact process. Connectedness of the graph implies that there is always a probability of infection greater or equal to $\frac{\lambda}{r}$ if there are uninfected sites. Let E denote the event that $(\xi^A_t)_{t\geq 0}$ reaches full occupancy starting from $t = 0$, then

$$\mathbb{P}[E] \geq \prod_{i=1}^{|V_{G_2}\setminus A|} \frac{\lambda}{r} > 0.$$

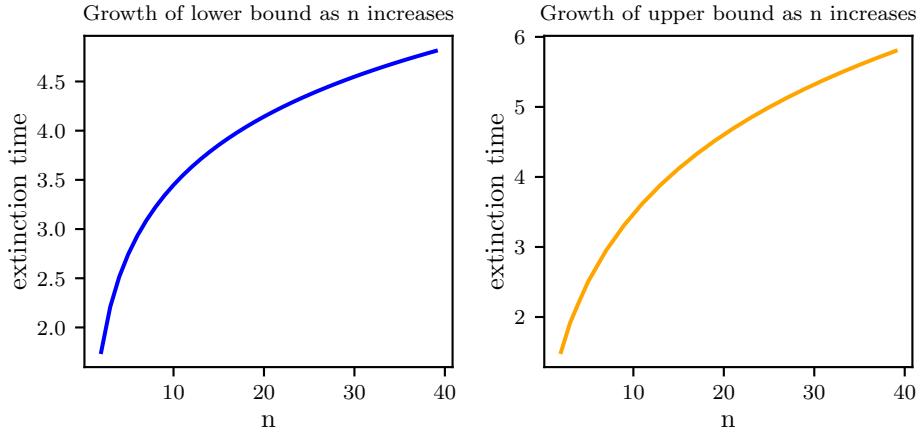
Now assuming full occupancy, we will use the same trick to show that there is a strictly positive probability that all sites of G_1 recover at at time smaller than T, whilst none of $G_2 \setminus G_1$ do, all the while there are no infections. The probability that a node recovers before any new infections is therefore greater or equal to $\frac{|\xi_t|}{r}$ by Lemma 1.6. This gives

$$\mathbb{P}[\tau_{G_1} < \tau_{G_2}] \geq \prod_{i=1}^{|V_{G_1}|} \frac{i}{r} > 0,$$

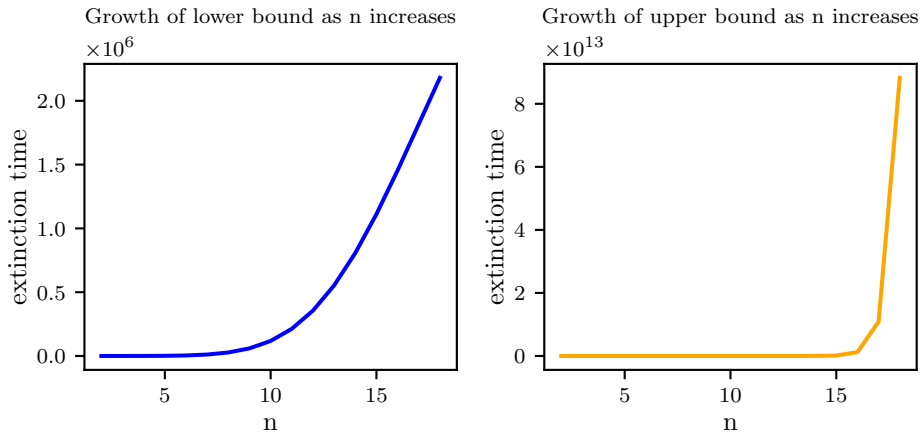
proving that $\tau_{G_1} \neq \tau_{G_2}$, completing the proof. □



(a) Growth of upper and lower bound for rate 1



(b) Growth of upper and lower bound for rate 0.25



(c) Growth of upper and lower bound for rate 9

Figure 10: Growth of upper and lower bounds for various rates

4 Upper and lower bounds on the critical value

In this section we will derive upper and lower bounds for $\lambda_c(\mathbb{Z}^d)$. We will do so by making a comparison to the Galton-Watson branching process to establish a lower bound in Subsection 4.1. Next, in Subsection 4.8, we find an upper bound by comparing contact processes in different dimensions in a specific way. Finally we determine an upper bound for the critical value of the contact process in dimension 1.

4.1 Lower bound

We can obtain an asymptotic lower bound by making a coupling to Galton-Watson trees. We will will explore the contact process until a certain number of the changes have occurred. To do so we define frontier events.

Definition 4.2. We say there is a frontier event at time t in the contact process anytime there is a recovery of an infected node, or an infection of an healthy node.

In order to couple with Galton-Watson trees we need to ensure the independence of the child distributions. We can use the time-homogeneity and the Markov property of the contact process to couple to a Galton-Watson tree. We will let the child distribution be the children after k frontier events. Doing this gives us the following lemma.

Lemma 4.3. *The contact process $\{\xi_t\}_{t\geq 0}^{\{0\}}$ on $\mathbb{Z}^d$ dies out if the expected number of infected sites after the first $k \in \mathbb{Z}_{\geq 1}$ frontier events is less or equal to* 1.

Proof. We will show that we can generate a Galton-Watson tree from the contact process $\{\xi_t\}_{t\geq 0}^{\{0\}}$ on $\mathbb{Z}^d$ such that extinction of the tree, implies extinction of the contact process. The Galton-Watson process we will construct will have the following child distribution X,

$$X := \textit{the total number of infected site after the first } k \textit{ frontier events.}$$

Let 0 correspond to the first node of the contact process, we will consider Harris' visual representation. We consider the first frontier event that can be reached from 0. If this is a recovery we are done. If it is an infection, we now look for the next frontier event that can be reached from the two infected sites. For this next event we again distinguish between recovery and infection. In the case of recovery, we continue considering with the one infected site left. If it is an infection, we continue with all the sites that have been infected. We repeat this until we have seen k frontier events. Doing this gives the children for the first node given by X. For the next step, it is important to remember which nodes have been visited at what times.

Next, for each of the children of the first node. We do the same exploration. However now, anytime we come across a node that we have already visited, this node will continue generating Poisson events for infections and recoveries i.i.d. to the contact process. Any infections caused by this node, will also generate these event independently from the contact process. After finishing the exploration, any events that ended in the contact process, will continue generating nodes according to the described procedure. The other nodes, that generate events independent of the contact process, will just sample from X i.i.d. from this point onwards. The process we described is illustrated in figure 11 for the contact process with $d = 1$.

To complete the proof, we realise that, if the Galton-Watson tree dies out after n generations, that means that the contact process has died out before hitting kn frontier events. This is because we have considered consider all the frontier events. We can trivially see that $\mathbb{P}[X = 1] < 1$ for all

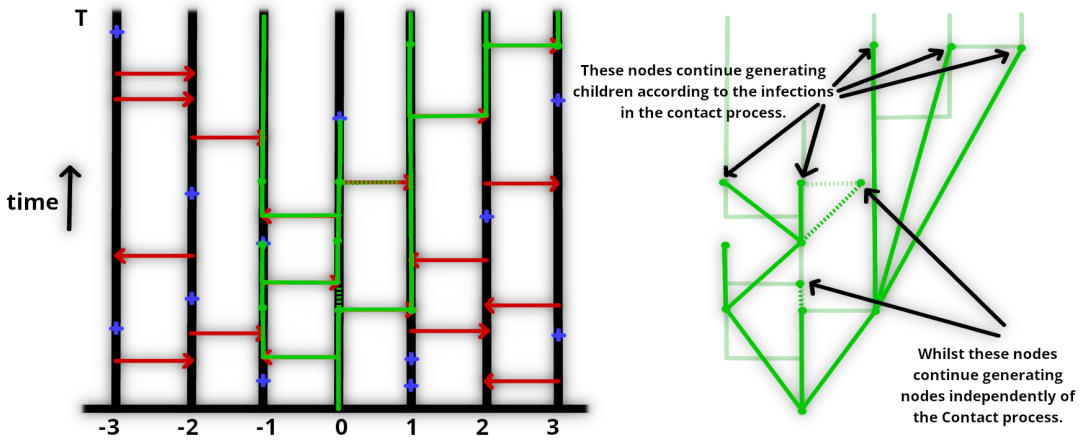


Figure 11: Coupling with Galton Watson trees in one dimension

possible $k \in \mathbb{Z}_{\geq 1}$, because there is always a chance that the first node recovers, equal to $\frac{1}{2*\lambda+1}$, which means that the contact process dies out if $\mathbb{E}[X] \leq 1$ by Theorem 1.14.

□

For small values of k, it is possible to exactly determine the distribution of children. This allows us to establish an asymptotic lower bound.

Theorem 4.4. *For all $d \in \mathbb{Z}_{\geq 1}$ we have that $\lambda_c\left(\mathbb{Z}^d\right) \geq \frac{1}{8} + \frac{\sqrt{18d-1}}{8d\sqrt{2d-1}}$.*

Proof. We will determine the number of children after 2 frontier events, starting with 1 infection. Denote this by X. The probability that the infection remains after 1 frontier event is $\frac{2d\lambda}{2d\lambda+1}$, in which case there will be two infected sites. Either one of these sites infect another site, with probability $\frac{2(2d-1)\lambda}{2(2d-1)\lambda+2}$ or one of them recovers with probability $\frac{2}{2(2d-1)\lambda+2}$. Here we use that a cluster of two infected sites has $2(2d-1)$ uninfected neighbours. Next, we calculate the expected number children;

$$\begin{aligned}
\mathbb{E}[X] &= \frac{2d\lambda}{2d\lambda+1}\left(\frac{2}{2(2d-1)\lambda+2} + 3\frac{2(2d-1)\lambda}{2(2d-1)\lambda+2}\right) \\
&= \frac{2d\lambda}{2d\lambda+1}\frac{6(2d-1)\lambda+2}{2(2d-1)\lambda+2} \\
&= \frac{12d(2d-1)\lambda^2+4d\lambda}{4d(2d-1)\lambda^2+2(2d-1)\lambda+4d\lambda+2}.
\end{aligned}$$

Now we need to solve $\mathbb{E}[X] \leq 1$, expanding gives

$$\begin{aligned}
&12d(2d-1)\lambda^2+4d\lambda \leq 4d(2d-1)\lambda^2+2(2d-1)\lambda+4d\lambda+2 \\
&4d(d-1)\lambda^2-(2d-1)\lambda-1 \leq 0.
\end{aligned}$$

Now, by solving this quadratic equation we find that the infection survives if

$$\begin{aligned}\lambda &\geq \frac{2d-1+\sqrt{(2d-1)^2+16d(2d-1)}}{8d(2d-1)} \\ &= \frac{2d-1+\sqrt{36d^2-20d+1}}{8d(2d-1)} \\ &= \frac{1}{8}+\frac{\sqrt{(2d-1)(18d-1)}}{8d(2d-1)} \\ &= \frac{1}{8}+\frac{\sqrt{18d-1}}{8d\sqrt{2d-1}}.\end{aligned}$$

Which finally allows us to conclude using Lemma 4.3 that $\lambda_c\left(\mathbb{Z}^d\right) \geq \frac{1}{8}+\frac{\sqrt{18d-1}}{8d\sqrt{2d-1}}$. □

Note that Theorem 4.4 can be improved further by increasing the number of frontier events we consider. Doing so will result in polynomials of increasingly higher degrees. In Subsection 5.2 we will do some numerical experiments to see how far we can go in dimension 1. For the remainder of this subsection, we will consider $(X_n)_{n\in\mathbb{Z}_{\geq 0}}$, where X_n denote the number of infections after n frontier events. If for any realisation of the contact process, the number of infection is finite, X_n will be zero for all m after the infection has died out. We would like to show that the method to obtain lower bounds from this chapter allows us to find arbitrarily good lower bounds, as the number of frontier events approaches infinity. Ultimately, we will be unsuccessful, but the following results takes some steps towards this goal, and shows that it is plausible that this is in fact true.

The primary obstacle to show this result is the case, is showing that $\mathbb{E}[X_n]$ converges to 0 in the subcritical case. One might think to try and apply Lebesgue's dominated convergence theorem, since $X_n \to 0$. However, it proves to be very difficult to find a random variable with finite first moment to bound X_n. Instead we will use some results on exponential decay. The proofs of these results are way beyond the scope of this thesis, and together they would probably be a suitable topic for a thesis on their own. For this reason we will just state these results, without the proof.

First we start with a bound on the expected number of infected sites after some time. The following can be found in [8, p. 60].

Theorem 4.5. *There exists a constant c independent of t such that*

$$\mathbb{E}\left[|\xi_t^{\{0\}}|\right] \leq c(1+t^d)\sqrt{\mathbb{P}[\xi_t^0 \neq \varnothing]}.$$

Next we will state a result due to Carol Bezuidenhout and Geoffrey Grimmett in 1990[3], which is often referred to as exponential decay in the subcritical regime. It is also worth mentioning David Griffeath who proved exponential decay for $\mathbb{Z}$ in 1981. We use the version stated in [8, p. 67].

Theorem 4.6. *(C. Bezuidenhout, Geoffrey Grimmett, 1990) Suppose $\lambda < \lambda_c(\mathbb{Z}^d)$, then there are positive constants $\varepsilon(\lambda) > 0$ such that*

$$\mathbb{P}\left[\xi_t^{\{0\}} \neq \varnothing\right] \leq e^{-\varepsilon(\lambda)t}$$

Finally, we can combine these two theorems to show $\mathbb{E}[X_n]$ converges to 0 in the subcritical case.

Theorem 4.7. *Consider the contact process on $\mathbb{Z}^d$. Let X_n denote the number of infections after n frontier events, then*

1. *if $\lambda > \lambda_c$, $\mathbb{E}\left[|\xi_t^{\{0\}}|\right] \to \infty$.*
2. *if $\lambda < \lambda_c$, $\mathbb{E}\left[|\xi_t^{\{0\}}|\right] \to 0$,*

Proof. If $\lambda > \lambda_c$, X_n diverges to infinity with positive probability, so $\mathbb{E}\left[|\xi_t^{\{0\}}|\right] \to \infty$.

If $\lambda < \lambda_c$, we use Theorem 4.5 and Theorem 4.6. Then, since $\lim_{n\to\infty} \frac{t^d}{e^t} = 0$,

$$\mathbb{E}\left[|\xi_t^{\{0\}}|\right] \leq c(1+t^d)\sqrt{\mathbb{P}[\xi_t^{\{0\}} \neq \varnothing]} \leq c(1+t^d)e^{-\frac{1}{2}\varepsilon(\lambda)t} \to 0.$$

□

Theorem 4.7 shows that as time goes to infinite, the expectation $\mathbb{E}\left[|\xi_t^{\{0\}}|\right]$ behaves in the way that we want $\mathbb{E}[X_n]$ to behave. It might seem like the same convergence for $\mathbb{E}[X_n]$ should then immediately follow from this Theorem, but when one writes out the details we find many technical challenges. If one would further research the method from this paper, completing this proof would be a good place to start.

In Subsection 5.2 we will compute $\mathbb{E}[X_n]$ for increasing value of n to get numerical lower bounds. In this section we will see that the lower bound increases rather slowly as n increases.

4.8 Upper bound

From Theorem 2.17 we can immediately conclude the following statement.

Corollary 4.9. *For all $d \in \mathbb{N}$ we have that $\lambda_c(\mathbb{Z}^{d+1}) \leq \lambda_c(\mathbb{Z}^d)$*

This implies that an upper bound of the contact process on $\mathbb{Z}$ gives an upper bound for all $\mathbb{Z}^d$. We can do much better however. The following is a result by Richard Holley and Thomas Liggett.

Theorem 4.10. *(R. Holley, T. M. Liggett, 1978) Let $\{\xi_t\}_{t\geq 0}$ by the contact process on $\mathbb{Z}^{nd}$ with parameter λ and $\{\zeta_t\}_{t\geq 0}$ the contact process on $\mathbb{Z}^n$ with parameter λd. Then,*

1. $$\mathbb{P}[\xi_t^{\{0\}} \neq \varnothing] \leq \mathbb{P}[\zeta_t^{\{0\}} \neq \varnothing]$$

   for all $t \geq 0$.

2. $d\lambda_c\left(\mathbb{Z}^{nd}\right) \leq \lambda_c\left(\mathbb{Z}^n\right)$

This is a result from [7, p.307]. We have stated a slightly more generalised version than the one there, which allows to compare $\mathbb{Z}^{nd}$ with $\mathbb{Z}^n$ for all n, instead of just the case $n = 1$.

Proof. The implication $1 \implies 2$ follows by Lemma 2.16 and the definition of the critical value. To proof the first point we define a function $\pi_{n,d} \colon \mathbb{Z}^{dn} \mapsto \mathbb{Z}^n$. Note that we are referring to the sets Z^d. The function is defined as follows,

$$\pi_{n,d}(x_1, \ldots, x_{nd}) = \left(\sum_{i=1}^{d} x_i, \sum_{i=d+1}^{2d} x_i, \ldots, \sum_{i=(n-1)d+1}^{dn} x_i \right). \tag{4.1}$$

and for any finite set A,

$$\pi_d(A) = \{\pi_{n,d}(x) \mid x \in A\}.$$

We will now make a coupling between $\{\xi_t\}_{t\geq 0}$ and $\{\zeta_t\}_{t\geq 0}$ that ensures that

$$\zeta_t \subset \pi_{n,d}(\xi_d). \tag{4.2}$$

If we have found this coupling, it is clear that 1 holds. First note that at $t = 0$, (4.2) is satisfied. For each $y \in \zeta_t$, there must at least one x_y such that $\pi_{n,d}(x_y) = y$. For each t we let $X_t = \{x_y | y \in \zeta_t\}$ be the set that contain exactly one x_y for each $y \in \zeta_t$. If we consider (4.1) we can see that for every neighbour in $\mathbb{Z}$ there are d neighbours in $\mathbb{Z}^{nd}$ since we can disturb d different coordinates in the same way to get the same output of $\pi_{n,d}$. Let us now write down the exact coupling.

For $x_y \in X_T$,	$(\zeta_t, \xi_t) \mapsto (\zeta_t \setminus \{\pi_{n,d}(x_y)\}, \xi_t \setminus \{x_y\})$ *with rate* 1,
For $x \in \xi_t \setminus X_t$,	$(\zeta_t, \xi_t) \mapsto (\zeta_t, \xi_t \setminus \{x\})$ *with rate* 1,
For $x \notin \xi_t$,	$(\zeta_t, \xi_t) \mapsto (\zeta_t \cup \{\pi_{n,d}(x)\}, \xi_t \cup \{x\})$ *with rate* $\lambda\lvert\{x_y \in X_t \colon \{x, x_y\} \in E\}\rvert$,
For $x \notin \xi_t$,	$(\zeta_t, \xi_t) \mapsto (\zeta_t, \xi_t \cup \{x\})$ *with rate* $\lambda\lvert\{y \in \xi_t \setminus X_t \colon \{x, y\} \in E\}\rvert$.

With this coupling, for a recovery in ζ_t, we will there is a recovery in ξ_t. If $y - 1$ is not in B, we say there is a infection in $y - 1$ due to the presence of y if any of the neighbours of x_y with $\pi_{n,d}(x_y) = y - 1$ get infected due to the presence of x_y, even if this neighbour was already infected in ξ_t. The construction is analogous for $y + 1$. This coupling ensures that the infection rate of $\{\zeta_t\}_{t\geq 0}$ will be d times the infection rate of $\{\xi_t\}_{t\geq 0}$. Any points in $\{\zeta_t\}_{t\geq 0}$ that are not used will evolve independently of $\{\xi_t\}_{t\geq 0}$ according to the normal contact process. Finally, using this construction, it is clear that 4.2 remains satisfied, since if for $x \in \mathbb{Z}^n$ the final point in $\pi_{n,d}^{-1}(x)$ recovers, x must also recover, and if there is an infection to $x \in \mathbb{Z}^n$, it is ensured that there is at least one node is infected in $\pi_{n,d}^{-1}(x)$. We have now constructed a coupling with all desired properties, which completes the proof. □

In order to derive an upper bound on $\lambda_c(\mathbb{Z})$, we will make a comparison with another stochastic process, called oriented percolation. We will find that a realisation of a certain percolation process is contained within the visual representation of the contact process. Then, by referring to results on the survival of percolation process, we can show an upper bound.

Theorem 4.11. *The process* $\{\xi_t\}_{t\geq 0}$ *survives if* $\left(1 - e^{-\lambda T}\right) e^{-T} > p_c$ *for all* $T \in \mathbb{R}$. *Where* p_c *denotes the critical probability for a certain type of* 1*-independent oriented percolation.*

Proof. We will use Harris' visual representation of the contact process on the graph $\mathbb{Z}_{\geq 0}$, and make a partition in time with time steps T. We will define an 1-independent oriented percolation process based on this visual representation. We let points in $(z_1, z_2) \in \mathbb{Z}_{\geq 0} \times \mathbb{Z}_{\geq 0}$ correspond to z_1 at time step $(z_1 + z_2)T$ in Harris's visual representation. We say that the edge between (z_1, z_2) and $(z_1 + 1, z_2)$ is open in the percolation process if the following conditions hold:

- There is an infection from z_1 to $z_1 + 1$ between $(z_1 + z_2)T$ and $(z_1 + z_2 + 1)T$. Denote the time of infection by $\tilde{T}$.
- There are no recoveries of the node z_1 between $(z_1 + z_2)T$ and $\tilde{T}$
- There are no recoveries of the node $z_1 + 1$ between $\tilde{T}$ and $(z_1 + z_2 + 1)T$

For the vertical edges we say that the edge between (z_1, z_2) and $(z_1, z_2 + 1)$ is open if

- There are no recoveries of the node z_1 between $(z_1 + z_2)T$ and $(z_1 + z_2 + 1)T$

How paths in the contact process are related to paths in oriented percolation is visualised in Figure 12. The probability of at least one infection in a time interval of length T is equal to $1 - e^{-}\lambda T$ and the probability of no recoveries in these time intervals is equal to e^{-T}. Consequently, this gives an one-independent oriented percolation process where edges are open with probability greater or equal to $e^{-T}\left(1 - e^{-\lambda T}\right)$. The probabilities are also 1-independent, since the probabilities of an edges being open only depend on whether or not the neighbouring edges are open.

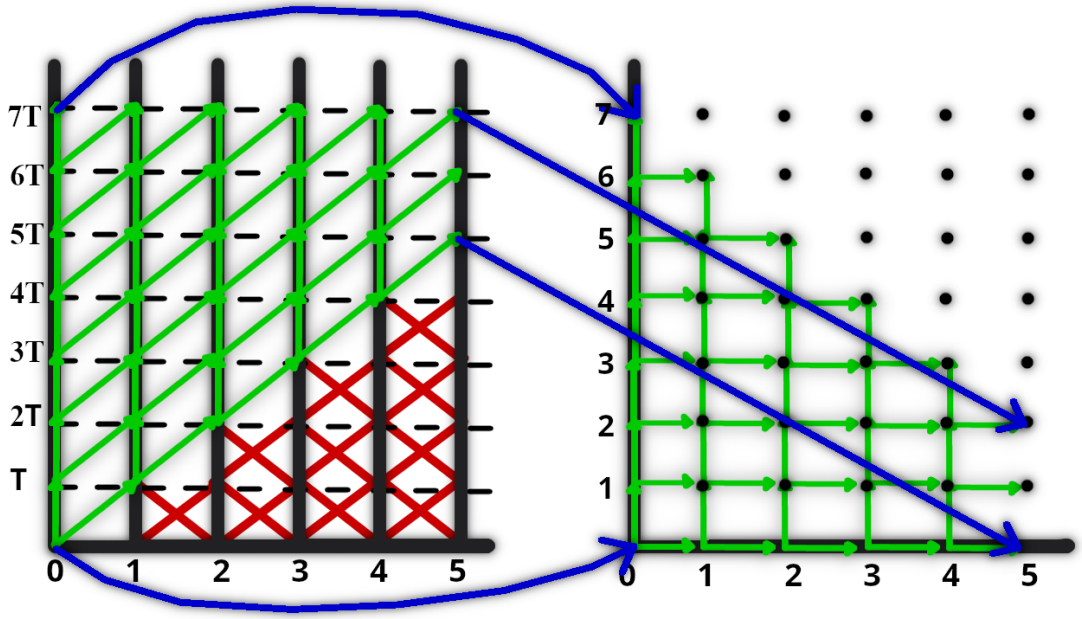


Figure 12: Coupling with Oriented percolation

This final part of the proof is showing that the event of an infinite path in the oriented percolation process is a subset of the event of an infinite path in the visual representation, which implies that the infection survives.

Let us consider an infinite path in the oriented percolation process, which starts at $(0, 0)$ without loss of generality. If $(0, 0)$ is open there is an infection to node 1, and no recoveries on node 0 or 1 in the next time step. Meaning there is a path to $(0, 1)$ and to $(1, 0)$ in the visual representation as desired. We can repeat this reasoning to all further nodes in the infinite path. This means that an infinite path in our 1-independent oriented percolation process implies a path in the visual representation. □

The event that neighbouring edges are open is positively correlated in the 1-independent percolation model above. This fact suggests that the critical probability of the model is smaller than that of normal oriented percolation.

Lemma 4.12. *The critical value of the* 1*-independent percolation model described in Theorem 4.11 is smaller or equal to* $p_c(\mathbb{Z}^2)_{oriented}$.

Proof. The proof consists of two steps, we first relate the critical percolation of the 1-independent model to that of another 1-independent percolation model such that the probability of an edge

being open, in the absence of other information, is the same for all edges. Secondly, we show that the critical probability of this second model is smaller than that of ordinary oriented percolation.

To do the first step of the proof, note that we can get a 1-independent percolation model where all edges are open with probability $\left(1-e^{-\lambda T}\right)e^{-T}$ by resampling open vertical edges, so that after the resampling, the probability of any vertical edge being open is $\left(1-e^{-\lambda T}\right)e^{-T}$. Doing this only increases the critical probability, since more edges get closed. Denote this new model by $1OP$

Now note that if an edge is open, it never decrease the probability that the a neighbouring edge is open in the dependent oriented percolation process. Since the requirement for an edge to be open, overlaps with the requirements for a neighbouring edge to be open. Thus, if $p = e_1, e_2, \ldots, e_n$ is a path of length n in oriented percolation,

$$\begin{aligned}\mathbb{P}[p \text{ open in } 1OP] &= \mathbb{P}[e_1 \text{ open in } 1OP]\,\mathbb{P}[e_2 \text{ open in } 1OP \mid e_1 \text{ open in } 1OP]\ldots \\ &\quad \mathbb{P}[e_n \text{ open in } 1OP \mid e_{n-1} \text{ open in } 1OP] \\ &\geq \mathbb{P}[e_1 \text{ open in } OP]\,\mathbb{P}[e_2 \text{ open in } OP]\ldots\mathbb{P}[e_n \text{ open in } OP] \\ &= \mathbb{P}[p \text{ open in } OP]\end{aligned}$$

Let Λ_n denote the box with radius n surrounding the origin. Then we get,

$$\begin{aligned}\mathbb{P}[\underline{0} \rightsquigarrow \delta\Lambda_n \text{in 1-independent model}] &= \mathbb{P}\left[\bigcup_{p \text{ path from } \underline{0} \text{ to } \delta\Lambda_n} p \text{ open in 1-independent}\right] \\ &\geq \mathbb{P}\left[\bigcup_{p \text{ path from } \underline{0} \text{ to } \delta\Lambda_n} p \text{ open in ordinary } OP\right] \\ &= \mathbb{P}[\underline{0} \rightsquigarrow \delta\Lambda_n \text{in ordinary } OP.]\end{aligned}$$

Taking the limit in the inequality above than proves the desired result. □

To complete the upper bound, we need to determine an upper bound of the critical value of oriented edge percolation. To get an upper bound that is as good as possible, we will refer to one result from literature on oriented percolation. The following result is proven in [1].

Theorem 4.13. *(P. Ballister, B. Bollobàs, A. Stacey, 1993) The following inequality holds on critical value of oriented edge percolation* $p_c^{oriented} \leq 0.6863$.

Finally we can combine all these results to find an upper bound on the critical value of the contact process on $\mathbb{Z}$, by finding an optimal value of T.

Corollary 4.14. *The value* $\lambda_c(\mathbb{Z})$ *is smaller than* 8.3730.

Proof. We have to find the value of T such that

$$\left(1-e^{-\lambda T}\right)e^{-T} < 0.6863$$

holds for the smallest possible value of λ_{min}. Then, Theorem 4.11, Lemma 4.12 and Theorem 4.13 allows us to conclude that $\lambda_c(\mathbb{Z}) < \lambda_{min}$.

Assuming that T is positive allows us to substitute $T = -\log a$.

$$\begin{aligned}&\left(1-e^{\lambda T}\right)e^{-T} = 0.6863 \\ &\left(1-a^{\lambda}\right)a = 0.6863 \\ &a^{\lambda+1} = a - 0.6863 \\ &\lambda = -1 + \log_a\left(a - 0.6863\right)\end{aligned}$$

We substitute $a = e^{-T}$ back in and minimise $\lambda = -1 + \log_{e^{-T}}\left(e^{-T} - 0.6863\right)$. There exists no nice algebraic expression for this, but numerical estimation gives a reasonable estimate $T \approx 0.2673$, yielding $\lambda_{min} < 8.3730$ □

It is worth noting that connections have already been made between the contact process and oriented percolation. The connections have been very impactful to the study of the contact process and can be read about in the 1990 paper by Carol Bezuidenhout and Geoffrey Grimmett [2].

To end this chapter we combine all the results on the critical value of the contact process obtained so far.

Corollary 4.15. *The following bounds on the critical value of $\mathbb{Z}^d$ hold.*

$$\frac{1}{8} + \frac{\sqrt{18d-1}}{8d\sqrt{2d-1}} \leq \lambda_c(\mathbb{Z}^d) < \frac{8.3730}{d}$$

Proof. Combine Theorem 4.4, Theorem 4.10 and Corollary 4.14. □

These estimates, especially the upper bound, are not particularly tight. The coupling with oriented percolation requires strong results on the critical value of oriented percolation, which makes it a rather weak result. Much tighter bound are available in the literature, namely it is known that [7, p. 308]

$$\frac{1}{2d-1} \leq \lambda_c(\mathbb{Z}^d) < \frac{2}{d}. \tag{4.3}$$

5 Numerical methods

While working on this thesis, extensive use has been made of computational methods. In this final chapter, I have compiled the most interesting results. Full scripts are provided in the appendix, together with a link to a GitHub repository with minimal working examples.

5.1 Visualisation

One useful tool to help understand a complex stochastic process, like the contact process, is a visualisation. Being able to see how the process behaves could give insights that are not obvious from the mathematical definition. In Appendix A.2 one can find a script for a simulation of the contact process on a small part of $\mathbb{Z}^2$, plotting how it evolves in real-time. It includes controls for adjust the rate in real time with the scroll wheel, toggling full-screen with F11 and toggling a graph with the total number of infections using the left mouse button.

The heart of the simulation is the `contact_process` function. We generate Poisson events for all recoveries and infections in the time interval. The final state of the node is then updated in global list `inf` according to the most recent event. This sort of simulation is sometimes called tau-leaping, and is not entirely exact. One can imagine a situation where a node is healthy, get infected and infects one of its neighbours within one timestep. This infection of its neighbour would not be registered by this simulation.

However the big advantage of this technique is that the time required to simulate it does not grow as fast when a large number of nodes is infected. This makes it ideal for visualisation, as drawing the picture already takes up a large amount of the computing resources. The visualisation has been used in introduction to generate Figure 1.

5.2 Improved lower bound

In Lemma 4.3 we derived a result which allows us to find a lower bound on the critical value of the contact process. We have done so for 2 frontier events in Theorem 4.4. It is interesting to ask ourselves, how far can we go? In the subsection we will first numerically prove a stronger lower bound in one dimension, using the computer algebra package `sympy`. Finally, we will numerically calculate the expected value for $\lambda = 1.05$ to show that this value is a lower bound of the critical value $\lambda_c(\mathbb{Z})$.

Let us first look at the algebraic solver. We will consider what the contact process looks like, with an equivalence up to horizontal shifts. That means that in our program we will consider $\xi_t = \{-1, 1, 2\}$ to be the same as $\xi_t = \{2, 4, 5\}$. This will drastically reduce the number of steps our algorithm takes as we consider larger numbers of frontier events.

In our program we will start with a dictionary containing the state string `xox` and associated probability 1. This state string represents the class of states where one site is infected. Now, to calculate the distribution of frontiers after the next frontier event, we loop through every state string. For this state string we will consider all possible events and the probability that they occur. We track of these states and probabilities in a new dictionary, which can than be used for the next iteration of the loop.

A full implementation can be found in Appendix A.3. Using this implementation we now find the next lower bound.

Theorem 5.3. *The following lower bound holds*

$$\lambda_c(\mathbb{Z}) \geq 0.8958876.$$

Proof. Using the script in Appendix 4.3 until $d = 11$ together with Lemma 4.4 yields that $\lambda_c(\mathbb{Z}) \geq 0.8958876$. □

However we can go further still. If we fix the value of λ and omit the symbolic expressions, the script can be sped up significantly, allowing us to go to greater depths than $d = 11$. We see that for $\lambda = 1$ the expected value dips below 1 after $d = 24$. However, not unsurprisingly, once we increase the value to $\lambda = 1.1$, by $d = 24$ the expected value is still approximately 1.29.

Theorem 5.4. *The following lower bound holds,*

$$\lambda_c(\mathbb{Z}) \geq 1.05.$$

Proof. Using the adapted script found in Appendix A.3 we see that with rate 1.05, for $d = 34$ the expected number of children after d frontier events is approximately 0.999052027980148 and hence with great certainty smaller than 1. Allowing us to conclude the lower bound with Lemma 4.4 □

For both of the previous theorems, results were obtained after less than an hour of computation time. The amount of possible states after d frontier events scales roughly like 2^d, so the complexity of the algorithm is exponential. Given this fact, its unlikely that more computing time will lead to drastically better results.

One avenue of possible improvement we came across while working on these results comes from the following observation. If we let X_n denote the number of infection after n frontier events, it seems that in the supercritical case $\mathbb{E}[X_n]$ is strictly increasing. If this is indeed true, in order to show $\lambda \leq \lambda_c$ it suffices to show that $\mathbb{E}[X_n] - \mathbb{E}[X_{n+1}] \geq 0$ for some value of n. This is obviously much less difficult than showing that $\mathbb{E}[X_n] \leq 1$ for some n. A bit of time was spent trying to prove this fact, but it seems rather difficult to do so. Nonetheless, this could be an avenue of further research.

5.5 Numerical estimation of the critical value

Finally, we want to see if we can numerically determine the critical value of the contact process on $\mathbb{Z}$. We will use the bisection method to find the right most zero of the survival function ψ as defined in Definition 2.10. To do so, we will employ a suitable Monte Carlo method. We will simulate all the events in a contact process, starting with just the origin infected, for some number of attempts. If in any of these attempts, we exceed some threshold, we will conclude that this value is right of the rightmost zero. Otherwise we assume it to be left of the rightmost zero.

The simulation stores the contact process in a dictionary with tuples as keys storing node class objects. This node class contains a boolean to indicate whether the node is infected, and a count of all the uninfected neighbours of this node. Indexing our dictionary with tuples allows us to write the algorithm for arbitrary dimensions.

To generate the next event, we uniformly sample in $[0, 1]$ and than pick whether or not this event is a recovery or an infection by checking if our sampled value is smaller than the total infection rate divided by the total infection and recovery rate, than we uniformly sample an valid edge to infect or node to recover respectively. This is a valid approach by Lemma 1.6, and is much faster than simulating Poisson processes.

An implementation of this method is included in Appendix A.4. Some steps have been taken to optimise this script, for example, we are using the `operator.add` function to utilise `c` language speeds for checking all neighbouring tuples. Additionally, we use the `loky` library to allow for multiprocessing, yielding practical speed-up of about 4 to 8 times.

Using a threshold of 1000 infections, with 25 attempts, we tried estimating the critical value of the contact process for dimensions 1 to 20, starting from the bounds by Liggett (4.3). The threshold was too low to obtain any meaningful results for dimension 11 to 20, but for dimension 1 to 10 we got the results in Table 1 after several hours of computing.

Dimension	1	2	3	4	5	6	7	8	9	10
Estimate	1.646	0.413	0.219	0.153	0.114	0.099	0.079	0.072	0.060	0.054

Table 1: Estimates of the critical value for dimension 1 to 10

One might wonder how good these estimates are. However, we can compare them to other numerical estimates, like the ones by Munir Sabag and Mário de Oliveira [10], who did estimations up to dimension 5 for the conserved contact process back in 2002. It is noted by the authors that these values seems to agree with those of the ordinary contact process, which suffices for our purposes. Their estimations, corrected to undo normalisation, are shown in Table 2.

Dimension	1	2	3	4	5
Estimate	1.6489	0.41218	0.21947	0.14938	0.11384

Table 2: Estimates of the critical value for dimension 1 to 5 by Sabad and de Oliveira

Comparing these values with our estimations, we see that for these first 5 dimensions, we have a deviation of at most 0.005.

We can also employ this method to estimate how our survival function ψ from Definition 2.10 develops by counting the number of times we exceed the threshold in our total amount of attempts. Since this requires us to do many more attempts, we have to use a lower threshold to stay within a reasonable amount of computing time. One of these estimations is shown in Figure 5. This was generated with 250 attempts and a threshold of 100, requiring about 2 hours of computing. We calculated an empirical probability for several values, increasing the density as we approach the critical value, to compensate for the larger slope.

6 References

[1] Paul Balister, Bela Bollobas, and Alan Stacey. Upper bounds for the critical probability of oriented percolation in two dimensions. *Proceedings: Mathematical and Physical Sciences*, 440(1908):201–220, 1993.

[2] Carol Bezuidenhout and Geoffrey Grimmett. The Critical Contact Process Dies Out. *The Annals of Probability*, 18(4):1462 – 1482, 1990.

[3] Carol Bezuidenhout and Geoffrey Grimmett. Exponential decay for subcritical contact and percolation processes. *The Annals of Probability*, pages 984–1009, 1991.

[4] G. Grimmett and D. Stirzaker. *Probability and Random Processes*. Probability and Random Processes. OUP Oxford, 2001.

[5] Geoffrey Grimmett. *Percolation*, volume 321 of *Grundlehren der mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, second edition, 1999.

[6] G.F. Lawler. *Introduction to Stochastic Processes*. Chapman & Hall/CRC Probability Series. CRC Press, 2018.

[7] T.M. Liggett. *Interacting Particle Systems*. Grundlehren der mathematischen Wissenschaften : a series of comprehensive studies in mathematics. Springer-Verlag, 1985.

[8] T.M. Liggett. *Stochastic Interacting Systems: Contact, Voter and Exclusion Processes*. Die Grundlehren der mathematischen Wissenschaften in Einzeldarstellungen. Springer, 1999.

[9] S.M. Ross. *Introduction to Probability Models*. Probability and Statistics. Academic Press, 2007.

[10] Munir M. S. Sabag and Mário J. de Oliveira. Conserved contact process in one to five dimensions. *Phys. Rev. E*, 66:036115, Sep 2002.

[11] R. van der Hofstad. *Random Graphs and Complex Networks*. Number v. 1 in Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 2017.

A Python scripts

All scripts have been confirmed to work with Python version 3.13.2. In the table below you can find all packages used, with the specific version.

Package	Version	Available at
numpy	2.1.3	`https://pypi.org/project/numpy/2.1.3/`
matplotlib	3.10.1	`https://pypi.org/project/matplotlib/3.10.1/`
scipy	1.15.2	`https://pypi.org/project/scipy/1.15.2/`
networkx	3.4.2	`https://pypi.org/project/networkx/3.4.2/`
loky	3.5.2	`https://pypi.org/project/loky/3.5.2/`
psutil	7.0.0	`https://pypi.org/project/psutil/7.0.0/`

The code in the chapter has been reduced to a minimal working example, wherever that makes sense. This means that all code that is not relevant to solving the problem at hand, such as plot labels, has been removed. All scripts are accessible at this GitHub page.

A.1 Expected extinction time on cyclic graphs

In subsection 3.1 we derived a system of equation to find and lower and upper bound on the expected extinction time of the contact process on a cyclic graph given different initial configurations. We used a symbolic solver to derive Figure 9. This appendix contains the scripts to solve these systems. First, we have a script to solve the system for the lower bound algebraically.

```
1  from sympy import symbols, Rational
2  import sympy.plotting as spp
3  from sympy.matrices import Matrix
4  import math
5  '''
6  We define a sympy symbol, specify the dimension of the system, and initialise the
   ↪  matrix and vector.
7  '''
8  l = symbols("l")
9  n = 5
10 b = [1/(2*l + i + 1) for i in range(n - 1)] + [Rational(1, n)]
11 A = []
12 for i in range(n - 1):
13     A.append([-(i + 1)/(2*l + i + 1) if i == j + 1 else 1 if i == j else -(2 *
       ↪  l)/(2*l + i + 1) if i == j - 1 else 0 for j in range(n)])
14 A.append([0 for _ in range(n-2)] + [-1, 1])
15 '''
16 We solve the system, and plot the result.
17 '''
18 r = Matrix(A).LUsolve(Matrix(b))
19 p1 = spp.plot(r[-1], (l, 0, 3), show=False)
20 p1.show()
```

We can easily adapt this script to solve the upper bound system, by replacing line 10 to line 14 with the following code snippet.

```
b = [1 / (2 * i * l + i + 1) if i <= math.floor((n - 1) / 2) else 1 / (2 * (n - i - 1) * l + i + 1) for i in range(n - 1)] + [Rational(1,n)]
A = []
for i in range(n - 1):
    A.append([
        -(i + 1) / (2 * min(i, n - i - 1) * l + i + 1) if i == j + 1 else 1 if i == j else
        -(2 * min(i, n - i - 1) * l) / (2 * min(i, n - i - 1) * l + i + 1) if i == j - 1 else 0 for j in range(n)
    ])
A.append([0 for _ in range(n-2)] + [-1, 1])
```

We also studied the behaviour of the bounds for fixed lambda, as n grows. To solve these larger systems, we use a sparse matrix implementation. This can solve the systems at a fraction of the time required compared to the regular solver. These scripts have been used to generate the plots in Figure 10.

```
from scipy.sparse import diags
from scipy.sparse.linalg import spsolve
import math
import numpy as np
import matplotlib.pyplot as plt
'''
We define a rate, specify the range of dimension for which we solve the system, and initialise the matrix and vector.
'''
l = 0.25
n_values = range(2, 40)  # Range of n values to observe growth
solutions = []
'''
Solve the system and plot the solution
'''
for n in n_values:
    b = np.array([1 / (2 * l + i + 1) for i in range(n - 1)] + [1 / n])
    diagonals = [
        [-((i + 1) / (2 * l + i + 1)) for i in range(1, n)],  # Sub-diagonal
        [1] * n,                                              # Main diagonal
        [-(2 * l) / (2 * l + i + 1) for i in range(n - 1)]    # Super-diagonal
    ]
    A = diags(diagonals, offsets=[-1, 0, 1], shape=(n, n)).toarray()
    A[-1, -2:] = [-1, 1]
    r = spsolve(A, b)
    solutions.append(r[-1])
plt.plot(n_values, solutions)
plt.show()
```

Where again the script can easily be adapt to work for the upper bound by replacing line 15 to 19 by the next snippet.

```
b = np.array([1 / (2 * i * l + i + 1) if i <= math.floor((n - 1) / 2) else 1 / (2 * (n - i - 1) * l + i + 1) for i in range(n - 1)] + [1 / n])
diagonals = [
```

```
    [-(i + 1) / (2 * min(i, n - i - 1) * l + i + 1) for i in range(1, n)],  # Sub-diagonal
    [1] * n,  # Main diagonal
    [-(2 * min(i, n - i - 1) * l) / (2 * min(i, n - i - 1) * l + i + 1) for i in range(n - 1)]  # Super-diagonal
]
```

A.2 Visualisation

The following script does an estimation of the contact process as described in subsection 5.1. As a reminder, it includes controls to adjust the rate in real time with the scroll wheel, toggle full screen with F11 and toggle a graph with the total number of infections using the left mouse button.

```
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
import tkinter as tk
from tkinter import *
import psutil
import numpy as np
import networkx as nx

'''
We configure a Tkinter window, and the associated objects required to draw the network and graph.
'''
root = Tk()
defaultFont = tk.font.nametofont('TkDefaultFont')
frame= Frame(root).pack(fill= BOTH, padx= 0, pady=0)
root.title("Contact process")
screen_width = root.winfo_screenwidth()
screen_height = root.winfo_screenheight()
fig = plt.figure(frameon=True, figsize=(screen_width/100,screen_height/100), dpi=102)
canvas = FigureCanvasTkAgg(fig, root)

'''
Keybinds:
button_0    :   Display a graph that shows the number of infection over time.
scroll      :   Increase/decrease the rate of infection.
F11         :   Toggle fullscreen.
'''
root.bind("<F11>", lambda e: root.attributes("-fullscreen", not root.attributes("-fullscreen")))
root.bind("<Button-1>", lambda event: globals().__setitem__('display_infections', not display_infections))
root.bind("<MouseWheel>", lambda event: globals().__setitem__('r', max(0, r + 0.1 if event.delta > 0 else r - 0.1)))

'''
Main variables:
```

```
G                   :   The graph on which we run the contact process.
dt                  :   The timestep in ms at which we update the graph.
T                   :   The final time in ms the simulation
r                   :   The initial infection rate per second
display_infection   :   Bool that determines whether or not a graph is displayed that shows recent infections.
inf                 :   A list of infected nodes
inf_count           :   Stores the number of infected nodes at each timestep
'''
scale = 35
G = nx.grid_2d_graph((int)(screen_height/scale), (int)(screen_width/scale))
dt, T, r = 500, 1000000, 0.5
global display_infections; display_infections = False
inf = {list(G)[i] : 1 for i in range(len(list(G)))}
pos = {(x,y):(y,-x) for x,y in G.nodes()}
inf_count = []

def generate_poisson_events(rate, time_duration):
    '''
    rate            :   The rate of the Poisson process
    time_duration   :   The time interval in which to generate events.
    Runs the poisson process and returns the time of the last hit. If not hits occur, returns -1.
    '''
    num_events = np.random.poisson(rate * time_duration)
    event_times = np.sort(np.random.uniform(0, time_duration, num_events))
    if num_events > 0:
       return np.max(event_times)
    return -1

def contact_process(graph, dt, rate):
    '''
    graph     :   The graph on which we do the contact process
    dt        :   The simulation timestep
    rate      :   The rate of the contact process
    Updates the inf array to represent the contact process and append the number of infection at this time to inf_count
    '''
    for i in graph.nodes():
        recov_time = generate_poisson_events(1/1000, dt)
        inf_time = -1
        for j in nx.all_neighbors(graph, i):
            if inf[j] == 1:
                inf_time = max(inf_time, generate_poisson_events(rate/1000, dt))
        if inf_time >= 0 or recov_time >= 0:
            inf[i] = 0 if recov_time > inf_time else 1
    inf_count.append(sum(1 for value in inf.values() if value == 1))

'''Generates colours for networkx to draw infected nodes red, and healthy ones blue'''
def generate_colour_map(inf):
    return ['#BF211E' if value == 1 else '#1f77b4' for value in inf.values()]

```

```
82
83  def timeStep(current_time):
84      '''
85      current_time    :   The time at which we want to display the contact process
86      This function clears the previous figure does the following three things
87      1.  Draw a label that show the current rate
88      2.  Draw G
89      3.  If display_infections is true, draw a graph that shows the number of
        ↪  infections
90      '''
91      plt.clf()
92      r_label = Label(root, text=f"Rate (l): {r:.2f}", font=("Open Sans", 36),
        ↪  fg="black", bg="white", bd=5, relief="solid")
93      r_label.place(relx=0.5, rely=0.03, anchor="n")
94      r_label.config(text=f"Rate (l): {r:.2f}")
95
96      col = generate_colour_map(inf)
97      ax = plt.gca()
98      ax.set_xlim([min(x for x, y in pos.values()) + 1, max(x for x, y in
        ↪  pos.values()) - 1])
99      ax.set_ylim([min(y for x, y in pos.values()) + 1, max(y for x, y in
        ↪  pos.values()) - 1])
100     ax.set_axis_off()
101     nx.draw(G, pos, ax=ax, node_size=screen_width/(0.1*scale), node_color=col,
        ↪  with_labels=False, edgecolors='black', linewidths=1.5, node_shape='s')
102     plt.subplots_adjust(left=0, right=1, top=1, bottom=0)
103
104     if(display_infections):
105         plt.subplot(1, 2, 1, facecolor=(1, 1, 1, 0.8))
106         plt.plot(inf_count[max(0, len(inf_count) - 100): len(inf_count) -
            ↪  1],linewidth=5)
107         plt.ylim(0, len(G.nodes()))
108         plt.xlabel("time (ms)", labelpad=-35, loc='center', fontsize=18),
            ↪  plt.ylabel("infections", labelpad=-35, loc='center', fontsize=18)
109         plt.title("Infected sites over time", pad=-20, loc='center', y=0.95,
            ↪  fontsize=36)
110         plt.gca().tick_params(labelsize=0)
111         for spine in plt.gca().spines.values():
112             spine.set_edgecolor('black')
113             spine.set_linewidth(5)
114
115     canvas.draw()
116     canvas.get_tk_widget().pack(side=TOP, fill=BOTH, expand=True)
117
118 '''
119 Main loop:
120 Runs the contact and visualises the contact process until time T.
121 '''
122 for i in range((int)(T/dt)):
123     if not (1 in inf.values()):
124         break
125     root.update_idletasks()
126     root.update()
```

```
    contact_process(G, dt, r)
    root.after(0, timeStep(i*dt))
root.mainloop()
```

A.3 Improved lower bound

This is the script associated with the algebraic solver in subsection 5.2.

```
import sympy as sp
'''
First initialise key variables.  The variable initial is a dictionary that has all frontiers in a step,
with the associated probability. In this dictionary, zeroes denote infected sites, crosses healthy ones.
The max number of iterations the program will run is stored in d. For the sympy algebra we define a
symbol lambda.
'''
initial = { "xox" : 1}
d = 10
l = sp.Symbol('l')
'''
state   :   String that represents a frontier state.
Finds total rate of events that change the frontier.
'''
def getTotalRate(state):
    rate = 2*l
    for i in range(1, len(state) - 1):
        rate += 1 if state[i] == 'o' else l * (int(state[i - 1] == 'o') + int(state[i + 1] == 'o'))
    return rate
'''
string      :   String to change
insert      :   The value to insert into string
position    :   Position at which we will insert string
Returns string with insert inserted at position.
'''
def stringReplace(string, insert, position):
    return string[:position] + insert + string[position + 1:]
'''
heal        :   A bool that indicates whether or not the event is a recovery
i           :   The current index in state
totalRate   :   The total rate of infection/healing in state
state       :   A string that indicates currect state of the frontier
probability :   The probability of reaching state
next        :   The dictionary of possible states with probabilities after one more event
Returns updated next with the events at i accounted for.
'''
def event(heal, i, totalRate, state, probability, next):
    if(heal):
        st = stringReplace(state, 'x', i)
        while st[1] == 'x' and st != 'xx':
```

```
            st = st[1:]
        while st[len(st) - 2] == 'x' and st != 'xx':
            st = st[:len(st) - 1]
        prob = probability * 1/totalRate
    else:
        st = stringReplace(state, 'xo' if i == 0 else 'ox' if i == len(state) - 1
        ↪  else 'o', i)
        prob = probability * 1/totalRate * l
    next[st] = next.get(st, 0) + prob
    return next
'''
state_dict  :   Dictionary with frontier states and associated probabilities
Returns the expected number of children with the frontier distribution state_dict.
'''
def expectation(state_dict):
    return sum(prob * state.count('o') for state, prob in state_dict.items())

'''
Using the event function, this snippet of code determine and prints the expect
↪  number
of infections after d frontier events. We loop through all states, and to all
↪  nodes
in these states, appending the result of every possible event with the associated
probability to next.
'''
next_state = initial
for index in range(d+1):
    next = {}
    for state in next_state:
        probability = next_state[state]
        totalRate = getTotalRate(state)
        if(state == "xx" ):
            if (state in next):
                next[state] += probability
            else:
                next.update({ state :  probability} )
        elif(state != "xx" ):
            next = event(False, 0, totalRate, state, probability, next)
            next = event(False, len(state) - 1, totalRate, state, probability,
            ↪  next)
            for i in range(1, len(state) - 1):
                if state[i] == 'o':
                    next = event(True, i, totalRate, state, probability, next)
                else:
                    if state[i - 1] == 'o':
                        next = event(False, i, totalRate, state, probability, next)
                    if state[i + 1] == 'o':
                        next = event(False, i, totalRate, state, probability, next)
    next_state = next
    print(f"iteration {index}: l = {sp.nsolve(expectation(next_state) - 1, l, 1)}")
'''
We finally plot the solution to show how the expected value develops.
'''
```

```
p1 = sp.plot(expectation(next_state), xlim = (0, 3), ylim = (0,2), show=False)
p1.show()
```

We can do some slight adaptations to the script to do numerical approximations of the lower bound. By changing `l` in line 10 to a fixed value an than replacing line 61 to 96 with the following snippet we get the expected value for a fixed value of λ.

```
'''
Using the event function, this snippet of code determine and prints the expect
↪  number
of infections after d frontier events. We loop through all states, and to all
↪  nodes
in these states, appending the result of every possible event with the associated
probability to next.
'''
next_state = initial
for index in range(d):
    next = {}
    for state in next_state:
        probability = next_state[state]
        totalRate = getTotalRate(state)

        if(state == "xx"):
            if (state in next):
                next[state] += probability
            else:
                next.update({ state :  probability} )

        elif(state != "xx"):
            next = event(False, 0, totalRate, state, probability, next)
            next = event(False, len(state) - 1, totalRate, state, probability,
            ↪  next)

            for i in range(1, len(state) - 1):
                if state[i] == 'x':
                    if state[i - 1] == 'o':
                        next = event(False, i, totalRate, state, probability, next)
                    if state[i + 1] == 'o':
                        next = event(False, i, totalRate, state, probability, next)
                elif state[i] == 'o':
                    next = event(True, i, totalRate, state, probability, next)
    next_state = next
    print("expected children at depth " + str(index) + " = " +
    ↪  str(expectation(next_state)))
```

A.4 Numerical estimation of the critical value

This final script numerically estimates the critical value of the contact process for the dimensions 1 to 20. The script can easily be modified to exclusively generate estimates for dimension 1 or to estimate the survival functions ψ.

```python
import numpy as np
import loky
from loky import as_completed
import random
import operator
'''
attemps          :      The total number of contact processes that are tried.
thres            :      The total number of infections required to declare survival
max_dimension    :      The largest dimension for which we will determine the
↪  critical probability.
'''

attempts = 25
thres = 500
max_dimension = 20
max_steps = np.inf
neighbour_offset = []
'''
Node object to represent each node in the contact process
Each node has an infected status and a count of uninfected neighbours
'''
class Node:
    def __init__(self, infected = True, uninfected_neighbours=2):
        self.infected = infected
        self.uninfected_neighbours = uninfected_neighbours
    def __repr__(self):
        return f"Node(infected={self.infected},
        ↪  uninfected_neighbours={self.uninfected_neighbours})"
'''
Runs one timestep of the contact process
total_infections    :    Total number of infections in the system
total_rate          :    Total rate of infection in the system
key_list            :    List of keys representing the current state of the system
contact_process     :    Dictionary representing the current state of the system
rate                :    The rate of infection
Returns the updated list of keys, contact process, total infections, and total
↪  rate
'''
def time_step(total_infections, total_rate, key_list, contact_process, rate):
    infection = np.random.uniform(0, 1) < total_rate / (total_infections +
    ↪  total_rate)
    if(not infection):
        index = np.random.randint(0, len(key_list))
        toheal = key_list[index]
        contact_process[toheal].infected = False
        total_rate -= contact_process[key_list[index]].uninfected_neighbours * rate
        total_infections -= 1
        for offset in neighbour_offset:
            neighbour_key = tuple(map(operator.add, key_list[index], offset))
            contact_process[neighbour_key].uninfected_neighbours += 1
            if contact_process[neighbour_key].infected:
                total_rate += rate
        key_list[index] = key_list[-1]
        key_list.pop()
```

```
    elif(infection):
        options = []
        infect = random.choices(key_list,
        ↪  weights=[contact_process[key].uninfected_neighbours for key in
        ↪  key_list], k=1)[0]
        for offset in neighbour_offset:
            neighbour_key = tuple(map(operator.add, infect, offset))
            if neighbour_key in contact_process and not
            ↪  contact_process[neighbour_key].infected:
                options.append(neighbour_key)

        new = np.random.randint(0,len(options))
        contact_process[options[new]].infected = True
        total_rate += contact_process[options[new]].uninfected_neighbours * rate
        total_infections += 1
        key_list.append(options[new])
        for offset in neighbour_offset:
            neighbour_key = tuple(map(operator.add, options[new], offset))
            if neighbour_key not in contact_process:
                contact_process[neighbour_key] = Node(False, 2*dimension - 1)
            else:
                contact_process[neighbour_key].uninfected_neighbours -= 1
                if contact_process[neighbour_key].infected:
                    total_rate -= rate
    return total_infections, total_rate, key_list, contact_process

'''
Threshold   :    The threshold for the number of infections
dimension   :    Dimension of the graph Z^d
Run one iteration of the contact process until the number of infections is greater
↪  than 100 or smaller or equal to 0
Returns True if the number of infections is greater than or equal to the threshold,
↪  False otherwise
'''
ti = 0
tr = 0
def run_contact_process(rate, dimension):
    akeys_lst = []
    akeys_lst.append(tuple((0 for _ in range(dimension))))
    cprocess = {tuple([0 for _ in range(dimension)]): Node(True, 2*dimension)}
    for i in range(dimension):
        for offset in [1, -1]:
            neighbour_key = tuple(tuple((0 for _ in range(dimension)))[j] + (offset
            ↪  if j == i else 0) for j in range(dimension))
            cprocess[neighbour_key] = Node(False, 2*dimension - 1)
    ti = 1
    tr = 2 * rate * dimension
    steps = 0
    while ti < thres and ti > 0 and steps < max_steps:
        steps += 1
        ti, tr, akeys_lst, cprocess = time_step(ti,tr,akeys_lst, cprocess, rate)
    if ti >= thres:
        return True
```

```
    return False

''''
low         :    The lower bound for the critical value
high        :    The upper bound for the critical value
attempts    :    The number of attempts to run the contact process
epsilon     :    The tolerance for the bisection method
dimension   :    Dimension of the graph Z^d
Given a lower and upper bound, estimate the critical value using bisection method.
Returns lower and upper estimate for the critical value
'''
def crit_value_bisection(low, high, attempts, epsilon, dimension):
    with loky.get_reusable_executor(max_workers=14) as executor:
        while high - low > epsilon:
            found_true = False
            tasks_completed = 0
            mid = (low + high) / 2
            futures = set()
            for _ in range(attempts):
                future = executor.submit(run_contact_process, mid, dimension)
                futures.add(future)
            for future in as_completed(futures):
                tasks_completed += 1
                result = future.result()
                if result:
                    found_true = True
                if found_true:
                    break
            if found_true:
                high = mid
            else:
                low = mid
            print(f"The critical value is likely between {low:.6f} and
            ↪  {high:.6f}.")
    return high, low

'''
Determine the critical value in for various dimensions.
'''
for dimension in range(1, max_dimension + 1):
    neighbour_offset = []
    for i in range(dimension):
        for offset in [1, -1]:
            neighbour_offset.append([offset if j == i else 0 for j in
            ↪  range(dimension)])
    rate_low = 1/(2*dimension - 1)
    rate_high = 2/(dimension)
    rate_high, rate_low = crit_value_bisection(rate_low, rate_high, attempts,
    ↪  0.001, dimension)
    print(f"The critical value in dimension {dimension} is likely between
    ↪  {rate_low:.6f} and {rate_high:.6f}.")
```